oop4fun.eu

2021-1-SK01-KA220-SCH-00027903

OOP4FUN: OBJECT ORIENTED PROGRAMMING FOR FUN

VODIČ ZA NASTAVNIKE SREDNJIH ŠKOLA

Urednici:
Michal Varga, Josef Rak,
Dušan Savić, Zlatko Stapić

Универзитет у Београду

Факултет организационих наука

OOP4FUN: OBJECT ORIENTED PROGRAMMING FOR FUN

Приручник за наставнике средњих школа

Уредници:

Michal Varga, Josef Rak, Dušan Savić, Zlatko Stapić

Аутори:

Peter Sedláček, Nika Kvaššayová, Jozef Kostolný, Michal Mrena, Patrik Rusnák, Peter Sobe, Ilija Antović, Miloš Milić, Tatjana Stojanović, Davor Fodrek, Lidija Kozina, Marko Mijač, Dijana Plantak Vukovac, Antonela Čižmešija, Dijana Peras, Goran Hajdin, Lea Masnec

Београд, новембар 2024.

IMPRESSUM

Прво издање

Наслов:
OOP4FUN: OBJECT ORIENTED PROGRAMMING FOR FUN - Priručnik za nastavnike srednjih škola

Уредници:
Michal Varga, Josef Rak, Dušan Savić, Zlatko Stapić

Аутори:
Peter Sedláček, Nika Kvaššayová, Jozef Kostolný, Michal Mrena, Patrik Rusnák, Peter Sobe, Ilija Antović, Miloš Milić, Tatjana Stojanović, Nina Turajlić, Davor Fodrek, Lidija Kozina, Marko Mijač, Dijana Plantak Vukovac, Antonela Čižmešija, Dijana Peras, Goran Hajdin, Lea Masnec

Издавач:
Универзитет у Београду
Факултет организационих наука
Јове Илића 154, 11000 Београд

За издавача:
dr Marko Mihić

Преводиоци:
Dušan Savić, Ilija Antović, Miloš Milić, Tatjana Stojanović, Nina Turajlić,
Snežana Bogdanović,Zoran Vučetić Dragan Grujović i Ana Stamenić

Графичко обликовање и прелом текста:
Nina Turajlić, Dušan Savić

Штампарија:
VIBBACO DOO

ISBN:
ISBN 978-86-7680-478-5 (штампано издање)

Тираж:
100 ком.
Место и година издања:
Београд, новембар 2024.

Финансирано средствима Европске уније.
Овај ūриручник објављен је на чешком, словачком, немачком, хрватском и српском језику

Подаци о пројекту

Назив пројекта :
Object Oriented Programming for Fun

Akronim projekta:
OOP4FUN

Broj ugovora:
2021-1-SK01-KA220-SCH-00027903

Пројектни координатор:
Žilinska univerzita v Žiline (Словачка)

Партнери на пројекту:

Sveučilište u Zagrebu (Хрватска)
Srednja škola Ivanec (Хрватска)
Univerzita Pardubice (Чешка Република)
Gymnázium Pardubice (Чешка Република)
Obchodna akademia Povazska Bystrica (Словачка)
Hochschule fuer Technik und Wirtschaft Dresden (Немачка)
Gymnasium Dresden-Plauen (Немачка)
Универзитет у Београду (Србија)
Гимназија Ивањица (Србија)

Више о пројекту доступно на: https://www.oop4fun.eu/

САДРЖАЈ

Списак табела

0. Уводне информације

Циљ овог приручника јесте да представи курикулум предмета са материјалима који ће помоћи наставницима у припреми наставе за поучавање ученика у решавању задатака везаних за програмирање користећи основе објектно-оријентисаног програмирања (ООП).

Основни концепти ООП су објашњени коришћењем програмског језика Јава, док је Greenfoot коришћен као развојно окружење. Јава је тренутно веома популаран и широко коришћен програмски језик у пракси. Greenfoot представља развојно окружење засновано на оквирима (енгл. *frame-based source code editor*) који користи Stride језик и који је пре свега намењен за развој компјутерских 2D игара. Greenfoot је развојно окружење које користи и графичку и текстуални нотацију што омогућава једноставан развој апликација које користе графичке елементе (објекте) који могу међусобно комуницирати. Теоријски концепти ООП-а су сведен на минимум, а ученици ће одмах започети са практичним радом на развоју једне рачунаске игре. Коришћењем овог уџбеника наставницима даје могућност да ђацима прикажу концепте ООП-а на забаван и креативан начин.

Ученици ће научити основе ООП кроз игру, у тиму или самостално. Научиће да међусобно заједно сарађују, осмишљавају и пројектују своју игру (апликацију). У оквиру курса концепти ООП-а као што су енкапсулација, наслеђивање или асоцијација се објашњавају постепено (лагано) кроз развој компјутерских 2 D игара, где се ови концепти користе на једноставан и интуитиван начин. Процес развоја компјутерске игре заснива се на тимском раду и практично примењује знања и вештине из других области информатике и сродних предмета (на пример мултимедијом). Пројектовање сваке компјутерске игре може бити креативно тако да ученици могу индивидуално проширити игру. Пројектовањем компјутерских игара, ђаци ће на практичним примерима усвојити и применити стечена знања на прави начин (у реалним апликацијама).

Курс је усмерен на увођење иновативног приступа поучавању ђака концептима ООП парадигме. ООП је данас доминантна парадигма за развој апликација, због чега је пожељно да ученици поседују знања и вештине из ове области. У оквиру предмета представљено је развојно окружење које користи различите облике уређивања изворног кôда (уређивање засновано на принципу оквира – енгл. frame-based editing, али и писањем изворног кôда), што омогућава поучавање ученика са различитим нивоима претходног програмерског знања. Својом једноставношћу и јасноћом, овај алат подржава брзо и интуитивно разумевање наставних тема, што позитивно утиче на ученике и њихову мотивацију.

Програмирањем интерактивних игара у Greenfoot развојном окружењу, ученик ће стећи знања и вештине које ће му омогућити да:

- идентификују проблем,
- идентификује објекте који су неопходни за решавање конкретног проблема
- идентификују класе, њихове атрибуте и методе,
- идентификује и правилно користи везе међу објектима (асоцијација, наслеђивање),
- дефинишу алгоритам за решавање проблема
- користи елементе изворног кôда (гранање, петље) за имплементацију дефинисаног алгоритма,
- ефикасно користи средства за отклањање грешака у изворном кôду,
- израде једноставну апликацију са графичким интерфејсом у Greenfoot окружењу.

Образовни исходи се могу сумирати на следећи начин:

- разумевање основних принципа објектно-оријентисаног програмирања,
- разумевање основа развоја алгоритама,
- разумевање синтаксе програмског језика *Java*,
- способност анализе извршавања програма на основу изворног кôда,
- способност развоја сопствених програма применом ООП-а.

Модеран приступ у осмишљацању предавања, посебно за основно и средње образовање, је дефинисање и дељење наставних сценарија (енг. *teaching scenario*). Наставни сценарији „се

перципирају као савремени педагошки приступ који омогућава индивидуализацију наставног процеса узимајући у обзир различите потребе ученика. Настава заснована на наставним сценаријима је фокусирана на релевантно знање и вештине за ученике, укључујући и потребе за дигитално друштво. Пажљиво планирање наставних сценарија може отклонити могуће замке и недостатке који би могли утицати на наставни процес.“

У контексту образовања, наставни сценарији представљају детаљна опис начина извођења наставног часа. Ови сценарији се често користе у обуци наставника за симулацију наставних ситуација из стварног света и стога се сматрају најбољим алатом за представљање наших иновативних идеја поучавања и учења. Будући да сценарији поучавања обично укључују информације о циљевима учења, садржају који се поучава, карактеристикама ученика, коришћеним методама поучавања и стратегијама евалуације, они се такође могу усагласити са елементима потребним за наше артефакте дизајна учења.

Како би се обезбедио структуриран приступ дефинисању наставних сценарија поучавања, дефинисали смо следећи шаблон (Табела 1) који би био попуњен конкретним подацима везаним за одређени сценарио учења. Следећа табела садржи кратак опис како дефинисати сваки елемент наставног сценарија.

Сходно томе, наш тематски план подучавања, као и алат за креирање наставних сценарија, доступан је на следећој веб адреси: **Детаљи курса (learning-design.eu)**. Материјали који прате овај приручник се могу преузети са *Moodle* платформе на адреси: **https://oop4fun.fon.bg.ac.rs/**

Табела 1. Предлог шаблона за документовање наставног сценарија

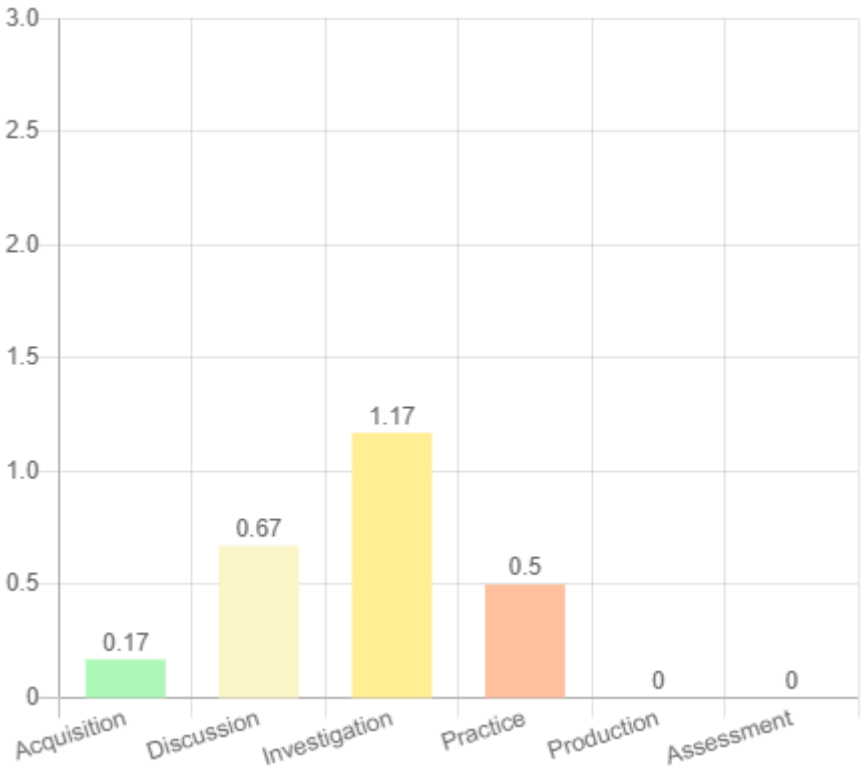
Назив	Одаберите описан назив који привлачи пажњу.
Образовни циљеви и исходи учења	Јасно наведите очекиване исходе учења. Шта би ученици требало да знају, разумеју или буду у стању даураде након савладавања сценарија?
Циљна група	Наведите циљну групу (и разред) којој је сценарио намењен, као и неопходно предзнање.
Временски план	Процените време које ће бити потребно да би се реализовао сценарио, укључујући и специфичне временске оквире за сваку од активности. На пример, уводна реч наставника (5 минута), самостално истраживање од стране ученика (10 минута), тимски рад на програмирању решења (20 минута), презентовање/дискусија (10 минута).
Материјали и ресурси	Наведите материјале, ресурсе и алате који ће бити потребни и наставницима и ученицима. То може да укључује уџбенике, материјале на Интернету, мултимедијалне ресурсе, софтвер итд.
Опис	• Представите наставни сценарио, објасните његови сврху и значај. • Дајте преглед основних активности у које ће ученици бити укључени како би достигли образовне циљеве. Укључите и детаље као што су дискусије, практичан рад, тимски рад, такмичења итд. • Прецизирајте начин на који ће ученици бити организовани, тј. да ли ће радити самостално или у тимовима. Колико ће тимови имати чланова? • Изложите пројекте/проблеме/задатке на којима ће ученици радити. Препоручује се примена приступа заснованих на проблему или приступа заснованих на пројекту. Поред тога, они би требало да одражавају ситуације из свакодневног живота. • Објасните начин на који ће се ученицима (тимовима) додељивати пројекти/проблеми/задаци. • Ако је предвиђен тимски рад, пружите детаљније информације о начину сарадње. • Наведите појединости свих активности у којима ученици треба да учествују. • У случају да се примењује нпр. приступ обрнуте учионице (енгл. *flipped classroom*), прецизирајте који део тематске целине ученици треба да самостално истраже.
Вредновање	Наведите детаљне информације о начину вредновања залагања и знања ученика. • Ко ће вршити вредновање: (1) наставници, (2) сам ученик (самовредновање), (3) други ученици (међусобно вредновање) • Који критеријуми ће се користити за вредновање? • Колико често ће се вршити вредновање? • итд.

Дељење решења	Објасните како ће ученици поделити своја решења са наставницима и другим ученицима. На пример, ученици (сви или само један део ученика) могу представити своја решења/резултате пред наставником и осталим ученицима, након чега може уследити поређење и дискусија.

1. УПОЗНАВАЊЕ СА *GREENFOOT* ОКРУЖЕЊЕМ

Greenfoot представља образовни алат (тј. развојно окружење са интегрисаним уређивачем/едитором програмском кôда и *2D* визуелизацијом) за развој компјутерских игара и симулација коришћењем програмског језика *Java.* Поред тога што је визуелан, *Greenfoot* је и интерактиван. *Greenfoot* омогућава писање самих програма у стандардном текстуалном формату (тј. коришћењем програмског језика *Java*), док са друге стране обезбеђује визуелизацију кôда и његовог извршавања.

1.1. Структура наставних активности и задаци



Слика 1. Расподела ангажовања ученика према типу учења за тематску целину „Упознавање са Greenfoot окружењем“

Задатак 1.1. Креирање *Greenfoot* пројекта

Креирајте нов пројекат, доделите му погодан назив (нпр. `TowerDefense` тј. одбрана куле) и сачувајте га на одговарајућој локацији.

[Commit: 9046f5353d857dcc112abd92d7b7170abcc64a80]

1.2. Наставни сценарији

Припремљен је један наставни сценарио за тематску целину „Упознавање са *Greenfoot* окружењем“.

1.2.1. СЦЕНАРИО: **Истраживање развоја рачунарских игара на креативан начин**

Табела 2. Истраживање развоја рачунарских игара на креативан начин

Назив	Истраживање развоја рачунарских игара на креативан начин
Образовни циљеви и исходи учења	Током ове лекције, поред тога што су успешно инсталирали *Greenfoot* и уверили се у његове могућности кроз примере пројеката, ученици су такође имали прилику да, кроз игру, испробају развојно окружење. Овај разиграни увод треба да постави тон за узбудљиво истраживање развоја игара, подстакне креативност, тимски дух и страственији приступ програмирању у *Greenfoot* окружењу.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Развијају се основне вештине програмирања, са нагласком на стицање знања о променљивима, функцијама, петљама и гранању.
Временски план	Укупно време потребно за реализацију сценарија: **90** минута. 1. Увод (5 минута) 2. Брзи изазов (10 минута) 3. Играње игара са наставником (30 минута) 4. Формирање тимова и додела пројектног задатака (5 минута) 5. Тимски рад и програмирање (30 минута) 6. Међусобно вредновање и повратне информације (10 минута) 7. Домаћи задатак (30 минута) 8. Вредновање (30 минута)
Материјали и ресурси	Веб страница *Greenfoot* окружења и упутство за његово преузимање. Примери које је припремио наставник. Ресурси на Интернету за проналажење додатних примера.
Опис	Кроз овај наставни сценарио ученици ће завирити у свет *Greenfoot*-а путем игре, истраживања и тимског рада. Након што наставник представи тематску јединицу и дефинише циљеве лекције, почиње брзи изазов. Ученици имају задатак (замишљен у виду такмичења) да пронађу неопходна упутства, а затим преузму и на својим рачунарима инсталирају *Greenfoot* (окружење које им је у датом тренутку још увек непознато). Три најбржа ученика добијају одређену награду као подстицај (значке, бодови, поени, слаткиши итд.) Следеће изненађење представља чињеница да ће током наредних 30 минута играти игре са наставником. Ово је блок који води наставник, а који је усмерен на отварање, компајлирање и покретање пар једноставних показних примера (пројекти почетног или средњег нивоа сложености). Ученици ће се тако упознати са основним

	елементима *Greenfoot* развојног окружења и основним поступцима рада са пројектним датотекама и ресурсима. Ученици ће, потом, бити распоређени у тимове (3-4 ученика по тиму) и биће им додељен једноставан задатак (**Задатак 1.1.**). Од тимови ће бити затражено да измене „нешто“ у одабраном показном примеру, са циљем да учине игру забавнијом или да у њу уведу неки фактор изненађења. Ученици заједнички раде на изменама. Уколико „разбуцају“ кôд тако да не могу више самостално да га исправе, могу замолити наставника за помоћ или кренути испочетка (поновним преузимањем почетне верзије кôда). Ово ће им помоћи да схвате зашто је пожељно користити неки систем за управљање верзијама кôда (нпр. *git*). Наставник бира тим (или два тима) који треба да представи своје решење осталим ученицима, како би га они вредновали и дали повратне информације. Наставник разговара са ученицима о резултатима. За домаћи задатак, сваки од ученика треба да потражи примере *Greenfoot* пројеката, а затим да одабере онај који му се највише допада и представи га остатку групе, тако што ће са њима поделити адресу пројекта, разлоге због којих га је одабрао и два до три снимка екрана развојног окружења и тока игре. Како би се увели елементи игре у наставу (енгл. *gamification*) и подстакла мотивације путем надметања, ученици треба да путем гласања изаберу три најбоља пројекта (при чему ученик не може гласати за свој пројекат). На крају се проглашавају победници и додељују им се одређене награде као подстицаји (значке, бодови, поени, слаткиши итд.)
Вредновање	Активности у оквиру ове лекције ће омогућити наставницима да дају ученицима формативне повратне информације базиране на њиховом учешћу у спроведеним дискусијама, али и на праћењу њиховог рада у окружењу обрнуте учионице (енгл. *flipped classroom*), као и рада у тиму. Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе. Међусобно вредновање ће се вршити *online*, као део домаћег задатка. Кроз ову активност ученици ће се подсетити важних аспеката лекције и биће подстакнути да инсталирају *Greenfoot* окружење. Разматрањем различитих примера, ученици ће лакше усвојити изложено градиво, што доприноси и повећању општег достигнућа постављених образовних циљева и исхода учења целе групе.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

1.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Увод (5 минута)

Циљ: Упознавање са програмским језицима (текстуалним и визуелним), интегрисаним развојним окружењима (енгл. *integrated development environment - IDE*) и изворним кôдом (енгл. *source code*).

Концепти за дискусију: програмски језици, интегрисана развојна окружења и изворни кôд.

Активности: Наставник даје кратак увод у лекцију и уводи концепте као што су:

- програмски језик
- интегрисано развојно окружење (алат за програмирање) и
- изворни кôд.

Наставник показује примере изворног кôда у различитим програмским језицима, као што су *Java*, *C* и *Python*, али и примере програма у *Scratch*-у или неком другом визуелном језику.

Наставник не залази „у дубину“, већ настоји да објасни ове концепте путем једноставног примера, нпр. покушава да упореди савладавање програмског језика са учењем матерњег или страног језика.

- Природни језици имају своју граматику у правопис. Исто важи и за програмске језике. Док пишемо, придржавамо се одређених граматичких и правописних правила, као и приликом писања програма у неком програмском језику.
- Када желимо да напишемо причу на нашем матерњем језику, можемо да користимо свеску и оловку као помоћне „алатке“. Аналогно томе, када пишемо програм у неком програмском језику, наша „алатка“ је интегрисано развојно окружење;
- Резултат процеса писања може бити прича написана на папиру, док резултат процеса програмирања представља програм који смо развили и који називамо изворни кôд.

Наставник може осмислити и сопствени уводни пример.

2. Брзи изазов (10 минута)

Циљ: Упознавање са *Greenfoot* окружењем и начином његовог инсталирања.

Концепти за дискусију: *Greenfoot* окружење, упутство за инсталирање.

Активности: Наставник ученицима даје задатак да открију шта је *Greenfoot*. Наставник тражи од ученика да пронађу упутство за његово инсталирање. Ученици раде на задатку, након чега им наставник демонстрира како да пронађу, преузму и покрену инсталацију *Greenfoot* окружења. Упутства за преузимање и инсталирање *Greenfoot* окружења су доступна на *Moodle* платформи.

3. Играње игара са наставником (30 минута)

Циљ: Покретање различитих *Greenfoot* пројеката .

Концепти за дискусију: *Greenfoot* пројекти: пројекти на вебу и самостални/независни (енгл. *standalone*) пројекти.

Активности: Наставник тражи од ученика да на Интернету пронађу примере пројекта развијених у *Greenfoot* окружењу. Док ученици раде на задатку, наставник надгледа њихов рад.

Наставник показује ученицима како да пронађу примере готових пројеката развијених у *Greenfoot* окружењу. На пример, у *Google* претраживачу пројекти се могу наћи навођењем кључних речи као што су „*Greenfoot Java project example*“ или слично.

Наставник објашњава да постоје две врсте *Greenfoot* пројеката:

- они који се могу покренути у веб прегледачу (енгл. *web browser*) и
- они који се могу преузети са Интернета, а затим отворити и покренути у самом *Greenfoot* окружењу.

Наставник представља различите врсте пројеката:

- пројекте који се могу покренути директно у веб прегледачу и
- пројекте који се, након преузимања, могу покренути у самом *Greenfoot* окружењу.

Наставник може да преузме неколико примера пројеката или да приступи пројектима који се налазе на одређеним адресама.

4. Формирање тимова и додела пројектног задатака (5 минута)

Циљ: Упознавање са учењем заснованом на пројектима путем једноставног пројектног задатка.

Концепти за дискусију: /.

Активности: Наставник формира тимове, одређује пројекат на којем ученици треба да раде и припрема пројектни задатак. Примери пројектних задатака се могу пронаћи на *Moodle* платформи. Наставник бира један репрезентативан пројекат (који не треба да буде претерано сложен) и за дати пројекат дефинише одговарајући пројектни задатак (тј. проблем) који ученици треба да реше.

5. Тимски рад и програмирање (30 минута)

Циљ: Овладавање начином креирања *Greenfoot* пројекта. Ученици раде на задатку који им је додељен.

Концепти за дискусију: креирање *Greenfoot* пројеката.

Активности: Наставник покреће *Greenfoot* окружење и показује ученицима како да креирају пројекат. Наставник задаје ученицима задатак да креирају *Greenfoot* пројекат.

> **Задатак 1.1. (Креирање *Greenfoot* пројекта):** Креирајте нов пројекат, доделите му погодан назив (нпр. `TowerDefense`, тј. одбрана куле) и сачувајте га на одговарајућој локацији.

Наставник наглашава да се пројекат (који је идентичан пројекту који су они управо креирали) може:

- преузети са *git*-а[1], уносом следеће наредбе:

[Commit: `9046f5353d857dcc112abd92d7b7170abcc64a80`]

- преузети са *git*-а, али у ZIP формату, са следеће адресе: **https://oop4fun.fon.bg.ac.rs/**

Наставник такође наглашава да ученици, током данашње лекције, неће даље радити на пројекту који су управо креирали (мада ће рад на њему наставити током наредних лекција), већ ће радити само на постојећим пројектима.

Ученици раде на задатку и решавају проблем.

[1] Неопходно је да *git* буде инсталиран на рачунару.

6. Међусобно вредновање и повратне информације (10 минута)

Циљ: Дискусија о предложеним решењима.

Концепти за дискусију: /.

Активности: Наставник надгледа рад ученика и, уколико је потребно, пружа им смернице (инструкције). Када ученици заврше са радом, наставник бира тим који ће представити своје решење. Наставник дискутује са ученицима о предложеном решењу.

7. Домаћи задатак (30 минута)

Циљ: Проучавање *Greenfoot* пројеката.

Концепти за дискусију: /.

Активности: Наставник дефинише задатак који ученици треба да реализују у једном од постојећих пројеката.

8. Вредновање (30 минута)

Циљ: Укључивање ученика у процес вредновања решења.

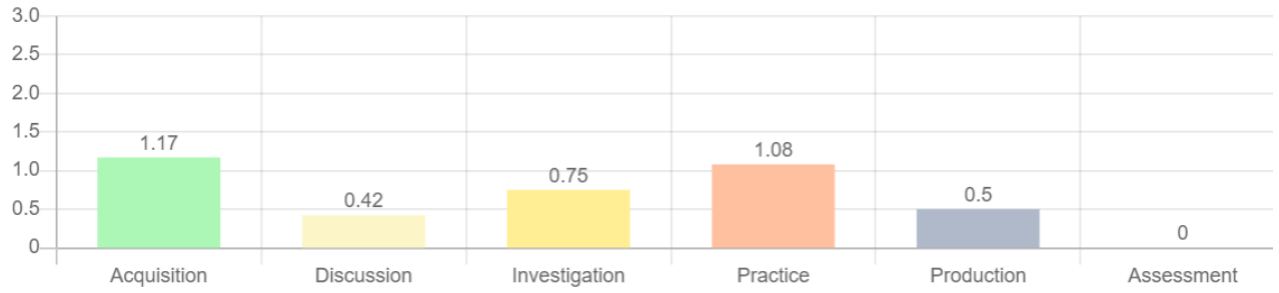
Концепти за дискусију: /.

Активности: Ученици представљају своје пројекте. Наставник тражи од свих ученика да учествују у вредновању представљених пројеката, нпр. тако што са њима дели линк ка *Google* упитнику коју треба да попуне. Ученици треба да, путем бодовања, изаберу три најбоља тима.

Након тога наставник даје осврт на представљене пројекте. Наставник дели своје утиске о раду ученика и указује на то да ли је задовољан, да ли су испунили или надмашили његова очекивања.

2. КЛАСЕ И ОБЈЕКТИ

2.1. Структура наставних активности и задаци



Слика 2. Расподела ангажовања ученика према типу учења за тематску целину „Класе и објекти“

Задатак 2.1. Идентификовање објеката

Идентификујте објекте у вашем окружењу и наведите њихове карактеристике, као и активности које могу да изврше. Да ли је могуће идентификовати објекте који немају карактеристике? Да ли је могуће идентификовати објекте који немају понашање? Да ли је могуће идентификовати нематеријалне објекте (тј. објекте које не можете физички додирнути)?.

Задатак 2.2. Припрема света

Преправите изворни кôд класе `MyWorld` (двоструким кликом на њу) како бисте дефинисали свет димензије **`12 x 24`** ћелија. Величина сваке појединачне ћелије треба да буде **`50`** пиксела.

[Commit: a593cd4a92d0fa0db78275614c3e41a2e96b4e57]

Задатак 2.3. Припрема слике за свет

Пронађите или креирајте одговарајућу слику за позадину света. Можете користити неку од понуђених слика или одабрати неку другу слику. За позадину можете користити једну велику слику која ће прекрити читаву површину света или једну мању слику која ће се понављати.

[Commit: 1184980643db082cfdd6bde9984bceaddf010d49]

Задатак 2.4. Дефинисање класе непријатеља

Уведите новог протагонисту – непријатеља. Непријатељ ће напредовати према играчевој кули (`Tower`) како би је најпре оштетио, а на крају и срушио. Дефинишите нову класу, доделите јој погодан назив (нпр. `Enemy`) и придружите јој одговарајућу слику.

[Commit: 4981400623729c3d112b54454b6e6151e18426bf]

Задатак 2.5. Креирање инстанце класе непријатеља

Креирајте инстанцу класе `Enemy`. Прегледајте интерно стање креиране инстанце. Креирајте још једну инстанцу класе `Enemy` и поставите је на неку другу позицију. Упоредите интерна стања те две инстанце.

Задатак 2.6. Позивање метода

Пошаљите поруку инстанци класе `Enemy` (позивом одговарајуће методе), како би се она преместила на позицију **`[12,6]`** и била окренута надоле. Какав ће утицај то имати на интерно стање дате инстанце?

2.2. Наставни сценарији

Припремљена су два наставна сценарија за тематску целину „Класе и објекти“.

2.2.1. СЦЕНАРИО: **Упознавање са класама и објектима кроз развој игара у *Greenfoot* окружењу**

Табела 3. Упознавање са класама и објектима кроз развој игара у Greenfoot окружењу

Назив	Упознавање са класама и објектима кроз развој игара у *Greenfoot* окружењу
Образовни циљеви и исходи учења	Након ове лекције, ученици би требало да разумеју основне концепте класе и објекта. Разумевање ових концепата се продубљује кроз развој игре у *Greenfoot* окружењу, чиме се подстиче креативност, тимски дух и страственији приступ програмирању.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Развијају се основне вештине програмирања, са нагласком на стицање знања о објектима и класама.
Временски план	Укупно време потребно за реализацију сценарија: **100** минута. *1*. Објекат (10 минута) *2*. Идентификација објеката и њихових особина (15 минута) *3*. Класа и инстанца (објекат) (15 минута) 4. Сналажење у *Greenfoot* окружењу (10 минута) *5*. Конструктор (10 минута) 6. Задатак **2.2.** (15 минута) 7. Постављање слике (10 минута) 8. Задатак **2.3.** (15 минута)
Материјали и ресурси	Веб страница *Greenfoot* окружења и упутство за његово преузимање. Примери које је припремио наставник. Ресурси на Интернету за проналажење додатних примера.
Опис	Кроз овај наставни сценарио ученици ће се упознати са принципима програмирања који се односе на класе и објекте, а из перспективе развоја игара у *Greenfoot* окружењу. Лекција почиње десетоминутним уводом од стране наставника, који уводи ученике у свет објектно-оријентисаног програмирања тако што им приближава концепт објекта и његових карактеристика коришћењем примера из свакодневног живота. Потом започиње брзи изазов од 10 минута, током којег ученици треба да, у задатом текстуалном опису, препознају објекте и њихове карактеристике. Након тога следи петоминутни блок, током којег наставник, заједно са ученицима, разматра решење датог задатка. Током наредних 15 минута наставник објашњава разлику између класе и објекта и уводи, на највишем нивоу апстракције, концепт наслеђивања.

	Наставник затим покреће *Greenfoot* окружење и током наредних 10 минута упознаје ученике са основим класама: `World` (свет), `Actor` (протагониста) и `MyWorld` (мој свет). Наставник представља и објашњава изворни кôд ових класа, који се аутоматски генерише када се креира пројекат. У оквиру наредног десетоминутног блока, наставник задаје **Задатак 2.2.**, а затим показује ученицима како да припреме свет за апликацију коју треба да развију. Током наредних 25 минута, наставник најпре објашњава како се подешава слика за одређену класу, а затим заједно са ученицима решава **Задатак 2.3.**
Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

2.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Објекат (10 минута)

Циљ: Разумевање концепта објекта кроз примере из свакодневног живота**.**

Концепти за дискусију: објекти и карактеристике објеката.

Активности: Наставник уводи појам објекта. Наставник треба да приближи овај концепт ученицима коришћењем примера из свакодневног живота. На пример, наставник може да пита ученике како се зову, колико су високи, када су рођени, која им је боја очију итд. Наставник поставља оваква питања како би навео ученике да размисле о томе како се разликују једни од других, припремајући их на тај начин за наредно питање: „По чему се ученици међусобно разликују?“.

Наставник закључује да свака особа (наставник, ученици) има одређене карактеристике по којима се разликује од других и наглашава да је свако од нас заправо представља „један“ објекат. Наставник објашњава концепт особине као карактеристике коју свако од нас има, заједно са конкретном вредношћу одређене карактеристике која је специфична за сваког појединца. Сходно томе, објекти се међусобно разликују по вредностима њихових карактеристика.

2. Идентификација објеката и њихових особина (15 минута)

Циљ: Идентификовање објеката и њихових особина**.**

Концепти за дискусију: објекти и њихове особине.

Активности: Наставник тражи од ученика да, у датом тексту, идентификују објекте и њихове особине.

3. Класа и инстанца (објекат) (15 минута)

Циљ: Стицање знања о класама и инстанцама тј. објектима. Разликовање појмова класе и објекта.

Концепти за дискусију: класа, објекат (инстанца).

Активности: Наставник објашњава разлику између концепта класе и концепта инстанце класе (тј. објекта) коришћењем примера из свакодневног живота, и уводи, на највишем нивоу апстракције, концепт наслеђивања. Наставник поставља питања ученицима, како би их навео да увиде разлику између класе и објекта. Наставник усмерава дискусију о идентификованим објектима и њиховој класификацији.

4. Сналажење у *Greenfoot* окружењу (10 минута)

Циљ: Увођење класе `World` (свет) и креирање њене инстанце.

Концепти за дискусију: инстанца света тј. класе `World`.

Активности: Наставник покреће *Greenfoot* окружење и креира једноставан пројекат. Наставник кроз овај пројекат објашњава ученицима како да уведу разматране концепте. Наставник указује на то да сваки пројекат, који је креиран у *Greenfoot* окружењу, садржи три класе: `World` (свет), `Actor` (протагониста) и `MyWorld` (мој свет). Наставник затим представља класу `MyWorld` (мој свет) и разјашњава њену улогу.

Наставник истиче то да позадина (енгл. *background*) било које *Greenfoot* апликације заправо представља матрицу која се састоји од појединачних ћелија (енгл. *cells*). Након тога, наставник демонстрира ученицима како се може дефинисати величина позадине (тј. димензија матрице), као и величина сваке појединачне ћелије матрице. Наставник објашњава да се сваки објекат који се приказује на табли („сцени“) налази у једној од ћелија. Наставник показује изворни кôд сваке од класа.

5. Конструктор (10 минута)

Циљ: Увођење концепта конструктора.

Концепти за дискусију: конструктор.

Активности: Наставник приказује изворни кôд и уводи концепт конструктора.

6. Задатак **2.2.** (15 минута)

Циљ: Учешће ученика у раду на пројектном задатку.

Концепти за дискусију: класа `World`.

Активности: Наставник задаје ученицима задатак:

> **Задатак 2.2. (Припрема света):** Дефинишите свет димензије **12 x 24** ћелија. Величина сваке појединачне ћелије треба да буде **50** пиксела.

Док ученици решавају задатак наставник прати њихов рад, а затим бира једног ученика који треба да представи и објасни своје решење.

Опис решења:

> Преправите изворни кôд класе `MyWorld` (двоструким кликом на њу) како бисте дефинисали свет димензије **12 x 24** ћелија. Величина сваке појединачне ћелије треба да буде **50** пиксела.

[Commit: a593cd4a92d0fa0db78275614c3e41a2e96b4e57]

7. Постављање слике (10 минута)

Циљ: Постављање слике света у *Greenfoot* пројекту.

Концепти за дискусију: слика за класу `World` (свет).

Активности: Наставник указује на то да позадина *Greenfoot* пројекта (тј. света) може бити и слика. Наставник објашњава ученицима да се као позадина може користити било једна слика која ће прекрити читаву површину света, било слика која одговара димензијама ћелије.

Наставник показује ученицима како да подесе слику за класу `MyWorld`. Ученици прате упутства наставника и раде заједно са њим, корак по корак.

8. Задатак **2.3.** (15 минута)

Циљ: Савладавање начина подешавања позадине света у *Greenfoot* пројекту.

Концепти за дискусију: позадина класе `World` (свет).

Активности: Наставник задаје ученицима задатак.

> **Задатак 2.3. (Припрема слике за свет):** Пронађите или креирајте одговарајућу слику и поставити је као позадину света.

Наставник ученицима појашњава задатак. Наставник може да покаже ученицима довршене, унапред припремљене, примере. Наставник затим ученицима даје адресу или *git* наредбу путем које треба да преузму пројекат у оквиру којег треба да раде на задатку. Ученици овај прелиминаран пројекат могу да преузму и из *git* репозиторијума.

Ученици решавају задатак самостално или у групама. Наставник прати рад ученика. Након тога, наставник решава задатак, корак по корак, док ученици прате његова упутства.

Опис решења:

- Пронађите или креирајте одговарајућу слику за позадину света. Можете користити неку од понуђених слика (десним кликом на класу `MyWorld` и избором опције `Set image...` у њеном контекстном менију) или одабрати неку другу слику (копирајте жељену слику у поддиректоријум `images` који се налази у оквиру директоријума вашег пројекта, а затим је изаберите на претходно описан начин).
- Као позадину можете користити једну већу слику која ће прекрити читаву површину света (израчунајте потребну величину слике на основу димензија света) или једну мању слику која ће се понављати (употребите квадратну слику величине једне ћелије).

[Commit: `1184980643db082cfdd6bde9984bceaddf010d49`]

2.2.2. СЦЕНАРИО: **Креирање класа и објеката кроз развој игара у *Greenfoot* окружењу**

Табела 4. Креирање класа и објеката кроз развој игара у Greenfoot окружењу

Назив	Креирање класа и објеката кроз развој игара у *Greenfoot* окружењу
Образовни циљеви и исходи учења	Након ове лекције, ученици би требало да разумеју основне концепте класе и објекта, својства класе и методе.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Развијају се основне вештине програмирања, са нагласком на стицање знања о објектима, класама, својствима класа и методама.
Временски план	Укупно време потребно за реализацију сценарија: **130** минута. *1*. Основни појмови (25 минута) *2*. Задатак **2.4.** (10 минута) *3*. Задатак **2.5.** (30 минута) *4*. Интерфејс (5 минута) *5*. Поруке и методе (15 минута) *6*. Задатак **2.6.** (30 минута) *7*. Обнављање теоријских концепата (15 минута)
Материјали и ресурси	Веб страница *Greenfoot* окружења и упутство за његово преузимање. Примери које је припремио наставник. Ресурси на Интернету за проналажење додатних примера.
Опис	Кроз овај наставни сценарио ученици постепено, путем неколико структурираних задатака, продубљују знање о концептима објектно-оријентисаног програмирања. Ученици ће најпре посветити 25 минута дефинисању класе `Enemy` (**Задатак 2.4.**), а затим следи 15-минутни задатак који је фокусиран на дефинисање атрибута и метода дате класе. Следећих 30 минута ће ученици посветити креирању објекта, тј. инстанце класе `Enemy` (**Задатак 2.5.**), примењујући стечено знање о креирању и иницијализацији објеката. У оквиру наредног петоминутног блока разматра се концепт интерфејса класе, са нагласком на дефинисању операција које објекат може да изврши. Током следећих 15 минута уводе се концепти порука и метода и ученици уче о томе како објекти међусобно комуницирају путем позива одговарајућих метода. Наредни блок од 30 минута је посвећен практичној примени, те ученици решавају **Задатак 2.6.** како би утврдили знање о понашању објеката. У последњем блоку од 15 минута даје се теоријски осврт, односно рекапитулација кључних концепата као што су класе, објекти, инстанце, интерна стања, интерфејси, поруке и методе, чиме се осигурава темељно савладавање постављених образовних циљева.

Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

2.2.2.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Основни појмови (25 минута)

Циљ: Утврђивање знања о класама, њиховим својствима и објектима.

Концепти за дискусију: класа, својства и објекти.

Активности: Током увода у лекцију наставник се, заједно са ученицима, осврће на претходно обрађене концепте. Наставник, кроз дискусију, појашњава појмове класе, својства класе и објекта.

Наставник задаје ученицима писани задатак да, у задатом тексту, идентификују класе, њихова својства и њима одговарајуће објекте, заједно са припадајућим вредностима.

Наставник бира ученика који треба да изложи своје решење. Остали ученици учествују у овој активности тако што изражавају своје мишљење.

Наставник постепено објашњава како се дефинише класа у *Greenfoot* окружењу. Ученици прате упутства наставника и раде заједно са њим, корак по корак.

2. Задатак **2.4.** (10 минута)

Циљ: Учешће ученика у раду на пројектном задатку.

Концепти за дискусију: дефинисање класе.

Активности: Наставник задаје ученицима задатак:

> **Задатак 2.4. (Дефинисање класе непријатеља):** Уведите новог протагонисту – непријатеља. Непријатељ ће напредовати према играчевој кули (`Tower`) како би је најпре оштетио, а на крају и срушио. Дефинишите нову класу, доделите јој погодан назив (нпр. `Enemy`) и придружите јој одговарајућу слику.

Наставник ученицима појашњава задатак. Наставник може да преузме унапред припремљен пример довршеног пројекта и покаже ученицима шта се од њих очекује. Наставник затим ученицима даје адресу или *git* наредбу путем које треба да преузму пројекат у оквиру којег треба да раде на задатку. Ученици овај прелиминаран пројекат могу да преузму и из *git* репозиторијума.

Ученици решавају задатак самостално или у групама. Наставник прати рад ученика. Након тога, наставник решава задатак, корак по корак, док ученици прате његова упутства.

Опис решења:

> Дефинишите нову класу, као изведену класу класе `Actor` (десним кликом на класу `Actor` и избором опције `New subclass...` у њеном контекстном менију). Доделите јој погодан назив (нпр. `Enemy`) и придружите јој одговарајућу слику.
>
> **[Commit: 4981400623729c3d112b54454b6e6151e18426bf]**

Наставник треба да упути ученике у то да постоје одређене конвенције којих се треба придржавати када је реч о додељивању назива класама (нпр. да је пожељно да назив класе почне великим словом), али и да се осврне на целокупну конвенцију у погледу именовања у програмском језику *Java*.

3. Задатак **2.5.** (30 минута)

Циљ: Разумевање концепта стања објекта. Овладавање начином креирања инстанце класе у *Greenfoot* окружењу.

Концепти за дискусију: стање објекта, инстанца.

Активности: Наставник објашњава концепт стања објекта путем једноставног примера. На пример, наставник стоји на одређеној удаљености од улаза у учионицу. Ова раздаљина одређује његов тренутни положај у учионици, с тим да се корачањем унапред, или уназад, сама раздаљина мења. Сходно томе, положај наставника у односу на улаз би се могао исказати путем променљиве чија ће се вредност мењати током времена. Дакле, положај наставника се дефинише растојањем од улаза, чиме се илуструје како стање неког објекта (у овом случају наставника) карактерише вредност његовог атрибута (растојање) у датом тренутку.

Наставник излаже задатак:

> **Задатак 2.5. (Креирање инстанце класе непријатеља)**: Креирајте инстанцу класе `Enemy`. Прегледајте интерно стање креиране инстанце. Креирајте још једну инстанцу класе `Enemy` и поставите је на неку другу позицију. Упоредите интерна стања те две инстанце.

Наставник отвара последњу верзију пројекта, а ученици прате упутства наставника и раде заједно са њим, корак по корак.

Опис решења:

> Наставник креира инстанцу класе `Enemy` (десним кликом на класу `Enemy` и избором опције `new Enemy()` у њеном контекстном менију), а затим је поставља на сцену левим кликом на жељену позицију. Наставник затим демонстрира како се може прегледати интерно стање креиране инстанце (десним кликом на дату инстанцу на сцени и избором опције `Inspect` у њеном контекстном менију).
>
> Наставник тражи од ученика да креирају још једну инстанцу класе, али да је поставе на неку другу позицију, како би упоредили интерна стања те две инстанце.

4. Интерфејс (5 минута)

Циљ: Увођење концепта интерфејса као скупа активности које се могу извршити над одређеним објектом.

Концепти за дискусију: интерфејс.

Активности: Наставник уводи концепт интерфејса путем једноставних примера. На пример, уколико се посматра нека особа и активности које она обавља током дана (као што су буђење, доручковање, одлазак на посао итд.), без залажења у појединости (како ће се пробудити, где доручкује и шта једе за доручак и како иде на посао), скуп датих активности се може поистоветити са интерфејсом. Путем интерфејса се прописује које активности објекти одређене класе могу извршавати, али се не прецизира начин на који се оне извршавају (тј. не прописује се на који начин ће неко бити пробуђен – будилник или позив, шта ће јести за доручак и да ли ће ићи на посао пешке, јавним превозом или колима).

5. Поруке и методе (15 минута)

Циљ: Увођење концепта методе.

Концепти за дискусију: методе.

Активности: Наставник уводи концепт методе. Како би појаснио дати концепт, наставник успоставља везу између својства (карактеристике, атрибута) и начина промене вредности датог својства.

Пример 1: Ако се посматра класа **Особа** и њено својство **старост**, може се уочити да се вредност овог својства се конзистентно повећава за један, сваке године, на исти дан. Са друге стране, својства као што су **висина** и **тежина**, ће се чешће мењати јер деца расту и тежина им варира.

Пример 2: Ако се посматра кретање неке особе од тачке **А** до тачке **Б** и оно опише путем корака – нпр. корак напред, окрет за 45 степени на лево, 8 корака напред, окрет за 30 степени на десно и још 5 корака напред – ови појединачни кораци, тј. радње, се могу објединити у такозвану методу.

Наставник показује ученицима како да у *Greenfoot* окружењу пронађу методе одређене класе које могу позвати над неким објектом дате класе.

Наставник ученицима показује методе које су дефинисане у класи `Actor`. Наставник затим демонстрира како се позивају методе над неким објектом.

6. Задатак **2.6.** (30 минута)

Циљ: Овладавање начином позивања метода над објектом.

Концепти за дискусију: позивање метода.

Активности: Наставник излаже задатак:

> **Задатак 2.6. (Позивање метода)**: Пошаљите поруку инстанци класе `Enemy` (позивом одговарајуће методе), како би се она преместила на позицију `[12,6]` и била окренута надоле. Какав ће утицај то имати на интерно стање дате инстанце?

Наставник отвара последњу верзију пројекта, а ученици прате упутства наставника и раде заједно са њим, корак по корак.

Опис решења:

> Наставник позива одговарајућу методу над инстанцом класе `Enemy` (десним кликом на дати објекат на сцени и избором опције *`inherited from Actor`* у њеном контекстном менију, а затим избором методе `setLocation(int x, int y)`). Наставник објашњава ученицима шта ће се догодити након позива методе и како ће се променити интерно стање дате инстанце.

Наставник показује ученицима како се дефинишу методе. Наставник дефинише методу `setPosition(int x, int y)` која омогућава постављање инстанце класе `Actor` на одређену позицију (задату путем њених координата). Наставник указује на то да је ова метода заправо еквивалента методи `setLocation(int x, int y)`, те да је пре дефинисање неке нове методе пожељно проверити да ли је таква метода већ дефинисана, како би се избегло дуплирање. Наставник наглашава да назив методе треба да јасно одражава њену сврху и понашање, тј. да се оно може одмах разумети искључиво на основу назива. Наставник такође истиче да би назив методе требало да буде концизан и даје примере добро дефинисаних, лоше дефинисаних и неправилно дефинисаних метода. Методе које корисник може да изврши (тј. позове) над неким објектом видљиве су када се на дати објекат кликне десним тастером миша.

Наставник објашњава ученицима како да дефинишу методу. Ученици прате упутства наставника и раде заједно са њим, корак по корак.

Наставник тражи од ученика да дефинишу методу која ће обезбедити да се `Actor` спусти (смањивањем вредности **y** координате), као и методу која ће обезбедити да се `Actor` попне (повећањем вредности **y** координате). Наставник прати рад ученика и, уколико је потребно, усмерава сваког појединачно.

7. Обнављање теоријских концепата (15 минута)

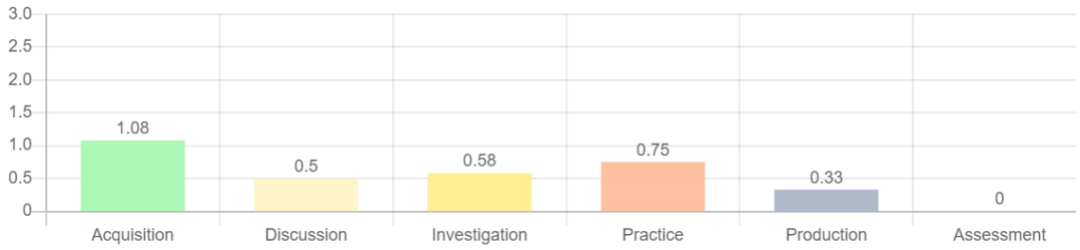
Циљ: Резимирање целокупне области

Концепти за дискусију: класа, објекат, инстанца, интерно стање, идентитет, порука, метода.

Активности: Наставник даје резиме лекције и осврће се заједно са ученицима, на све претходно обрађене концепте.

3. АЛГОРИТМИ

3.1. Структура наставних активности и задаци



Слика 3. Расподела ангажовања ученика према типу учења за тематску целину „Алгоритми“

Задатак 3.1. Дефинисање једноставног алгоритма

Напишите на папиру поступак помоћу којег ћете описати како пешак прелази улицу.

Задатак 3.2. Дефинисање општег алгоритма

Дефинишите општи алгоритам за припрему топлог напитка. Размислите о томе какви треба да буду улазни подаци како би дати алгоритам био општи.

Задатак 3.3. Позивање метода

Додајте наредбу у тело методе `act()` како бисте померили инстанцу `Enemy` класе за два корака унапред. Затим креирајте додатне инстанце класе `Enemy` и позовите ову методу над сваком од инстанци. Да ли сте очекивали такав ефекат?

[Commit: `7ba327ebeba6a13be68d9d21cc7e74b0da376132`]

Задатак 3.4. Документовање метода

Додајте коментар за документовање методе `act()`.

[Commit: `68b1c82c7df2c7826f2d3f78373498569adab7e9`]

Задатак 3.5. Документовање класа

Измените коментар за документовање `Enemy` класе. Додајте верзију класе и аутора, а затим уочите промене до којих је дошло у генерисаној *HTML* страници.

[Commit: `1a7a9f83c5271a7c0dfa46ce3b1ee65682b0c5e5`]

Задатак 3.6. Преглед документације

Истражите прозор са документацијом.

Задатак 3.7. Проучавање контрола *Greenfoot* окружења

Испробајте дугмад у главном прозору *Greenfoot* окружења. Креирајте више инстанци класе `Enemy`. Кликните на дугме `Act` - шта се догађа? Кликните на дугме `Run` - шта се догађа? Након што први пут кликните на дугме `Run`, кликните на дугме `Pause` - шта се догађа? Какав ефекат има померање клизача `Speed` на позивање `act()` методе након клика на дугме `Run`? Шта се догађа када кликнете на дугме `Reset`?

3.2. Наставни сценарији

Припремљена су два наставна сценарија за тематску целину „Алгоритми“.

3.2.1. СЦЕНАРИО: **Увод у алгоритме и алгоритамски начин размишљања**

Табела 5. Увод у алгоритме и алгоритамски начин размишљања

Назив	Увод у алгоритме и алгоритамски начин размишљања
Образовни циљеви и исходи учења	Након ове лекције, ученици би требало да су суштински разумели концепт алгоритма, усвојили алгоритамски начин размишљања, овладали вештинама пројектовања и имплементације основних алгоритама, и требало би да умеју да примене алгоритамске концепте за успешно решавање различитих проблема.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује и познавање концепата променљивих, функција, петљи и гранања, развијену способност логичког размишљања и решавања проблема. Потребно је и да ученици буду упознати и са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **90** минута. 1. Увод у једноставне алгоритме, алгоритам као низ корака (15 минута) 2. Задатак **3.1.** (20 минута) 3. Алгоритам и његове особине (15 минута) 4. Задатак **3.2.** (25 минута) 5. Развој алгоритама (15 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Кроз овај наставни сценарио ученици ће ући у свет алгоритама и алгоритамског начина размишљања. Лекција почиње уводом од 15 минута, чији је циљ упознавање ученика са основим аспектима алгоритама, са нагласком на њихов значај када је реч о решавању проблема. Након увода, ученици ће током 20 минута решавати **Задатак 3.1.** у оквиру којег ће се од њих очекивати да дефинишу једноставан алгоритам за решавање једног конкретног проблема. Ова активност омогућава ученицима да практично примене уведене концепте и усавршe своје алгоритамске вештине. Затим следи блок од 15 минута посвећен дискусији о алгоритмима и њиховим особинама. Теме које се обрађују укључују исправност, ефикасност и скалпирање, уз истицање значаја јасних и прецизних упутстава приликом пројектовања алгоритма. Како би продубили разумевање, ученици ће провести наредних 25 минута развијајући општи алгоритам за донекле сложенији проблем (**Задатак 3.2.**). Овај

	задатак треба да подстакне ученике на апстрактније и критичко размишљање, као и на примену алгоритамских принципа приликом решавања проблема из свакодневног живота. У последњем блоку од 15 минута, ученици ће се усредсредити на развој алгоритама, анализу и унапређење својих алгоритама. Циљ је да се препознају могућа побољшања, оптимизује ефикасност и осигура стабилност алгоритма. Током целе лекције, ученици ће радити самостално или у мањим групама, како би се подстакла сарадња и заједничко учење. Активним учешћем у дефинисању и анализи алгоритама, ученици развијају критичко мишљење и способност алгоритамског решавања проблема. До краја лекције, ученици ће стећи дубље разумевање концепта алгоритма и усвојити алгоритамски начин размишљања, што ће их опремити вештинама неопходним за решавање сложених проблема на систематичан и ефикасан начин.
Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

3.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Увод у једноставне алгоритме, алгоритам као низ корака (15 минута)

Циљ: Увођење концепта алгоритма**.**

Концепти за дискусију: основе алгоритама.

Активности: Наставник уводи концепт алгоритма путем примера из свакодневног живота. На пример, наставник може да пита ученике о њиховим јутарњим навикама, тј. шта раде од тренутка када се пробуде, па до тренутка када дођу у школу. Затим може да их пита да ли знају како се припрема пица или топли сендвич, односно да ли знају рецепт за припрему неког јела или колача.

Наставник повезује процес припреме јела или колача са развојем програма и наглашава да баш као што постоје рецепти за припрему јела, тако постоје и „рецепти“ за развој програма који се називају алгоритми.

Наставник закључује да алгоритам представља скуп корака који прописује, тј. дефинише, како се програм извршава.

Наставник даље објашњава да кораци не морају нужно да се извршавају узастопно, тј. један за другим, већ да извршавање одређених корака може зависити и од испуњености неког услова. Наставник тражи од ученика да наведу један такав пример (на пример, уколико дефинишемо алгоритам за припрему топлог сендвича, а немамо један од састојака, као што је шунка, али имамо сличан састојак, онда можемо или да га искористимо као замену или можемо да одемо у продавницу и купимо састојак који нам недостаје).

Наставник објашњава ученицима да се одређени кораци у алгоритму могу и поновити неколико пута. Наставник тражи од ученика да дају примере алгоритама у којима се одређени кораци понављају више пута.

2. Задатак **3.1.** (20 минута)

Циљ: Дефинисање једноставног алгоритма**.**

Концепти за дискусију: алгоритам.

Активности: Наставник задаје ученицима задатак.

> **Задатак 3.1. (Дефинисање једноставног алгоритма):** Напишите на папиру поступак помоћу којег ћете описати како пешак прелази улицу.

Наставник не појашњава додатно задатак, али прати начин на који ученици размишљају и како решавају задатак. Уколико неки ученик постави питање, које је од значаја за опис упутства за прелазак улице, наставник га похваљује и истиче зашто је дата информација значајна.

Након одређеног времена, наставник пита ученике да ли су обратили пажњу на то где пешак прелази улицу, тј. да ли прелази улицу на месту које је обележено као пешачки прелаз или не. Такође, наставник пита ученике да ли су обратили пажњу на то да ли постоји семафор на прелазу.

На крају, наставник бира неколико ученика који треба да прочитају своја упутства за прелазак улице.

3. Алгоритам и његове особине (15 минута)

Циљ: Упознавање са особинама алгоритама

Концепти за дискусију: особине алгоритама.

Активности: Наставник упознаје ученике са особинама алгоритама. Наставник објашњава да се алгоритми могу и графички приказати, и показује неколико примера.

4. Задатак **3.2.** (25 минута)

Циљ: Дефинисање општег алгоритма

Концепти за дискусију: дефинисање алгоритама

Активности: Наставник задаје ученицима задатак.

Задатак 3.2. (Дефинисање општег алгоритма): Дефинишите општи алгоритам за припрему топлог напитка. Размислите о томе какви треба да буду улазни подаци како би дати алгоритам био општи.

5. Развој алгоритама (15 минута)

Циљ: Овладавање начином развоја алгоритама

Концепти за дискусију: алгоритам.

Активности: Наставник објашњава ученицима да, и у математици, постоје одређени алгоритми који се користе за решавање проблема. Наставник пита ученике да ли могу дати неки такав пример.

Пример који наставник представља ученицима се односи на израчунавање вредности сложенијег аритметичког израза, са неколико математичких операција, при чему је неопходно водити рачуна о њиховом приоритету.

Наставник такође наводи и друге примере, као што је скапање комада намештаја, који је испоручен са упутством за склапање. Други пример може да се односи на упутства која добијамо од *GPS* уређаја када желимо да, користећи навигацију, стигнемо из тачке **А** у тачку **Б**.

Наставник тражи од ученика да осмисле, и напишу на папиру, сопствени алгоритам, након чега неки од ученика представљају своја решења.

3.2.2. СЦЕНАРИО: **Авантуре са *Greenfoot*-ом: Разоткривање начина позивања метода у програмском језику *Java*, документовања кôда и управљања током програма.**

Табела 6. Авантуре са Greenfoot-ом: Разоткривање начина позивања метода у програмском језику Java, документовања кôда и управљања током програма.

Назив	Авантуре са *Greenfoot*-ом: Разоткривање начина позивања метода у програмском језику *Java*, документовања кôда и управљања током програма.
Образовни циљеви и исходи учења	Након ове лекције, ученици би требало да су суштински разумели начин позивања метода, уз посебан нагласак на методе *Greenfoot* окружења: `act()` и `move(int)`. Требало би да умеју да користе кључну реч `this` када желе да референцирају актуелни објекат одређене класе. Поред тога, требало би да су у потпуности овладали позивањем метода класе и да су разумели синтаксу и параметре које је неопходно обезбедити приликом позива методе. Требало би да су такође развили способност примене техника позивања метода у циљу ефективног решавања задатака везаних за развој интерактивни игара. Даље, ученици би требало да су увидели значај документовања кôда и требало би да су способни да ефикасно документују *Java* кôд, и то на јасан и читљив начин. Коначно, требало би да су овладали контролама *Greenfoot* окружења које омогућавају прецизну манипулацију и интеракцију са елементима игре како би се осигурао занимљив и функционалан ток игре.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује и познавање концепата петљи и гранања. Потребно је и да ученици буду упознати и са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **110** минута. 1. Метода `act()` (10 минута) 2. Метода `move(int)` (20 минута) 3. Кључна реч `this` (5 минута) 4 .Задатак **3.3.** (10 минута) 5. Аутоматско довршавање кôда (5 минута) 6. Значај документовања програмског кôда (15 минута) 7. Задатак **3.4.** (5 минута) 8. Задатак **3.5.** (5 минута) 9. Задатак **3.6.** (10 минута) 10. Задатак **3.7.** (20 минута) 11. Дискусија: Алгоритам, особине, развој алгоритама, *Greenfoot* дугмад (5 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.

Опис	Кроз овај наставни сценарио ученици ће започети свој пут ка савладавању начина позивања метода у програмском језику *Java*, документовања кôда и управљања током програма у *Greenfoot* окружењу. Лекција почиње блоком од 10 минута који је посвећен разумевању сврхе методе `act()`, а затим је током наредних 20 минута фокус на појашњавању методе `move(int)`, два кључна елемента за развој програма у *Greenfoot* окружењу. Ученици ће затим провести 5 минута разматрајући значај кључне речи `this` када се жели референцирати актуелни објекат у контексту одређене класе. Након тога, ученици ће током 10 минута решавати **Задатак 3.3.** који има за циљ увежбавање начина позивања метода, уз коришћење одговарајуће синтаксе и параметара. Следи петоминутно упознавање са опцијом аутоматског довршавања кôда у *Greenfoot* окружењу, са нагласком на повећање ефикасности програмирања. Током наредних 15 минута ученици треба да схвате значај документовања кôда, као и то да јасна и концизна документација резултује бољом читљивошћу и лакшим одржавањем кôда. Следи петоминутни задатак (**Задатак 3.4.**) у оквиру којег се од ученика тражи да документују свој кôд, водећи рачуна о томе да документација треба да буде разумљива, не само њима, већ и другима. Надовезујући се на овај задатак, ученици ће провести још 5 минута додајући детаље у документацију свог кôда (**Задатак 3.5.**). Током следећег десетоминутног блока, ученици имају задатак да прегледају и прочитају документацију других ученика, како би стекли увид у различите приступе и стилове програмирања (**Задатак 3.6.**). Пре завршне петоминутне дискусије, ученици ће провести 20 минута проучавајући контроле *Greenfoot* окружења које омогућавају управљање током извршавања програмом и прецизну манипулацију и интеракцију са елементима игре како би се осигурао занимљив и функционалан ток игре (**Задатак 3.7.**). Током целе лекције, ученици ће радити самостално или у мањим групама, како би се подстакла сарадња и заједничко учење. Активним учешћем у дефинисању и анализи алгоритама, ученици развијају критичко мишљење и способност алгоритамског решавања проблема. До краја лекције, ученици ће стећи дубље разумевање начина позивања метода у програмском језику *Java*, документовања кôда и управљања током извршавања програма у *Greenfoot* окружењу, али и развити вештине кључне за развој игара, али и у другим доменима.
Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

3.2.2.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Метода `act()` (10 минута)

Циљ: Разумевање сврхе методе `act()` и тога како се нека метода позива из друге метода.

Концепти за дискусију: метода `act()`.

Активности: Наставник отвара последњу верзију `TowerDefense` пројекта. Наставник поставља инстанцу `Enemy` класе на позицију `0`, `0`. Наставник пита ученике шта мисле да ће се догодити када се позове `act()` метода над инстанцом класе `Enemy`. Да ли је то очекивано понашање?

Наставник треба да замоли ученике да му помогну да реши следећи задатак:

> **Задатак:** Потребно је померити инстанцу `Enemy` класе за два корака унапред, када се позове `act()` метода.

Наставник показује ученицима како да реше овај задатак (додавањем позива `move(int)` методе у `act()` методу), док ученици прате упутства наставника и дају своје предлоге.

2. Метода `move(int)` (20 минута)

Циљ: Употреба методе `move(int)` за померање објекта унапред и уназад**.**

Концепти за дискусију: методе за померање објекта.

Активности: Наставник објашњава `move(int)` методу. Наставник мења позицију инстанце `Enemy` класе и позива методу прослеђујући јој различите позитивне вредности, на пример **1** и **3**. Наставник пита ученике шта мисле да ће се догодити уколико позову `move` методу и проследе јој негативну вредност, на пример **-1**.

Наставник задаје ученицима задатак да имплементирају методу `backward()` која треба да обезбеди да се објекат помери за **1** корак уназад. Да ли је могуће померити објекат за **2** корака или **X** корака уназад? Шта је потребно променити да би се то обезбедило?

3. Кључна реч `this` (5 минута)

Циљ: Схватање улоге кључне речи `this`**.**

Концепти за дискусију: кључна реч `this`.

Активности: Наставник објашњава кључну реч `this`. Помоћу кључне речи `this` добија се референца на актуелни објекат, путем које се може приступити атрибутима који су специфични за дати објекат или се могу позивати методе у контексту датог објекта.

4 .Задатак **3.3.** (10 минута)

Циљ: Савладавање начина имплементације методе за вертикално померање објекта.

Концепти за дискусију: позивање методе.

Активности: Наставник пита ученике како би померили објекат вертикално, тј. на горе или на доле. Наставник затим тражи од ученика да пронађу погодну методу за померање горе/доле тако што ће, након десног клика на дати објекат, одабрати опцију `Inherited from Actor`. Ученици би требало да препознају следеће методе: `turn`, `setRotation`, `setLocation`, `getLocation`, `getRotation`. Наставник затим задаје ученицима задатак да имплементирају методе `up()` и `down()`. Док ученици решавају задатак, наставник прати њихов рад, и када заврше са радом појашњава како се дате методе могу имплементирати. Методе `up()` и `down()` мењају вредност `y` координате, при чему је повећавају односно смањују, респективно. Наставник сада може увести концепт параметара методе, али без залажења у детаље.

5. Аутоматско довршавање кôда (5 минута)

Циљ: Упознавање са опцијом аутоматског довршавања кôда (енгл. *autocomplete*) у *Greenfoot* окружењу.

Концепти за дискусију: аутоматско довршавање кôда (**CTRL+SPACE**).

Активности: Наставник показује ученицима како могу пронаћи методе које се могу позвати над одређеним објектом. Наставник истиче да ће им ова опција бити посебно корисна ако су заборавили како се нека метода тачно зове или ако су тек почели да програмирају и желе да „питају" *Greenfoot* окружење за помоћ како би брже написали кôд.

6. Значај документовања програмског кôда (15 минута)

Циљ: Разумевање улоге коментара и документације**.**

Концепти за дискусију: коментари и документација, прозор са документацијом.

Активности: Наставник упознаје ученике са специјалним линијама кôда које нису део самог програма, тј. које се не извршавају (коментари, коментари за документовање).

Наставник треба да истакне разлику између обичних коментара и документације (као посебног типа коментара).

Наставник показује ученицима изворни кôд класе `Enemy`, а затим и генерисани *HTML* документ који представља документацију дате класе. Наставник овом приликом објашњава и да постоје одређена правила којих се треба придржавати када се пише документација за наше класу или методе.

Наставник задаје ученицима задатак да истраже како се пише документација за *Java* класе и методе.

7. Задатак **3.4.** (5 минута)

Циљ: Упознавање са начином документовања метода**.**

Концепти за дискусију: ознаке (енгл. *tags*) за документовање метода.

Активности: Наставник задаје ученицима задатак.

> **Задатак 3.4. (Документовање метода):** Додајте коментар за документовање методе `act()`.
>
> **[Commit***:* `68b1c82c7df2c7826f2d3f78373498569adab7e9`**]**

8. Задатак **3.5.** (5 минута)

Циљ: Упознавање са начином документовања класа**.**

Концепти за дискусију: ознаке (енгл. *tags*) за документовање класа.

Активности: Наставник задаје ученицима задатак да додају коментар за документовање методе `act()`.

> **Задатак 3.5. (Документовање класа):** Измените коментар за документовање `Enemy` класе. Додајте верзију класе и аутора, а затим уочите промене до којих је дошло у генерисаној *HTML* страници.
>
> **[Commit:** `1a7a9f83c5271a7c0dfa46ce3b1ee65682b0c5e5`**]**

9. Задатак **3.6.** (10 минута)

Циљ: Овладавање начином прегледања документације класе како би се разумело понашање њених метода.

Концепти за дискусију: проучавање документације.

Активности: Наставник задаје ученицима задатак.

> **Задатак 3.6. (Преглед документације):** Истражите прозор са документацијом.

Наставник тражи од ученика да проучи документацију класа `Actor` и `World`. Наставник наглашава значај прегледања документације како би се пронашле методе које би могле бити од користи.

10. Задатак **3.7.** (20 минута)

Циљ: Упознавање са контролама доступним у *Greenfoot* окружењу**.**

Концепти за дискусију: дугмад у *Greenfoot* окружењу.

Активности: Наставник наставља рад на последњој верзији пројекта. Наставник тражи од ученика да додају два објекта класе `Enemy` и да позову методу `act()` сваке од инстанци.

Наставник затим указује на дугме `Act`. Наставник треба да кликне на `Act` дугме које се налази у главном прозору. Наставник поставља питања, како би ученици сами закључили и објаснили шта се догодило.

Наставник такође тражи од ученика да кликну на дугме `Run` и да закључе шта се догодило.

Наставник тражи од ученика да кликну на дугме `Reset` и објасне шта се догодило.

> **Задатак 3.7. (Проучавање контрола *Greenfoot* окружења):** Испробајте дугмад у главном прозору *Greenfoot* окружења. Креирајте више инстанци класе `Enemy`. Кликните на дугме `Act` - шта се догађа? Кликните на дугме `Run` - шта се догађа? Након што први пут кликните на дугме `Run`, кликните на дугме `Pause` - шта се догађа? Какав ефекат има померање клизача `Speed` на позивање `act()` методе након клика на дугме `Run`? Шта се догађа када кликнете на дугме `Reset`?

Након тога, наставник објашњава шта је потребно урадити како би се, сваки пут када се кликне на `Reset` дугме, појавила два објекта класе `Enemy` на табли на позицијама `(0,3)` и `(3,3)`. Наставник треба да објасни ученицима да је, уколико желе да се објекти појаве на сцени сваки пут када се кликне на `Reset` дугме, потребно изменити конструктор класе `World` тако да се унутар њега креирају објекти и постављају на жељене позиције.

11. Дискусија: Алгоритам, особине, развој алгоритама, *Greenfoot* дугмад (5 минута)

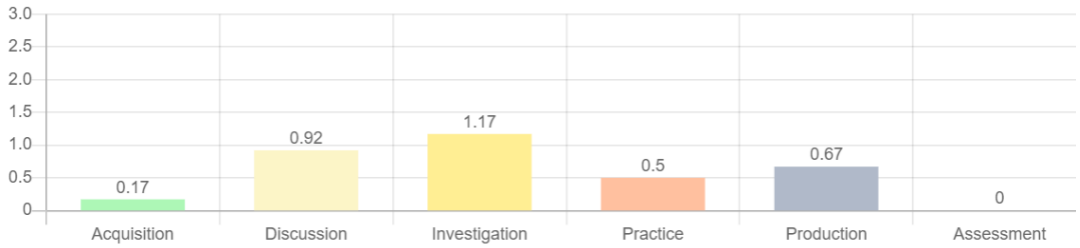
Циљ: Резимирање лекције**.**

Концепти за дискусију: методе за управљање кретањем, дугмад у *Greenfoot* окружењу, документација.

Активности: Наставник даје резиме лекције. Наставник затим може да истакне значај документовања кôда, као и да се осврне на конвенције за именовање метода и класа.

4. ГРАНАЊЕ

4.1. Структура наставних активности и задаци



Слика 4. Расподела ангажовања ученика према типу учења за тематску целину „Гранање“

Задатак 4.1. Праћење промена интерног стања објекта

Креирајте инстанцу класе `Enemy` и поставите је у центар сцене. Отворите прозор у којем се приказује интерно стање дате инстанце и поставите га тако да буде видљив док је апликација покренута. Затим покрените апликацију и пратите промене у вредностима атрибута `x`, `y` и `rotation` дате инстанце `Enemy` приликом позивања различитих метода. Како се ове вредности током померања (горе, доле, лево и десно) и окретања?

Задатак 4.2. Детектовање ивица света

Додајте кôд у тело методе `act()` како би се, када објекат класе `Enemy` стигне до ивице света, он окренуо за **180°**.

[Commit: `4927c3ff7eb39b51ba2738f2ab500fd6c32e3bb4`**]**

Задатак 4.3. Увођење нових класа

Уведите две нове класе, `Direction` и `Orb`, као изведене класе `Actor` класе. Припремите одговарајуће слике (максималне величине `50 x 50` пиксела) у неком графичком едитору. Затим придружите дате слике новим класама.

[Commit: `4ed6b37e6d481181d8b340639aa03391406b6c2e`**]**

Задатак 4.4. Детектовање колизије

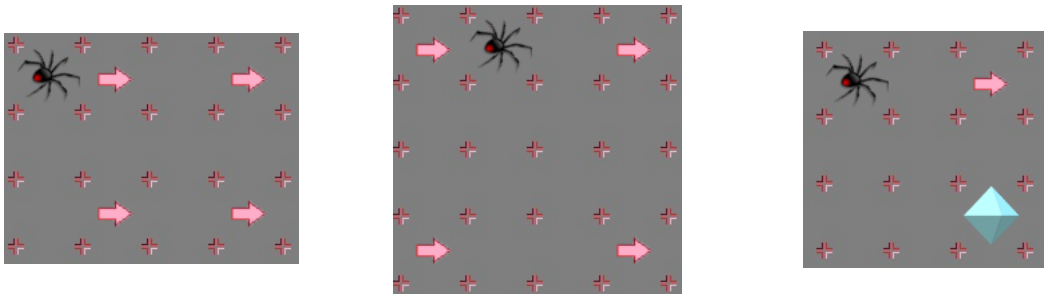
Додати кôд у тело `act()` методе `Enemy` класе, како би се обезбедило да се:

- непријатељ окрене за **90°**, у смеру кретања казаљке на сату, ако се затекне у истој ћелији у којој се налази инстанца класе `Direction`.
- непријатељ окрене за **90°**, у смеру супротном од кретања казаљке на сату, ако се затекне у истој ћелији у којој се налази инстанца класе `Orb`.

[Commit: `968e6f195e3def25e11bc41b664ba1715f7da11d`**]**

Задатак 4.5. Предвиђање начина кретања инстанце непријатеља (произвољна почетна поставка)

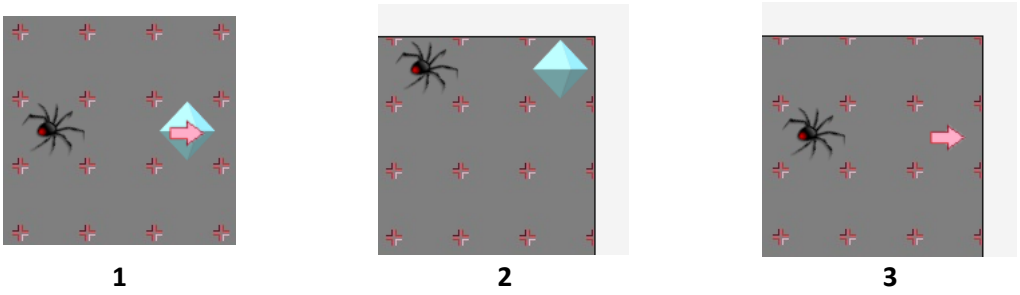
Припремите различите почетне поставке игре, инспирацију можете пронаћи на слици испод. Како ће се померати непријатељ? Покрените апликацију. Да ли је ваша претпоставка тачна? Ако није, зашто?



Слика 5. Примери почетних позиција инстанци

Задатак 4.6. Предвиђање начина кретања инстанце непријатеља (задата почетна поставка)

Размотрите почетну поставку на свакој од следећих слика. Како ће се померати непријатељ? Подесите апликацију (тј. поставите инстанце на одговарајуће почетне позиције) и покрените је. Да ли је ваша претпоставка тачна? Ако није, зашто?



Слика 6. Изазовније почетне поставке игре

Задатак 4.7. Имплементација потпуног и вишеструког гранања

Измените тело `act()` методе `Enemy` класе тако што ћете увести потпуно и угњеждено гранање. Потребно је дефинисати угњеждене услове. Детектовање да ли је непријатељ дошао до ивице света треба да буде примарни услов, затим треба проверити да ли додирује инстанцу класе `Direction` и, на крају, да ли додирује инстанцу класе `Orb`.

[Commit*:* `f017de8b49d4fc77f62afac4d842429560bcfb8b`**]**

Задатак 4.8. Предвиђање начина кретања инстанце непријатеља (промењено понашање)

Прођите поново кроз **Задатак 4.5.** и **Задатак 4.6.** Шта се променило?

4.2. Наставни сценарији

Припремљена су два наставна сценарија за тематску целину „Гранање“.

4.2.1. СЦЕНАРИО: **Истраживање непотпуног гранања кроз развој игара у *Greenfoot* окружењу**

Табела 7. Истраживање непотпуног гранања кроз развој игара у Greenfoot окружењу

Назив	Истраживање непотпуног гранања кроз развој игара у *Greenfoot* окружењу
Образовни циљеви и исходи учења	Наставни сценарио покрива непотпуно гранање (вишеструко гранање је намерно изостављено). Представљају се основе перцепције света (`World`) од стране протагонисте (`Actor`). Ученици ће умети да пишу кôд користећи услове. Након ове лекције, ученици ће знати како да употребе једноставне `if-else` наредбе и како да користе услове у кôду како бу контролисали понашање своје игре. Стећи ће основно знање о програмском језику *Java*, савладати неопходну синтаксу и научити како да анализирају и разумеју програмски кôд, што ће им помоћи да разумеју зашто се њихова игра понаша на одређени начин и како да разреше проблеме. Ученици ће бити способни да развију сопствене пројекте игара примењујући стечена знања о објектно-оријентисаном програмирању.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује и познавање концепата променљивих, функција, петљи и гранања. Потребно је и да ученици буду упознати и са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **50** минута. 1. Увод (5 минута) 2. Тумачење кôда (15 минута) 3. Непотпуно гранање (10 минута) 4. Задатак **4.1.** (10 минута) 5. Задатак **4.2.** (10 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Лекција почиње краћим уводом, након ког следи петнаестоминутни блок посвећен тумачењу кôда методе `turn(int)` класе `Actor`, чиме се успоставља основа за увођење концепта непотпуног гранања. Наставник усмерава дискусију са циљем да осигура да сви ученици истоветно разумеју улогу ове методе. Након тога, следи десетоминутни блок посвећен савладавању основа непотпуног гранања. Овај блок је кључан како би ученици усвојили суштинске принципе пре него што се упусте у писање кôда.

	Ученици затим учествују у десетоминутном истраживачком задатку (**Задатак 4.1.**), у оквиру којег анализирају интерно стање протагонисте, како би развили способност праћења и разумевања тока извршавања програма на основу изворног кôда. Наредни десетоминутни блок посвећен је практичном раду (**Задатак 4.2.**) и од ученика се очекује да у свој пројекат додају кôд који омогућава детектовање ивица света, додају понашање и подесе игру.
Вредновање	Активности у оквиру ове лекције ће омогућити наставницима да дају ученицима формативне повратне информације базиране на њиховом учешћу у спроведеним дискусијама, али и на праћењу њиховог рада у окружењу обрнуте учионице (енгл. *flipped classroom*), као и рада у тиму. Међусобно вредновање ће се вршити *online*, као део домаћег задатка. Кроз ову активност ученици ће се подсетити важних аспеката лекције и биће подстакнути да критички процењују рад других ученика. Разматрањем различитих примера, ученици ће лакше усвојити изложено градиво, што доприноси и повећању општег достигнућа постављених образовних циљева и исхода учења целе групе. Исходе учења и стечена знања ће ученици даље применити и током развоја тимског пројекта на којем раде.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

4.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Увод (5 минута)

Циљ: Утврђивање знања о претходно обрађеним, концептима. Представљање циљева лекције.

Концепти за дискусију: алгоритам, документовање програмског кôда.

Активности: Током увода у лекцију наставник се, заједно са ученицима, осврће на претходно обрађене концепте. Наставник затим представља нову област коју ће са ученицима обрађивати током лекције. Како би илустровао предмет лекције, наставник представља пример алгоритма са једноставним гранањем. Пример би могао бити алгоритам за прелазак улице на пешачком прелазу без семафора. Пешак не прелази одмах улицу, већ најпре проверава да ли има возила која долазе са леве или десне стране. Ако нема возила, пешак прелази улицу.

2. Тумачење кôда (15 минута)

Циљ: Анализа појединих метода класе `Actor`.

Концепти за дискусију: интерно стање објекта.

Активности: Наставник преузима најновију верзију пројекта:

- са *Moodle* платформе или
- из *git* репозиторијума

Наставник креира и поставља објекат класе `Enemy` на неко место на сцени. Наставник затим појашњава поједине методе класе `Actor`:

- `move(int)`
- `turn(int)`
- `setRotation(int)`

Током појашњавања самих метода, наставник указује на то како се мењају одређена својства објекта (на пример, позиција објекта на сцени, тј. вредности `x` и `y` координата). Наставник затим заједно са ученицима разматра како би се могла допунити метода `act()` да би се, сваки пут када се она позове, објекат класе `Enemy` померио за два корака унапред.

3. Непотпуно гранање (10 минута)

Циљ: Усвајање концепта непотпуног гранања**.**

Концепти за дискусију: гранање, непотпуно гранање.

Активности: Наставник наставља рад на пројекту. Поставља објекат класе `Enemy` на сцену. Наставник објашњава ученицима како могу да провере да ли се дати објекат налази у горњој половини сцене, а онда да прикажу поруку: "**`Pronadjen`**" (енгл. "**Found**").

Наставник ученицима показује методу `showText` која служи за приказивање текста на екрану.

4. Задатак **4.1.** (10 минута)

Циљ: Развијање способности праћења и разумевања тока извршавања програма

Концепти за дискусију: интерно стање објекта.

Активности: Наставник показује ученицима како се могу пратити промене интерног стање објекта:

> **Задатак 4.1. (Праћење промена интерног стања објекта)**: Креирајте инстанцу класе `Enemy` и поставите је у центар сцене. Отворите прозор у којем се приказује интерно стање дате инстанце и поставите га тако да буде видљив док је апликација покренута.

Затим покрените апликацију и пратите промене у вредностима атрибута `x`, `y` и `rotation` дате инстанце `Enemy` приликом позивања различитих метода. Како се ове вредности током померања (горе, доле, лево и десно) и окретања?

Ученици одговарају на постављена питања.

5. Задатак **4.2.** (10 минута)

Циљ: Осмишљавање и имплементација поступка детектовања досезања ивица света**.**

Концепти за дискусију: детектовање ивица света (`World`).

Активности: Наставник разговара са ученицима о томе како могу утврдити да ли се неки објекат налази, или не налази, на ивици (енгл. *edge*) света (`World`). На пример, ако су познате тачне димензије света, на основу позиције `(x,y)` може се утврдити да ли се објекат налази или не налази на његовој ивици.

Наставник поставља инстанцу класе `Enemy` било где на сцени (али не на њеној ивицу) и позива методу `isAtEdge()`. Наставник са ученицима дискутује о томе шта се догодило. Наставник затим премешта објекат на ивицу и указује на резултат извршавања методе `isAtEdge()`, која би сада требало да врати вредност `true`.

Наставник појашњава методу `isAtEdge()`.

Наставник задаје ученицима задатак:

> **Задатак 4.2. (Детектовање ивица света)**: Додајте кôд у тело методе `act()` како би се, када објекат класе `Enemy` стигне до ивице света, он окренуо за **180°** (позивањем методе `setRotation(int)`).

Наставник разговара са ученицима о томе како објекат који стигне до ивице света, може да настави своје кретање:

- уназад (без окретања)
- уназад (са окретањем).

Наставник, заједно са ученицима, допуњује програмски кôд `act()` методе.

Након тога, потребно је покренути `act()` методу и прокоментарисати са ученицима шта се догађа са објектом класе `Enemy` када он стигне до ивице света.

[Commit*:* `4927c3ff7eb39b51ba2738f2ab500fd6c32e3bb4`**]**

4.2.2. СЦЕНАРИО: **Истраживање потпуног гранања кроз развој игара у *Greenfoot* окружењу**

Табела 8. Истраживање потпуног гранања кроз развој игара у Greenfoot окружењу

Назив	Истраживање потпуног гранања кроз развој игара у *Greenfoot* окружењу
Образовни циљеви и исходи учења	Наставни сценарио покрива непотпуно и потпуно гранање, као и вишеструко гранање. Уводи нове начине перцепције света од стране протагониста. Ученици ће умети да пишу кôд користећи сложеније услове. Након ове лекције, ученици ће знати како да употребе сложене `if-else` наредбе и како да користе специфичније и софистицираније услове како бу контролисали понашање своје игре. Стећи ће додатно знање о програмском језику *Java*, савладати неопходну синтаксу и научити како да анализирају и разумеју сложенији програмски кôд, што ће им помоћи да разумеју зашто се њихова игра понаша на одређени начин и како да разреше проблеме. Ученици ће бити способни да развију сопствене пројекте игара примењујући нова сазнања.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује и познавање концепата променљивих, функција, петљи и гранања. Потребно је и да ученици буду упознати и са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **170** минута. 1. Задатак **4.3.** (30 минута) 2. Детектовање колизије (30 минута) 3. Задатак **4.4.** (10 минута) 4. Задатак **4.5.** (15 минута) 5. Задатак **4.6.** (15 минута) 6. Тумачење кôда: Потпуно гранање (15 минута) 7. Задатак **4.7.** (20 минута) 8. Задатак **4.8.** (30 минута) 9. Обнављање теоријских концепата (5 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Лекција почиње 30-минутном практичном вежбом (**Задатак 4.3.**) током које ученици уводе нове класе: `Direction` и `Orb`. Овај задатак помаже ученицима да боље разумеју објектно-оријентисану природу програмског језика *Java* и увиде значај правилног структурирања програма. У наредном блоку од 20 минута разматра се начин детектовања колизије, што треба да помогне ученицима да боље разумеју начин интеракције објеката у окружењу игре. Током наредних 10 минута ученици имају задатак да додају детекцију колизије у свој пројекат, и практично примене стечено знање о гранању (**Задатак 4.4.**).

	Следе истраживачки задаци од по 15 минута, у оквиру којих се од ученика тражи да предвиде начин кретања објеката класе **Enemy** са произвољном (**Задатак 4.5.**) и изазовнијом почетном поставке игре (**Задатак 4.5.**), кроз које развијају способност решавања проблема и аналитичке вештине. Наредна петнаестоминутна дискусија је усмерена на комплетно гранање, како би се осигурало да ученици увиђају разлику између непотпуног и потпуног гранања кôда. Наредни блок од 20 минута је посвећен практичном раду и од ученика се очекује да имплементирају сложено гранања са детекцијом колизије, како би кроз практичну примену сложенијих концепата утврдили стечено знање (**Задатак 4.7.**). Лекција се приводи крају изазовним истраживачким задатком (**Задатак 4.8.**) од 30 минута у оквиру којег се од ученика тражи да поново предвиде начин кретања непријатеља, овог пута са додатним искуством које су стекли решавањем претходних задатака, како бу применили целокупно стечено знање. Лекција се завршава петоминутним теоријским освртом, односно рекапитулацијом обрађених концепата.
Вредновање	Активности у оквиру ове лекције ће омогућити наставницима да дају ученицима формативне повратне информације базиране на њиховом учешћу у спроведеним дискусијама, али и на праћењу њиховог рада у окружењу обрнуте учионице (енгл. *flipped classroom*), као и рада у тиму. Међусобно вредновање ће се вршити *online*, као део домаћег задатка. Кроз ову активност ученици ће се подсетити важних аспеката лекције. Разматрањем различитих примера, ученици ће лакше усвојити изложено градиво, што доприноси и повећању општег достигнућа постављених образовних циљева и исхода учења целе групе. Исходе учења и стечена знања ће ученици даље применити и током развоја тимског пројекта на којем раде.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

4.2.2.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Задатак **4.3.** (30 минута)

Циљ: Увежбавање начина додавања нових класа у пројекат.

Концепти за дискусију: класа.

Активности: Наставник задаје ученицима задатак:

> **Задатак 4.3. (Увођење нових класа)**: Уведите две нове класе, `Direction` и `Orb`, као изведене класе `Actor` класе. Припремите одговарајуће слике (максималне величине `50x50` пиксела) у неком графичком едитору. Затим придружите дате слике новим класама.

Док ученици решавају задатак наставник прати њихов рад, а затим бира једног ученика који треба да представи и објасни своје решење.

[Commit*:* `4ed6b37e6d481181d8b340639aa03391406b6c2e`**]**

2. Детектовање колизије (30 минута)

Циљ: Разматрање начина детектовања колизије**.**

Концепти за дискусију: детектовање колизије.

Активности: Наставник поставља инстанцу класе `Enemy` на одређену позицију, а затим поставља инстанцу класе `Direction` у исти ред сцене тј. света. Наставник затим у `act()` методу додаје кôд путем којег се објекат помера један корак унапред.

Наставник објашњава ученицима како да утврде да ли се два, или више, објеката („протагониста“) налазе на истој позицији (тј. у истој ћелији) света. Наставник разјашњава методу `isTouching()`.

Наставник, заједно са ученицима, мења `act()` методу `Enemy` класе како би се обезбедило да се непријатељ окрене за **90°**, у смеру кретања казаљке на сату, ако се затекне у истој ћелији у којој се налази инстанца класе `Direction`.

Наставник, заједно са ученицима, прати шта се догађа са вредношћу атрибута `rotation`.

3. Задатак **4.4.** (10 минута)

Циљ: Утврђивање начина детектовања колизије.

Концепти за дискусију: /.

Активности: Наставник задаје ученицима задатак:

> **Задатак 4.4. (Детектовање колизије)**: Додати кôд у тело `act()` методе `Enemy` класе, како би се обезбедило да се непријатељ окрене за **90°**, у смеру супротном од кретања казаљке на сату, ако се затекне у истој ћелији у којој се налази инстанца класе `Orb`.

Док ученици решавају задатак наставник прати њихов рад, а затим бира једног ученика који треба да представи и објасни своје решење.

[Commit*:* `968e6f195e3def25e11bc41b664ba1715f7da11d`**]**

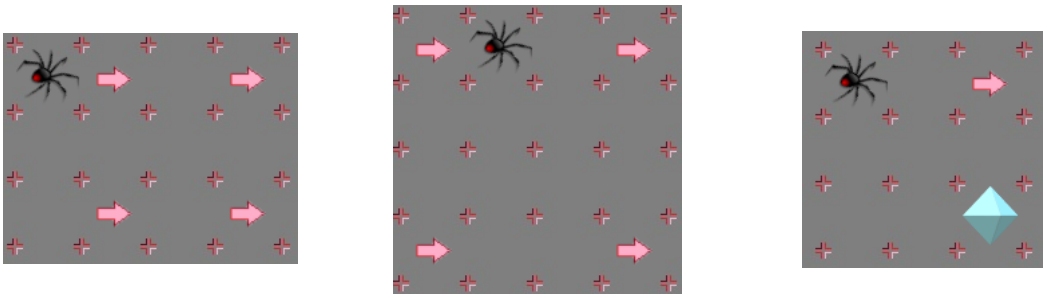
4. Задатак **4.5.** (15 минута)

Циљ: Разматрање начина кретања инстанце непријатеља.

Концепти за дискусију: начин кретања инстанце непријатеља.

Активности: Наставник задаје ученицима задатак:

Задатак 4.5. (Предвиђање начина кретања инстанце непријатеља): Припремите различите почетне поставке игре, инспирацију можете пронаћи на слици испод. Како ће се померати непријатељ? Покрените апликацију. Да ли је ваша претпоставка тачна? Ако није, зашто?



1 **2** **3**

Слика 7. Примери почетних позиција инстанци

Док ученици решавају задатак наставник прати њихов рад, а затим бира једног ученика који треба да представи и објасни своје решење.

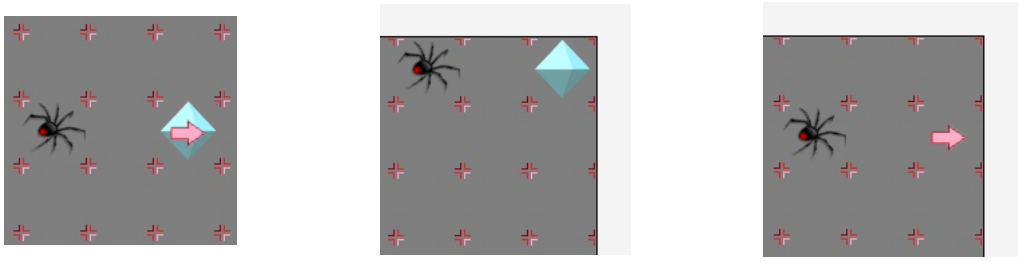
5. Задатак **4.6.** (15 минута)

Циљ: Утврђивање знања о начину кретања инстанце непријатеља.

Концепти за дискусију: начин кретања инстанце непријатеља.

Активности: Наставник задаје ученицима задатак:

Задатак 4.6. (Предвиђање начина кретања инстанце непријатеља): Размотрите почетну поставку на свакој од следећих слика. Како ће се померати непријатељ? Подесите апликацију (тј. поставите инстанце на одговарајуће почетне позиције) и покрените је. Да ли је ваша претпоставка тачна? Ако није, зашто?



1 **2** **3**

Слика 8. Изазовније почетне поставке игре

Док ученици решавају задатак наставник прати њихов рад, а затим бира једног ученика који треба да представи и објасни своје решење.

6. Тумачење кôда: Потпуно гранање (15 минута)

Циљ: Упознавање са потпуним и вишеструким гранањем**.**

Концепти за дискусију: `if-else` наредба, угњеждена `if-else` наредба, `switch` наредба.

Активности: Наставник тражи од ученика да опишу, на папиру, како пешак прелази улицу. Наставник прати рад ученика. Након одређеног времена,, наставник напомиње да је потребно обратити пажњу на то да ли на месту преласка улице постоји семафор. Уколико је неки ученик у међувремену већ поставио ово, или неко слично, питање, наставник га похваљује и наглашава да је веома важно да се, пре него што се приступи имплементацији, увек прво изанализира проблем и идентификују све могуће ситуације. Кроз ову активност би ученици требало да савладају потпуно и угњеждено гранање.

Наставник ће затим замолити ученике да поставе објекте класе `Orb` и класе `Direction` на различите ивице света. Ученици описују „погрешно" понашање непријатеља. Наставник треба да укаже на то да се може десити да истовремено буду испуњена два различита услова: непријатељ додирује инстанцу класе `Orb`/`Direction`, али додирује и ивицу света. Цртањем овакве ситуацију на табли или на папиру, ученици треба да увиде да је потребно увести потпуно и угњеждено гранање, након чега треба да прилагоде понашања непријатеља (тј. да промене методу `act()` класе `Enemy`).

7. Задатак **4.7.** (20 минута)

Циљ: Савладавање начина имплементације потпуног и вишеструког гранања**.**

Концепти за дискусију: угњеждена `if-else` наредба, `switch` наредба.

Активности: Наставник задаје ученицима задатак:

> **Задатак 4.7. (Имплементација потпуног и вишеструког гранања)**: Измените тело `act()` методе `Enemy` класе тако што ћете увести потпуно и угњеждено гранање. Потребно је дефинисати угњеждене услове. Детектовање да ли је непријатељ дошао до ивице света треба да буде примарни услов, затим треба проверити да ли додирује инстанцу класе `Direction` и, на крају, да ли додирује инстанцу класе `Orb`.

Док ученици решавају задатак наставник прати њихов рад, а затим бира једног ученика који треба да представи и објасни своје решење.

[Commit*:* `f017de8b49d4fc77f62afac4d842429560bcfb8b`**]**

8. Задатак **4.8.** (30 минута)

Циљ: Предвиђање кретања непријатеља са произвољном почетном поставком**.**

Концепти за дискусију: понашање објеката.

Активности: Наставник поставља објекте на сцену насумично, а ученици треба да објасне њихово кретање и понашање (појединачно или у паровима).

Наставник задаје ученицима задатак:

> **Задатак 4.8. (Предвиђање начина кретања инстанце непријатеља)**: Прођите поново кроз **Задатак 4.5.** и **Задатак 4.6.** Шта се променило?

9. Обнављање теоријских концепата (5 минута)

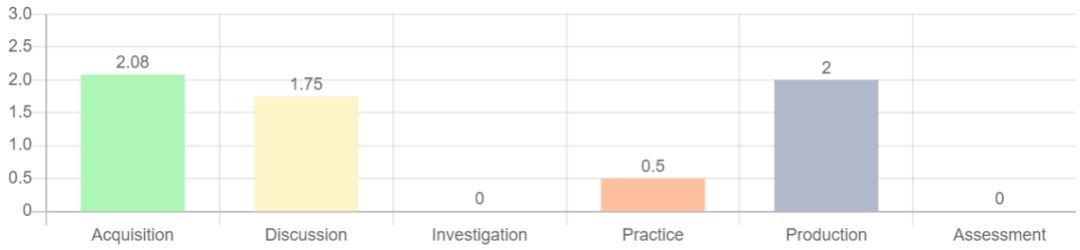
Циљ: Резимирање целокупне области**.**

Концепти за дискусију: гранање.

Активности: Наставник даје резиме лекције и осврће се заједно са ученицима, на све претходно обрађене концепте.

5. ПРОМЕНЉИВЕ И ИЗРАЗИ

5.1. Структура наставних активности и задаци



Слика 9. Расподела ангажовања ученика према типу учења за тематску целину „Променљиве и изрази“

Задатак 5.1. Ротирање објеката у правцу који најпре треба одредити

Промените тело `act()` методе класе `Enemy` тако да обезбедите да се њена инстанца ротира у правцу који је дефинисан у инстанци класе `Direction` коју додирује (оне ће имати исту ротацију). Позовите методу `getOneIntersectingObject(_cls_)` и сачувајте резултујућу инстанцу у погодној локалној променљивој (нпр. `Direction direction`) – неопходно је пре тога извршити конверзију, јер метода заправо враћа објекат типа `Actor`; дакле, методу треба позвати на следећи начин: `(Direction)getOneIntersectingObject(_cls_)`. Ако је добијен одговарајући резултат, позовите методу `getDirection()` како бисте добили неопходну информацију о ротацији (сачувајте је ако желите), а затим подесите ротацију инстанце непријатеља (`this`) помоћу методу `setDirection(int)`. Тестирајте своје решење.

[Commit*:* `97dddc4beba40ac785c7413bb245ba849cd956d2`**]**

Задатак 5.2. Промена назива класе `MyWorld` у `Arena`

Преименујте претходно дефинисану класу `MyWorld`. Нови назив класе треба да буде `Arena`. Поред тога, неопходно је да преименујете и конструктор класе `MyWorld()` у `Arena()`.

[Commit*:* `aaf73c9bfd9f76a2a1e504f5e78d2976f1cada12`**]**

Задатак 5.3. Креирање сопствену почетне поставке протагониста у Арени

Креирајте сопствену почетну поставку протагониста у Арени. Протагонисти треба да буду постављени на одговарајуће позиције унутар конструктора класе `Arena`. Потребно је креирати једну инстанцу класе `Enemy`, једну инстанцу класе `Orb` и барем једну инстанцу класе `Direction`. Након што декларишете и иницијализујете променљиве жељеног типа, потребно је да подесите својства креираних објекта позивањем одговарајућих метода. Коначно, креиране објекте треба да поставите на жељене позиције у арени позивањем методе `addObject(Actor,int,int)`. Тестирајте своје решење.

[Commit*:* `8b105ea2eaf697f08c321efe687ddd31e2d0a041`**]**

Задатак 5.4. Идентификовање проблема везаног за кретања непријатеља и предлог решења

Да ли уочавате неки проблем везан за кретања непријатеља? Шта је узрок? Како би се могао решити?

Задатак 5.5. Додавање атрибута `moveDelay` класи `Enemy`

Потребно је додати атрибут `moveDelay` (типа `int`) у класу `Enemy`. Потребно је и имплементирати параметризовани конструктор који треба да обезбеди да се дати атрибут иницијализује са вредношћу параметра. Коначно, потребно је ажурирати и кôд у класи `Arena` у складу са уведеним изменама.

[Commit*:* `6092489ce57541e77ae4e2ee886b20853df9f8a4`**]**

Задатак 5.6. (Имплементација кретања непријатеља уз задршку): Потребно је изменити методу `act()` класе `Enemy`, тако да се обезбеди да ће се непријатељ померити тек након што се метода `act()` изврши одређени број (`moveDelay`) пута. Такође је потребно додати нови атрибут `nextMoveCounter` типа `int` и иницијализовати га са **0**. Затим треба изменити методу `act()` тако да позива `this.move(1)` само ако атрибут `nextMoveCounter` има вредност **0**. Након померања, атрибут `nextMoveCounter` добија вредност атрибута `moveDelay`. Када непријатељ не може да се помери (јер `nextMoveCounter` још није достигао вредност **0**), потребно је смањити вредност атрибута `nextMoveCounter` за **1**.

[Commit*:* `bf26e6ed23911ccb712fae3e243cdedff3a89a7f`**]**

Задатак 5.7. Имплементација параметризованог конструктора класе `Direction`

Дефинишите параметризовани конструктор у класи `Direction`. Конструктор треба да прима један параметар: `rotation` типа `int`. У телу конструктора је потребно ротирати креирану инстанцу у складу са вредношћу параметра. Потребно је ажурирати и кôд у класи `Arena` у складу са уведеним изменама.

[Commit*:* `3c4b9ef57ab17bac2a0abc7fc5e76ea4b6e27e4b`**]**

Задатак 5.8. Преклапање конструктора у класи `Direction`

Дефинишите додатни непараметризовани конструктор у класи `Direction`. У телу овог конструктора је потребно позвати претходно дефинисани параметризовани конструктор са аргументом постављеним на **0** (ово одговара ротацији од **0°**). Потребно је ажурирати и кôд у класи `Arena` позивајући непараметризовани конструктор класе `Direction`, где је то могуће.

[Commit*:* `1e67e67523c66acea4e93363c9a3173302f424c8`**]**

5.2. Наставни сценарији

Припремљено је пет наставних сценарија за тематску целину „Променљиве и изрази“.

5.2.1. СЦЕНАРИО: **Увод у променљиве и типове података у *Greenfoot* окружењу**

Табела 9. Увод у променљиве и типове података у Greenfoot окружењу

Назив	Увод у променљиве и типове података у *Greenfoot* окружењу
Образовни циљеви и исходи учења	После ове лекције, ученици ће умети да правилно користе променљиве и типове података. Разумевање ових концепата се продубљује кроз развој игре у *Greenfoot* окружењу, чиме се подстиче креативност, тимски дух и страственији приступ програмирању.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Развијају се основне вештине програмирања, са нагласком на стицање знања о променљивима и типовима података. Потребно је да ученици буду упознати са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **45**-минута. 1. Увод (10 минута) 2. Идентификација променљивих (5 минута) 3. Типови података (15 минута) 4. Декларација променљивих (10 минута) 5. Иницијализација променљивих (5 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Кроз овај наставни сценарио ученици ће се упознати са принципима програмирања који се односе на променљиве и типове података, а из перспективе развоја игара у *Greenfoot* окружењу. Лекција почиње десетоминутним уводом од стране наставника, који даје контекст, надовезујући се на претходне лекције, који ће бити основа за увођење и дефинисање концепта променљиве. Након тога следи петоминутни блок, током којег ученици, заједно са наставником, промишљају и идентификују променљиве за своју игру. Будући да свака променљива мора бити неког типа, током наредних 15 минута се обрађују различити типови података. Кључне активности обухватају десетоминутни блок, који води наставник, а током којег ученици декларишу променљиве за своју игру и уче о значају имена и типа променљиве; након чега следи петоминутни блок посвећен иницијализацији променљивих, тј. додељивању вредности променљивима. У том контексту, може се променити понашање објеката у игри (нпр. ротација објекта, кретање објекта).

	Ученици ће наставити рад на игри која је изложена и започета током претходних лекција. Сходно томе, до краја ове лекције у игру ће бити уведени нови концепти, који се односе на променљиве и типове података.
Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

5.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Увод (10 минута)

   У уводном делу се успоставља контекст у односу на претходне лекције. Наставник уводи појам **променљиве**.

2. Идентификација променљивих (5 минута)

   Циљ: Идентификовање променљивих кроз дискусију, са нагласком на улогу променљивих у програмирању.

   Концепти за дискусију: променљиве и вредности.

   Активности:
   - Наставник уводи појам променљиве.
   - Од ученика се може затражити да промисле и идентификују променљиве за своју игру.
   - Наставник и ученици могу дискутовати о променљивима.
   - Тип променљиве се, током ове активности, може занемарити (или начелно споменути).

3. Типови података (15 минута)

   Циљ: Разумевање концепта типа података и препознавање примера његове примене у свакодневном животу, а затим фокусирање на типове променљивих који су потребни за даљи развој игре.

   Концепти за дискусију: променљиве, типови података, примери типова података који се могу пронаћи у свакодневном животу, типови променљивих потребни за игру.

   Активности:
   - Наставник уводи појам **типа података**.
   - Можете поразговарати са ученицима о примерима из свакодневног живота (нпр. цели бројеви се могу довести у везу са бројем присутних ученика, реални бројеви се могу довести у везу са ценом неког производа, текстуални тип се може довести у везу са именом ученика итд.).
   - Размотрити типове података у контексту *Greenfoot* окружења и програмског језика *Java*.
   - Детаљно продискутовати типове променљивих који су потребни за даљи развој игре.

4. Декларација променљивих (10 минута)

   Циљ: Примена знања о типовима података и променљивима, како би се декларисале променљиве неопходне за развој игре.

   Концепти за дискусију: променљиве, типови података, типови променљивих потребни за игру.

   Активности:
   - Разрадити типове података у контексту *Greenfoot* окружења и програмског језика *Java*.
   - Наставник треба да појасни разлику између декларације и иницијализације променљиве:
     - Када се декларише нека променљива, специфицира се да је променљива одређеног типа података, док сама вредност може, али не мора, бити обезбеђена.
     - Може се појаснити аналогијом (нпр. предвиђено је поље за унос броја телефона, али број телефона још увек није унет).
   - Декларисати променљиве које су потребне за даљи развој игре.
   - Могу се размотрити и додатни примери. На пример, ако је реч о методи `act()`, може се декларисати променљива за текст који треба да буде приказан.

5. Иницијализација променљивих (5 минута)

Циљ: Употреба типова података и променљивих и иницијализација променљивих неопходних за развој игре.

Концепти за дискусију: променљиве, типови података, вредности променљивих.

Активности:

- Представити могуће вредности и распоне (енгл. *range*) вредности, претходно уведених типова података.
- Обрадити вредности и распоне типова података у контексту *Greenfoot* окружења и програмског језика *Java*.
- Наставник треба да понови разлику између декларације и иницијализације променљиве:
  - Када се иницијализује нека променљива, задаје јој се почетна вредност која се у наставку програма може променити. Променљива се може иницијализовати и приликом њене декларације (тј. истовремено са декларацијом).
- Иницијализовати променљиве које су потребне за даљи развој игре..
- Могу се размотрити и додатни примери. На пример, ако је реч о методи `act()`, може се иницијализовати променљива за текст који треба да буде приказан.

5.2.2. СЦЕНАРИО: **Увод у операторе и изразе у *Greenfoot* окружењу**

Табела 10. Увод у операторе и изразе у Greenfoot окружењу

Назив	Увод у операторе и изразе у *Greenfoot* окружењу
Образовни циљеви и исходи учења	После ове лекције, ученици ће умети да правилно примене операторе у изразима програмског језика. Уводе се различите врсте оператора (тј. аритметички оператори, логички оператори, релациони оператори) и одговарајући изрази. Поред тога, уводе се изрази над објектима и концепт референтних променљивих. Наведени концепти се разматрају у контексту развоја игре развој игре у *Greenfoot* окружењу, чиме се подстиче креативност, тимски дух и страственији приступ програмирању.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује и познавање концепата петљи и гранања. Потребно је да ученици буду упознати са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **95**-минута. 1. Оператори (15 минута) 2. Аритметички оператори и изрази (10 минута) 3. Логички оператори (15 минута) 4. Релациони оператори (10 минута) 5. Логички изрази (10 минута) 6. Изрази над објектима (5 минута) 7. Референтне променљиве (15 минута) 8. Задатак **5.1.** (15 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Кроз овај наставни сценарио ученици ће се упознати са принципима програмирања који се односе на операторе и изразе, а из перспективе развоја игара у *Greenfoot* окружењу. Лекција почиње петнаестоминутним уводом од стране наставника, који даје контекст, надовезујући се на претходне лекције, и поставља основу за увођење концепта оператора. Након тога следи десетоминутни блок, током којег се разматрају аритметички оператори и изрази. Различити оператори и изрази у *Greenfoot* окружењу се илуструју и анализирају кроз практичне примере. Следећи блок од 15 минута се односи на логичке операторе који се користе за манипулисање логичким вредностима. Затим се посвећује 10 минута релационим операторима који се користе за поређење вредности. Надовезујући се на претходно обрађене концепте, у оквиру следећег десетоминутног блока се разматрају логички изразе у контексту *Greenfoot* окружења.

	Током следећих 5 минута пажња се усмерава на изразе над објектима, док је се у оквиру наредног блока од 15 минута разматрају референтне променљиве. Последњих 15 минута је посвећено практичном раду, па ученици решавају **Задатак 5.1.**, при чему их наставник усмерава, а по завршетку им пружа повратне информације. У оквиру ове активности предвиђено је и међусобно вредновање. Ученици ће наставити рад на игри која је започета током претходних лекција. Сходно томе, до краја ове лекције у игру ће бити уведени нови концепти, који се односе на операторе и изразе.
Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе. Статус самог пројекта отвара простор за дефинисање домаћих задатака. У овом контексту, могу се увести нове класе, изрази и вредности, како би се обезбедило додатно сложеније понашање (нпр. телепортовање, тунели итд.). Наставник може да поразговара са ученицима о различитим могућностима, а затим им може задати, као домаћи задатак, да имплементирају одређено понашање.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

5.2.2.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Оператори (15 минута)

Циљ: Разумевање концепта оператора, повезивање са примерима из свакодневног живота и упознавање са различитим врстама оператора**.**

Концепти за дискусију: променљиве, типови података, оператори, примери примене оператора у свакодневном животу.

Активности: Наставник уводи појам **оператора**. Наставник треба да приближи овај концепт ученицима коришћењем примера из свакодневног живота (нпр. куповина производа на пијаци). Након тога, наставник представља различите врсте оператора.

2. Аритметички оператори и изрази (10 минута)

Циљ: Разумевање концепта аритметичких оператора, повезивање са примерима из свакодневног живота и упознавање са различитим аритметичким операторима**.**

Концепти за дискусију: променљиве, типови података, оператори, примери примене аритметичких оператора у свакодневном животу.

Активности: Наставник може да укаже на операторе који су ученицима већ познати из других предмета (нпр. аритметички оператори у математици). Дати оператори се затим разматрају у контексту *Greenfoot* окружења и програмског језика *Java*. Наставник објашњава различите релевантне појмове: оператор, операнд и приоритет оператора. Могу се размотрити и додатни примери. На пример, могу се дефинисати локалне променљиве, како би се омогућило памћење позиције објекта (**x**,**y**), а затим њено мењање повећавањем вредности датих променљивих.

3. Логички оператори (15 минута)

Циљ: Разумевање концепта логичких оператора, повезивање са примерима из свакодневног живота и упознавање са различитим логичким операторима**.**

Концепти за дискусију: променљиве, типови података, оператори, примери примене логичких оператора у свакодневном животу.

Активности: Наставник може да укаже на операторе који су ученицима већ познати из других предмета (нпр. логички оператори у математици). Дати оператори се затим разматрају у контексту *Greenfoot* окружења и програмског језика *Java*. Наставник појашњава операторе, операнде и приоритет оператора. Додатни пример може подразумевати дефинисање локалних променљивих да би се проверило да ли је **x**-координата позиције објекта једнака њеној **y**-координати, уз примену логичког оператора како би се утврдило да ли се објекат налази на дијагонали.

4. Релациони оператори (10 минута)

Циљ: Разумевање концепта релационих оператора, повезивање са примерима из свакодневног живота и упознавање са различитим релационим операторима**.**

Концепти за дискусију: променљиве, типови података, оператори, примери примене релационих оператора у свакодневном животу.

Активности: Наставник може да укаже на операторе који су ученицима већ познати из других предмета (нпр. релациони оператори у математици). Дати оператори се затим разматрају у контексту *Greenfoot* окружења и програмског језика *Java*. Наставник појашњава операторе, операнде и приоритет оператора. Додатни пример може подразумевати дефинисање локалних променљивих да би се проверило да ли је **y**-координата позиције неког објекта мања од **y**-координате неког другог објекта, уз примену релационог оператора како би се утврдио однос између позиција та два објекта.

5. Логички изрази (10 минута)

Циљ: Разумевање концепта логичких израза, повезивање са примерима из свакодневног живота и упознавање са различитим логичким изразима**.**

Концепти за дискусију: променљиве, типови података, оператори, примери примене логичких израза у свакодневном животу.

Активности: Наставник може објаснити логичке изразе у контексту претходно обрађених оператора. Дати изрази се затим разматрају у контексту *Greenfoot* окружења и програмског језика *Java*. Наставник затим разматра операторе, операнде и приоритете оператора у контексту логичких израза. Могу се размотрити и додатни примери. На пример, логички изрази се могу користити да би се потврдило да ли се објекат налази унутар света (задате димензије).

6. Изрази над објектима (5 минута)

Циљ: Разумевање концепта израза над објектима, повезивање са примерима из праксе и упознавање са различитим изразима над објектима**.**

Концепти за дискусију: променљиве, типови података, оператори, примери примене израза над објектима у пракси.

Активности: Наставник може објаснити изразе над објектима у контексту објектно-оријентисаног програмирања. Дати изрази се затим разматрају у контексту *Greenfoot* окружења и програмског језика *Java*. Наставник затим разматра операторе, операнде, приоритете оператора и конверзију (енгл. *casting*) у контексту израза над објектима. Могу се размотрити и додатни примери. На пример, могу се упоредити референце два објекта како би се проверило да ли се поклапају.

7. Референтне променљиве (15 минута)

Циљ: Разумевање концепта референтних променљивих, повезивање са примерима из праксе и примена у развој игре**.**

Концепти за дискусију: променљиве, типови података, оператори, примери примене референтних променљивих у пракси.

Активности: Наставник може објаснити референтне променљиве у контексту објектно-оријентисаног пројектовања. Дати променљиве се затим разматрају у контексту *Greenfoot* окружења и програмског језика *Java*. Наставник треба да појасни и концепт `null` вредности.

8. Задатак 5.1. (15 минута)

Циљ: Продубљивање разумевање концепта променљивих, типова података, оператора и израза, и њихова примене у развој игре**.**

Концепти за дискусију: променљиве, типови података, оператори, изрази.

Активности: Наставник треба да објасни како се понаша `act()` метода класе `Enemy` када се користе локалне променљиве у кôду, као на пример променљива `rotation`. Наставник треба да разјасни разлику између коришћења `this.rotation` и коришћења `rotation`. Наставник треба да опише понашање методе `getOneIntersectingObject(_cls_)`. Повратна вредност ове методе се може сачувати у погодној локалној променљивој (потребно је извршити одговарајућу конверзију). Уколико такав објекат не постоји, метода враћа `null` вредност. На основу резултата логичке евалуације повратне вредности методе, врши се одговарајућа радња (тј. ротирање или окретање).

Наставник излаже задатак:

> **Задатак 5.1. (Ротирање објеката у правцу који најпре треба одредити)**: Промените тело `act()` методе класе `Enemy` тако да обезбедите да се њена инстанца ротира у правцу који је дефинисан у инстанци класе `Direction` коју додирује (оне ће имати исту ротацију). Позовите методу `getOneIntersectingObject(_cls_)` и сачувајте резултујућу инстанцу у погодној локалној променљивој (нпр. `Direction direction`) – неопходно је пре тога извршити конверзију, јер метода заправо враћа објекат типа `Actor`; дакле, методу треба позвати на следећи начин: `(Direction)getOneIntersectingObject(_cls_)`. Ако је добијен одговарајући резултат, позовите методу `getDirection()` како бисте добили неопходну информацију о ротацији (сачувајте је ако желите), а затим подесите ротацију инстанце непријатеља (`this`) помоћу методу `setDirection(int)`. Тестирајте своје решење.

Наставник усмерава ученике и прати њихов рад. Наставник дискутује са ученицима о решењима.

[Commit*: `97dddc4beba40ac785c7413bb245ba849cd956d2`*]**

5.2.3. СЦЕНАРИО: **Увод у конструкторе у *Greenfoot* окружењу**

Табела 11. Увод у конструкторе у Greenfoot окружењу

Назив	Увод у конструкторе у *Greenfoot* окружењу
Образовни циљеви и исходи учења	После ове лекције, ученици ће суштински разумети концепт конструктора. У оквиру овог наставног сценарија обрађују су основни теоријски концепти везани за конструкторе, тумачи се одговарајући кôд, а затим се, кроз практичан рад, примењује стечено знање. Разумевање се продубљује кроз развој игре у *Greenfoot* окружењу, чиме се подстиче креативност, тимски дух и страственији приступ програмирању.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује концепте објектно-оријентисаног програмирања. Потребно је да ученици буду упознати са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **65**-минута. 1. Основни теоријски концепти (10 минута) 2. Тумачење кôда (20 минута) 3. Задатак **5.2.** (5 минута) 4. Задатак **5.3.** (30 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Кроз овај наставни сценарио ученицима ће се приближити концепт конструктора, а из перспективе развоја игара у *Greenfoot* окружењу. Лекција почиње десетоминутним уводом у оквиру којег наставник објашњава основне теоријске концепте. Након тога следи блок од 20 минута, током којег се тумаче и дискутују различити примери кôда у *Greenfoot* окружењу. Током наредних 5 минута ученици решавају **Задатак 5.2.,** у оквиру којег се од њих очекује да преименују претходно дефинисану класу `MyWorld`. Нови назив класе треба да буде `Arena`. Важно је обратити пажњу на то да је такође потребно преименовати и конструктор класе. Последњих 30 минута је посвећено практичном раду, па ученици решавају **Задатак 5.3.** који се односи на креирање сопствене почетне поставке протагониста у Арени, при чему их наставник усмерава, а по завршетку им пружа повратне информације. У оквиру ове активности предвиђено је и међусобно вредновање. Ученици ће наставити рад на игри која је изложена и започета током претходних лекција. Сходно томе, до краја ове лекције у игру ће бити уведени нови концепти, који се односе на конструкторе.

Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе. Статус самог пројекта отвара простор за дефинисање домаћих задатака. У овом контексту, могу се увести нове класе, изрази и вредности, како би се обезбедило додатно сложеније понашање (нпр. телепортовање, тунели итд.). Наставник може да поразговара са ученицима о различитим могућностима, а затим им може задати, као домаћи задатак, да имплементирају одређено понашање.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

5.2.3.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Основни теоријски концепти (10 минута)

Циљ: Разумевање концепта конструктора**.**

Концепти за дискусију: конструктори, класе, објекти, кључне речи: `super`, `new`, `this`.

Активности: Наставник уводи концепт **конструктора** у контексту концепата класе и објекта у објектно-оријентисаном програмирању. Наставник објашњава да се конструктори користе за иницијализацију конкретне инстанце (тј. објекта) класе.

2. Тумачење кôда (20 минута)

Циљ: Продубљивање разумевање концепта конструктора и повезивање са примерима из праксе**.**

Концепти за дискусију: конструктори, класе, објекти, методе, параметри, кључне речи: `super`, `new`, `this`, примери примене конструктора у пракси.

Активности: Наставник разматра конструкторе у контексту концепата класе и објекта: конструктори се користе за иницијализацију конкретних инстанци (тј. објеката) класе. Поред тога, конструктори се аутоматски позивају, кад год се креира неки објекат, а могу бити имплицитно или експлицитно дефинисани. Постоје подразумевани конструктори (који су имплицитно дефинисани), као и параметризовани и непараметризовани конструктори (који су експлицитно дефинисани од стране програмера). Наставник такође треба да појасни разлику између параметризованих и непараметризованих конструктора. Како би приближио овај концепт ученицима, наставник треба да користи реалне примере.

3. Задатак **5.2.** (5 минута)

Циљ: Промена назива класе**.**

Концепти за дискусију: конструктори, класе, објекти.

Активности: Наставник задаје ученицима задатак:

> **Задатак 5.2. (Промена назива класе `MyWorld` у `Arena`)**: Преименујте претходно дефинисану класу `MyWorld`. Нови назив класе треба да буде `Arena`. Поред тога, неопходно је да преименујете и конструктор класе `MyWorld()` у `Arena()`.

[Commit*:* aaf73c9bfd9f76a2a1e504f5e78d2976f1cada12**]**

4. Задатак **5.3.** (30 минута)

Циљ: Разматрање односа конструктора и стања игре**.**

Концепти за дискусију: конструктори, класе, објекти, методе, параметри, кључне речи: `super`, `new`, `this`.

Активности: Наставник задаје ученицима задатак:

> **Задатак 5.3. (Креирање сопствену почетне поставке протагониста у Арени)**: Креирајте сопствену почетну поставку протагониста у Арени. Протагонисти треба да буду постављени на одговарајуће позиције унутар конструктора класе `Arena`. Потребно је креирати једну инстанцу класе `Enemy`, једну инстанцу класе `Orb` и барем једну инстанцу класе `Direction`. Након што декларишете и иницијализујете променљиве жељеног типа, потребно је да подесите својства креираних објекта позивањем одговарајућих метода. Коначно, креиране објекте треба да поставите на жељене позиције у арени позивањем методе `addObject(Actor,int,int)`. Тестирајте своје решење.

[Commit*:* 8b105ea2eaf697f08c321efe687ddd31e2d0a041**]**

5.2.4. СЦЕНАРИО: **Увод у атрибуте у *Greenfoot* окружењу**

Табела 12. Увод у атрибуте у Greenfoot окружењу

Назив	Увод у атрибуте у *Greenfoot* окружењу
Образовни циљеви и исходи учења	После ове лекције, ученици ће суштински разумети концепт атрибута. У оквиру овог наставног сценарија обрађују су основни теоријски концепти који се односе на атрибуте, а затим се, кроз практичан рад, примењује стечено знање. Разумевање се продубљује кроз развој игре у *Greenfoot* окружењу, чиме се подстиче креативност, тимски дух и страственији приступ програмирању.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује концепте објектно-оријентисаног програмирања. Потребно је да ученици буду упознати са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **100**-минута. 1. Задатак **5.4.** (30 минута) 2. Атрибути (10 минута) 3, Параметри конструктора (10 минута) 4. Задатак **5.5.** (20 минута) 5. Задатак **5.6.** (30 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Током ове лекције, ученици средњих школа се упознају са концептима везаним за атрибуте у контексту развоја игре користећи Greenfoot алат. Сесија почиње 30-минутним задатком који се односи на проблеме везане за кретање и потенцијална решења. На основу претходног задатка, концепти везани за атрибуте се уводе у следећој 10-минутној сесији. Након тога следи 10-минутни сценарио током којег се објашњавају и дискутују параметри конструктора. Следећа 20-минутна сесија под вођством наставника односи се на задатак. Дефинисан је нови атрибут који се односи на кретање у класи Enemy. Поред тога, дефинисан је параметарски конструктор. Повратне информације наставника и ученика укључене су у ову сесију. Коначно, у последњој 30-минутној сесији под вођством наставника, имплементира се кретање непријатеља. У овом контексту, метода act() је ажурирана. Повратне информације наставника и ученика укључене су у ову сесију. Ученици ће наставити рад на пројекту игре започетом на претходним сесијама. Сходно томе, до краја сесије, они ће бити упознати са концептима који се односе на атрибуте.

Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе. Статус самог пројекта отвара простор за дефинисање домаћих задатака. У овом контексту, могу се увести нове класе, изрази и вредности, како би се обезбедило додатно сложеније понашање (нпр. телепортовање, тунели итд.). Наставник може да поразговара са ученицима о различитим могућностима, а затим им може задати, као домаћи задатак, да имплементирају одређено понашање.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

5.2.4.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Задатак **5.4.** (30 минута)

   Циљ: Разматрање проблема везаног за померање непријатеља и могућег решења у контексту конструктора и атрибута**.**

   Концепти за дискусију: конструктори, атрибути, методе.

   Активности: Наставник излаже задатак:

   > **Задатак 5.4. (Идентификовање проблема везаног за кретања непријатеља и предлог решења)**: Да ли уочавате неки проблем везан за кретања непријатеља? Шта је узрок? Како би се могао решити?

   Наставник указује на то да се објекти класе `Enemy` тренутно померају за по две ћелије одједном. Да би се ово решило, брзина непријатеља се може другачије моделовати тако да се непријатељ увек помера само за по једну ћелију. Међутим, може се додатно увести нови атрибут `moveDelay`, који ће омогућити увођење задршке, тако да ће се непријатељ померити тек након што се метода `act()` изврши одређени број пута.

2. Атрибути (10 минута)

   Циљ: Разумевање концепта атрибута**.**

   Концепти за дискусију: конструктори, класе, објекти, атрибути.

   Активности: Наставник уводи концепт **атрибута** у контексту концепата класе и објекта у објектно-оријентисаном програмирању.

3, Параметри конструктора (10 минута)

   Циљ: Разумевање концепта параметара конструктора**.**

   Концепти за дискусију: конструктори, класе, објекти, атрибути, параметри, кључне речи: `super`, `new`, `this`.

   Активности: Наставник уводи концепт **параметара конструктора** у контексту концепата класе и објекта у објектно-оријентисаном програмирању.

4. Задатак **5.5.** (20 минута)

   Циљ: Утврђивање знања о атрибутима и параметрима конструктора**.**

   Концепти за дискусију: конструктори, класе, објекти, атрибути, параметри, кључне речи: `super`, `new`, `this`.

   Активности: Наставник задаје ученицима задатак:

   > **Задатак 5.5. (Додавање атрибута `moveDelay` класи `Enemy`)**: Потребно је додати атрибут `moveDelay` (типа `int`) у класу `Enemy`. Потребно је и имплементирати параметризовани конструктор који треба да обезбеди да се дати атрибут иницијализује са вредношћу параметра. Коначно, потребно је ажурирати и кôд у класи `Arena` у складу са уведеним изменама.

   [Commit*:* `6092489ce57541e77ae4e2ee886b20853df9f8a4`**]**

5. Задатак **5.6.** (30 минута)

   Циљ: Продубљивање разумевања атрибут и параметара конструктора**.**

   Концепти за дискусију: конструктори, класе, објекти, атрибути, параметри, кључне речи: `super`, `new`, `this`.

   Активности: Наставник задаје ученицима задатак:

Задатак 5.6. (Имплементација кретања непријатеља уз задршку): Потребно је изменити методу `act()` класе `Enemy`, тако да се обезбеди да ће се непријатељ померити тек након што се метода `act()` изврши одређени број (`moveDelay`) пута. Такође је потребно додати нови атрибут `nextMoveCounter` типа `int` и иницијализовати га са **0**. Затим треба изменити методу `act()` тако да позива `this.move(1)` само ако атрибут `nextMoveCounter` има вредност **0**. Након померања, атрибут `nextMoveCounter` добија вредност атрибута `moveDelay`. Када непријатељ не може да се помери (јер `nextMoveCounter` још није достигао вредност **0**), потребно је смањити вредност атрибута `nextMoveCounter` за **1**.

[Commit*:* `bf26e6ed23911ccb712fae3e243cdedff3a89a7f`**]**

5.2.5. СЦЕНАРИО: **Увод у преклапање конструктора у *Greenfoot* окружењу**

Табела 13. Увод у преклапање конструктора у Greenfoot окружењу

Назив	Увод у преклапање конструктора у *Greenfoot* окружењу
Образовни циљеви и исходи учења	После ове лекције, ученици ће суштински разумети идеју преклапања конструктора. У оквиру овог наставног сценарија обрађују су основни теоријски концепти везани за преклапање конструкторе, а затим се, кроз практичан рад, примењује стечено знање. Разумевање се продубљује кроз развој игре у *Greenfoot* окружењу, чиме се подстиче креативност, тимски дух и страственији приступ програмирању.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује концепте објектно-оријентисаног програмирања. Потребно је да ученици буду упознати са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **75**-минута. 1. Основни теоријски концепти (5 минута) 2. Задатак **5.7.** (25 минута) 3. Задатак **5.8.** (25 минута) 4. Обнављање теоријских концепата (20 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Кроз овај наставни сценарио ученицима ће се приближити идеја преклапања конструктора, а из перспективе развоја игара у *Greenfoot* окружењу. Лекција почиње петоминутним уводом у оквиру којег наставник објашњава основне теоријске концепте. Током наредних 25 минута ученици решавају **Задатак 5.7.**, у оквиру којег се од њих очекује да дефинишу параметризовани конструктор, при чему их наставник усмерава, а по завршетку им пружа повратне информације. Следи блок од 25 минута током којих ученици треба да практично примене стечена знања о преклапању конструктора (**Задатак 5.8.**). У оквиру обе ове активности предвиђено је и међусобно вредновање. Последњи блока од 20 минута, посвећен је теоријском осврту, односно рекапитулацији свих обрађених концепата (као што су променљиве, оператори, изрази, конструктори, атрибути и преклапање конструктора).
Вредновање	Увођење елемената игре и надметања у наставу (енгл. *gamification*) омогућава само неформално вредновање, али ће повећати ниво заинтересованости и истинске мотивисаности, као и степен усвојеног знања целокупне групе.

	Статус самог пројекта отвара простор за дефинисање домаћих задатака. У овом контексту, могу се увести нове класе, изрази и вредности, како би се обезбедило додатно сложеније понашање (нпр. телепортовање, тунели итд.). Наставник може да поразговара са ученицима о различитим могућностима, а затим им може задати, као домаћи задатак, да имплементирају одређено понашање.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

5.2.5.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Основни теоријски концепти (5 минута)

   Циљ: Усвајање концепта преклапања конструктора**.**

   Концепти за дискусију: конструктори.

   Активности: Наставник обрађује концепт преклапања конструктора.

2. Задатак **5.7.** (25 минута)

   Циљ: Утврђивање знања о параметризованим конструкторима**.**

   Концепти за дискусију: конструктори, параметри, атрибути, параметризовани конструктори.

   Активности: Наставник задаје ученицима задатак:

   > **Задатак 5.7. (Имплементација параметризованог конструктора класе `Direction`)**: Дефинишите параметризовани конструктор у класи `Direction`. Конструктор треба да прима један параметар: `rotation` типа `int`. У телу конструктора је потребно ротирати креирану инстанцу у складу са вредношћу параметра. Потребно је ажурирати и кôд у класи `Arena` у складу са уведеним изменама.

   [Commit*:* `3c4b9ef57ab17bac2a0abc7fc5e76ea4b6e27e4b`**]**

3. Задатак **5.8.** (25 минута)

   Циљ: Практична примена преклапања конструктора**.**

   Концепти за дискусију: конструктори, преклапање конструктора.

   Активности: Наставник задаје ученицима задатак:

   > **Задатак 5.8. (Преклапање конструктора у класи `Direction`)**: Дефинишите додатни непараметризовани конструктор у класи `Direction`. У телу овог конструктора је потребно позвати претходно дефинисани параметризовани конструктор са аргументом постављеним на **0** (ово одговара ротацији од **0°**). Потребно је ажурирати и кôд у класи `Arena` позивајући непараметризовани конструктор класе `Direction`, где је то могуће.

   [Commit*:* `1e67e67523c66acea4e93363c9a3173302f424c8`**]**

4. Обнављање теоријских концепата (20 минута)

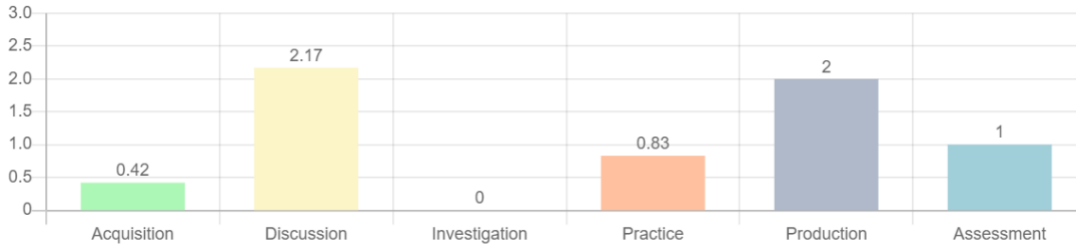
   Циљ: Резимирање целокупне области**.**

   Концепти за дискусију: промењљиве, оператори, изрази, конструктори, атрибути, преклапање конструктора.

   Активности: Наставник даје резиме лекције и осврће се заједно са ученицима, на све претходно обрађене концепте.

6. АСОЦИЈАЦИЈЕ

6.1. Структура наставних активности и задаци



Слика 10. Расподела ангажовања ученика према типу учења за тематску целину „Асоцијације“

6.2. Наставни сценарији

Припремљена су четири наставна сценарија за тематску целину „Асоцијације“.

6.2.1. СЦЕНАРИО: *Greenfoot* објекти у задатку : Истраживање метода и асоцијација

Табела 14. Greenfoot објекти у задатку : Истраживање метода и асоцијација

Назив	Greenfoot објекти у задатку : Истраживање метода и асоцијација	
Образовни циљеви и исходи учења	До краја сценарија,ученици треба да стекну солидно разумевање о томе како објекти могу да ступају у интеракцију једни са другима. Инстанца класе Enemy је у интеракцији са другим објектима, посебно класом Orb, унутар Греенфоот окружења. Требало би да покажу стручност у креирању и позивању метода унутар Јава класа, посебно у имплементацији и тестирању метода Arena.respawn(Enemy) и Orb.hit(Enemy). Поред тога, ученици треба да разумеју и ефикасно управљају атрибутима класе, укључујући дефинисање и коришћење атрибута Enemy.attack и Orb.hp.Ученици треба да буду вешти у инкапсулирању података у оквиру класе, што се демонстрира креирањем гетера као што је getEnemy.attack(),и разумеју важност енкапсулације података за сигуран и одржаван код. Они треба да разумеју и имплементирају преношење порука између објеката, обезбеђујући да инстанце класа ефикасно комуницирају како би извршиле радње у игри. Штавише, студенти треба да примењују технике позивања метода за решавање задатака развоја интерактивне игре, ефикасно користећи синтаксу и параметре потребне за позивање метода.	
Циљна група	Ученици средњих школа похађају курс OOP4Fun. Основно знање програмирања укључујући концепте итерације и селекције. Ученике треба упознати са Greenfoot-ом.	
Временски план	Задатак 5.1 - Дискусија о томе шта би требало да се деси када непријатељ стигне до кугле (10 минута) Задатак 5.2 – Дискусија о томе како инстанца класе Enemy треба да ступи у интеракцију са релевантним објектима користећи поруке када погодите инстанцу класе Orb (15 минута) Задатак 5.3 –АтрибутиEnemy.attack и Orb.hp (10 минута) Метода (15 минута) Задатак 5.4- Гетер атрибута Enemy.attack (5 минута) Задатак 5.5-Креирање и тестирање методеArena.respawn(Enemy)(10 минута) Задатак 5.6 - Креирање и тестирање методеOrb.hit(Enemy) (10 минута)	
Материјали и ресурси	Уџбеник из пројекта OOP4Fun. Ресурси из пројекта OOP4Fun. Изворни код пројекта из Github/Gitlab. Интернет ресурси.	
Опис	У овом 75-минутном сценарију учења, средњошколци ће заронити у интеракције објеката, креирање метода и руковање атрибутима унутар Greenfoot окружења. Ова сесија има за циљ да побољша разумевање ученика о томе како објекти	

	комуницирају и интерагују, што је кључни аспект објектно оријентисаног програмирања. Сесија почиње 10-минутном дискусијом о динамици између непријатељских објеката и кугли, истражујући шта треба да се деси када непријатељ стигне до кугле. Ово ће поставити основу за разумевање интеракција објеката у контексту игре. Након тога, ученици ће се укључити у 15-минутни задатак расправљајући о томе како инстанца класе Enemy треба да ступи у интеракцију са релевантним објектима користећи поруке када се судари са инстанцом класе Orb. Ова дискусија ће нагласити важност објектне комуникације и преношења порука. Затим, 10-минутни задатак ће се фокусирати на атрибуте Enemy.attack и Orb.hp. Ученици ће дефинисати и разумети ове атрибуте, кључне за управљање механиком игре. У наредних 15 минута, ученици ће применити методе, учећи како да их креирају и имплементирају у оквиру својих часова. Овај одељак ће учврстити њихово разумевање креирања и позивања метода. У 5-минутном задатку ученици ће морати да направе гетер за атрибут Enemy.attack, продубљујући своје знање о инкапсулацији и проналажењу података. Следећих 10 минута биће посвећено креирању и тестирању методеArena.respawn(Enemy). Ученици ће применити ову методу за руковање логиком поновног покретања за непријатељске објекте, обезбеђујући њихово разумевање функционалности методе и тестирање. Након тога, још 10-минутни задатак ће укључивати креирање и тестирање методе Orb.hit(Enemy), који ће управљати логиком интеракције када непријатељ удари куглу. Током сесије, ученици ће радити појединачно или у малим групама, промовишући сарадњу и вршњачко учење. Активним учешћем у дискусијама, задацима кодирања и методама тестирања, ученици ће развити вештине критичког мишљења и способности рачунарског решавања проблема. На крају сесије, студенти ће стећи свеобухватно разумевање интеракција објеката, креирања метода и управљања атрибутима у Греенфооту. Ове основне вештине ће их опремити за даље пројекте развоја игара и побољшати њихову општу вештину програмирања.	
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе.	
Дељење решења	Да би се њихови резултати дистрибуирали наставницима и колегама студентима, користиће се уобичајено подешавање Github/Gitlab и система за управљање учењем (нпр. Моодле). Ученици могу да наставе дискусију о овој теми на форуму који им је обезбеђен преко алата за управљање учењем.	

6.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1) **Задатак 5.1 - Дискусија о томе шта би требало да се деси када непријатељ стигне до кугле (10 минута)**

   Циљ: Наставник води дискусију о очекиваном понашању када непријатељ стигне до кугле. Ученици се подстичу да размишљају о динамици игре и исходима, као што је кугла која наноси штету или се игра завршава.

   Концепти за дискусију: Оштећење кугле, уклањање непријатеља, покретање догађаја у игрици (нпр. смањење здравља, играње звучних ефеката, завршетак игре).

   Активност: Лекција почиње прегледом претходно обрађених концепата како би се осигурало да ученици имају чврсту основу за нови материјал. Наставник укључује ученике у дискусију како би разјаснио концепте интеракције објеката, порука и метода у контексту Greenfoot окружења.

   Ученици сарађују у групама како би смислили исходе када непријатељ дође до кугле у њиховој игри. Они истражују алгоритме као што је смањење енергије(HP) кугле након контакта са непријатељем, расправљајући о сценаријима где би HP кугле могао да падне на нулу, што доводи до завршетка игре. Алтернативно, ако HP кугле остане изнад нуле, они се слажу да поново покрену непријатеља на другој локацији арене. Они такође разматрају интеграцију додатних догађаја у игри које покреће ова интеракција, као што су звучни ефекти или поруке на екрану. Током читаве сесије, наставник води дискусију, подстичући ученике да ускладе своје предложене алгоритме са одређеним сценаријем: обезбеђујући да када непријатељ стигне до кугле, HP кугле се смањује као што је претходно наведено.Ово упутство помаже ученицима да практично примене своје идеје, ојачавајући њихово разумевање динамике игре и интеракција у оквиру њихове игре.

2) **Задатак 5.2 – Дискусија о томе како инстанца класног непријатеља треба да ступи у интеракцију са релевантним објектима користећи поруке када погодите инстанцу класне кугле (15 минута)**

   Циљ: Наставник води ученике кроз процес разумевања како инстанца класе Enemy треба да ступи у интеракцију са другим објектима, посебно када удари у инстанцу класе Orb, користећи поруке.

   Концепти за дискусију: Позивање метода, преношење параметара, референце објеката.

   Активност:Ученици сарађују у паровима како би истражили како инстанца класе Enemy треба да ступи у интеракцију са инстанцом класе Orb користећи поруке у оквиру свог сценарија игре. Они анализирају и мапирају редослед порука и акција које треба да се одвијају када непријатељ удари куглу. Ова вежба има за циљ да продуби њихово разумевање позивања метода, преношења параметара и референци на објекте у контексту развоја игре. Док ученици дискутују и усавршавају своје идеје, наставник води сесију, усмеравајући их да обезбеде да редослед интеракција између објеката прати алгоритам који се шири на објекте који сарађују, као што је наведено у циљевима лекције. Да би помогао у овом процесу, наставник може да уведе и користи УМЛ дијаграм секвенце да визуелно опише интеракције између класа Enemy, Orb, Arena и Greenfoot. Овај визуелни алат помаже студентима да боље схвате ток порука и позива метода, ојачавајући њихово разумевање концепта објектно оријентисаног програмирања и њихову примену у Јава програмирању унутар Greenfoot окружења.

3) **Задатак 5.3 - Атрибут Enemy.attack и Orb.hp (10 минута)**

   Циљ: Наставник уводи атрибуте Enemy.attack и Orb.hp ,објашњавајући њихов значај у одређивању исхода интеракција**.**

   Концепти за дискусију: Атрибути класе, инкапсулација.

Активност: Наставник уводи концепт атрибута класе и енкапсулације, објашњавајући како атрибутиEnemy.attack и Orb.hp могу представљати битне карактеристике објеката у оквиру игре. Ученици уче да дефинишу и користе ове атрибуте у свом коду за моделирање непријатељске снаге напада и моћикугле. Они почињу додавањем новог целобројног атрибута који се зове напад у класу Enemy, укључујући параметар у конструктору за иницијализацију овог атрибута. Слично, они додају целобројни атрибут који се зове hp класи Orb, заједно са параметарским конструктором за постављање ове вредности приликом креирања објекта. Наставник води ученике кроз прилагођавање кода класе Arene да би се прилагодили овим новим атрибутима. Ово практично искуство помаже ученицима да разумеју улогу атрибута класе у интеракцијама објеката и важност енкапсулације у одржавању интегритета и безбедности кода.

Commit: 4ca1e9f25685990d2bdfe5b610c28422e0944f95

Метод (15 минута)

Циљ: Наставник даје преглед метода, објашњавајући како се оне користе за инкапсулацију радњи и понашања у оквиру наставе**.**

Концепти за дискусију: Дефиниција методе, позивање метода, параметри, повратне вредности.

4) **Активност:** Наставник почиње објашњавањем концепта метода као инкапсулираних радњи или понашања унутар разреда. На практичним примерима наставник демонстрира синтаксу и структуру дефиниција метода, илуструјући како се методе позивају на објекте. Ученици уче о различитим типовима метода, укључујући оне које изводе радње (void методе) и оне које враћају вредности (методе типа враћања). Наставник објашњава како се параметри прослеђују методама, наглашавајући важност типова параметара и редоследа. Кроз вођене вежбе кодирања, студенти вежбају дефинисање метода са различитим типовима параметара и повратa и позивање ових метода на инстанцама објекта. Они истражују сценарије у којима методе изводе радње, модификују стања објеката или враћају специфичне вредности, учвршћујући њихово разумевање функционалности метода унутар класе.

5) **Задатак 5.4 - Гетер атрибута Enemy.attack (5 минута)**

Циљ: Наставник објашњава појам гетер метода и њихову сврху у приступу вредностима атрибута.

Концепти за дискусију: Методе приступа (гетери), инкапсулација.

Активност: Наставник почиње објашњавањем сврхе гетер метода, наглашавајући како оне обезбеђују контролисан приступ вредностима атрибута уз одржавање енкапсулације. Ученици уче важност коришћења гетера за преузимање вредности приватних атрибута, појачавајући концепт заштите података у оквиру класе. Наставник затим води ученике кроз процес креирања гетер методе за атрибут напада у класиEnemy. Користећи практичан приступ, ученици имплементирају гетер метод, осигуравајући да враћа вредност атрибута напада. Наставник демонстрира исправну синтаксу и структуру за дефинисање гетера и како се користи унутар кода за приступ вредности атрибута. До краја активности, ученици би требало да буду у стању да креирају и користе гетер методе за приступ вредностима атрибута на контролисан начин, побољшавајући њихово разумевање енкапсулације и заштите података у објектно оријентисаном програмирању.

Commit*:* 72b7456ea4cc11416c57d72c89b6a7f7e9266e3e

6) **Задатак 5.5 – Креирање и тестирање методеArena.respawn(Enemy) (10 минута)**

Циљ: Наставник води ученике кроз креирање и тестирањеArena.respawn(Enemy) методе, који се бави поновним појављивањем непријатеља у игри.

Концепти за дискусију: имплементација методе, тестирање, механика игре**.**

Активност: Наставник почиње увођењем концепта имплементације методе и њеног значаја у дефинисању специфичних понашања у оквиру класе. Наглашавајући практичну примену,

ученици су вођени да креирају respawnметоду класеArena. Овај метод, који не враћа вредност, узима један параметар типа Enemy. Ученицима се налаже да у оквиру ове методе подесе локацију и ротацију непријатеља тако да одговарају вредностима које су првобитно постављене у конструктору. Наставник демонстрира исправну синтаксу и структуру за дефинисање ове методе, појачавајући кључне концепте имплементације методе и преношења параметара.

Затим ученици тестирају свој метод како би се уверили да исправно функционише. Они креирају инстанцу Arene и непријатеља, али не покрећу апликацију одмах. Уместо тога, они превлаче Enemy инстанцу на нову локацију, а затим приступају контекстуалном менију инстанце Arene да би позвали метод respawn. Наставник објашњава процес обезбеђивања паузирања апликације и активног поља параметара, водећи ученике да кликну десним тастером миша на непријатељску инстанцу да би исправно попунили поље параметра. Ученици посматрају израз уграђен у прозор, а затим кликну на дугме ОК да виде ефекат њихове методе поновног покретања.

Кроз ову активност ученици стичу практично искуство у писању и методама тестирања, разумевању како да програмски манипулишу објектима игре. Наставник осигурава да ученици схвате сваки корак, пружајући помоћ и појашњење по потреби, чиме се учвршћује њихово разумевање имплементације метода и механизме игре.

Commit*:* 43a221876b8acb4fd507175ec4c8f520121d1ab1

7) Задатак 5.6 – Креирање и тестирање метод Orb.hit(Enemy) (10 минута)

Циљ: Наставник упућује ученике на израду и провериOrb.hit(Enemy) метод, који дефинише интеракцију када непријатељ погоди куглу.

Концепти за дискусију: Интеракција метода, ажурирање стања објекта**.**

Активност: Наставник почиње објашњавањем сврхеOrb.hit(Enemy) методе, наглашавајући како обухвата логику интеракције између кугле и непријатеља. Ученици су затим вођени да додају ову методу у класу Orb. Овај метод, који не враћа вредност, узима један параметар типа Enemy.

Да би тестирали методу, ученици прате поступак корак по корак сличан оном који се користи за тестирање методе поновног појављивања. Они стварају инстанцу класе Orb и инстанцу класе Enemy. Без покретања апликације, они позивају контекстни мени инстанце Orb и бирају метод ударца. Наставник обезбеђује да ученици разумеју како да попуне поље параметра тако што десним тастером миша кликну на Enemy инстанцу док је апликација паузирана. Ова радња гради израз за позив методе, који ученици затим извршавају кликом на дугме ОК.

Наставник наглашава важност посматрања израза уграђеног у прозор за проверу позива методе. Ова вежба помаже ученицима да разумеју интеракцију метода и процес ажурирања стања објеката у контексту игре. Кроз ову практичну активност, ученици стичу практично искуство у примени и тестирању метода, ојачавајући своје разумевање интеракције метода и понашања у игри. Наставник пружа подршку и појашњење по потреби, осигуравајући да ученици успешно заврше задатак и разумеју његов значај.

Commit*:* fe03d520260f172066be35055a901487bf7c2ff7

6.2.2. СЦЕНАРИО: *Greenfoot* објекти у задатку: истраживање асоцијација и позиви напредних метода

Табела 15. Greenfoot објекти у задатку: истраживање асоцијација и позиви напредних метода

Назив	Greenfoot објекти у задатку: истраживање асоцијација и позиви напредних метода
Образовни циљеви и исходи учења	До краја сценарија, ученици треба да развију дубље разумевање о томе како објекти у различитим класама могу да формирају асоцијације и да ефикасно делују у Greenfoot окружењу. Ученици треба да покажу стручност у позивању Orb.hit(Enemy) методе из класе Enemy, показујући њихову способност да иницирају напредне позиве метода и олакшају комуникацију између објеката. Требало би да буду у стању да објасне функционалност кључних Greenfoot метода као што су Greenfoot.stop() и World.getWorldOfType(_cls_), разумевање њихове улоге у контроли извршења игре и ефикасном управљању инстанцама објеката, а такође, ученици треба да успешно имплементирају Orb.hit(Enemy) методу у оквиру својих пројеката, интегришући интеракције објеката и функционалности метода за креирање интерактивне и динамичке механике игре. Студенти треба да примењују основне принципе објектно оријентисаног програмирања, укључујући инкапсулацију и позивање метода, да би развили софистициране интеракције и функционалности у игри. Требало би да стекну практичан увид у различите аспекте развоја игре, укључујући појаву непријатеља, управљање стањем игре и стварање занимљивог искуства играча кроз интеракције структурираних објеката.
Циљна група	Ученици средњих школа похађају курс OOP4Fun.Основно знање програмирања укључујући концепте итерације и селекције. Ученике треба упознати са Greenfoot - ом.
Временски план	1. Асоцијације (10 минута) 2. Задатак 5.7 – Позивање методеOrb.hit(Enemy) из Enemy (15 минута) 3. Објашњење кода за методе Greenfoot.stop() и World.getWorldOfType(_cls_) (15 минута) 4. Задатак 5.8 – Имплементације методе Orb.hit(Enemy)(30 минута)
Материјали и ресурси	Уџбеник из пројекта OOP4Fun. Ресурси из пројекта OOP4Fun. Изворни код пројекта из Github/Gitlab. Интернет ресурси.
Опис	У овом 70-минутном сценарију учења, средњошколци ће ући у детаље асоцијација између објеката и напредних позива метода унутар Greenfoot окружења. Сесија има за циљ да продуби учениково разумевање концепта објектно оријентисаног програмирања и побољша њихову способност да имплементирају сложене интеракције у развоју игара.

	Сесија почиње 10-минутном дискусијом о повезаности између класа, фокусирајући се на то како објекти могу да комуницирају и сарађују у оквиру софтверског система. Ово основно разумевање поставља терен за истраживање сложенијих интеракција. Након тога, ученици ће се ангажовати у 15-минутном задатку који има за циљ позивање методе Орб.хит(Непријатељ) из Enemy класе. Овај задатак наглашава практичну примену позивања метода и преношења порука између објеката. Следећи сегмент укључује детаљно објашњење кода за методе Greenfoot.stop() и World.getWorldOfType(_cls_), траје 15 минута. Студенти ће стећи увид у то како ове методе функционишу у оквиру Greenfoot оквира, омогућавајући прецизну контролу над елементима игре и управљањем светом. Срж сесије је посвећен 30-минутном задатку где ће ученици применити метод Orb.hit(Enemy). Овај задатак изазива ученике да примене своје разумевање имплементације метода, преношења параметара и интеракције објеката како би креирали функционални механизам игре у оквиру својих пројеката. На крају сценарија, ученици ће се појавити са дубљим разумевањем асоцијација између објеката, вештином у напредним позивима метода као што су Orb.hit(Enemy) из Enemy класе, и увид у примену кључних Greenfoot метода. Ове вештине ће их опремити да креирају интерактивније и динамичније игре, користећи пуни потенцијал принципа објектно оријентисаног програмирања у развоју игара.
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе.
Дељење решења	Да би се њихови резултати дистрибуирали наставницима и колегама студентима, користиће се уобичајено подешавање Github/Gitlab и система за управљање учењем (нпр. Моодле). Ученици могу да наставе дискусију о овој теми на форуму који им је обезбеђен преко алата за управљање учењем.

6.2.2.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1) Асоцијација

Циљ: Наставник објашњава концепт асоцијације између објеката, наглашавајући како објекти различитих класа могу међусобно да комуницирају**.**

Концепти за дискусију: Асоцијације, интеракције објеката, односи класа.

Активност: Час почиње кратким прегледом асоцијација између класа у објектно оријентисаном програмирању. Наставник укључује ученике у дискусију како би разјаснио како објекти комуницирају једни са другима кроз асоцијације, користећи практичне примере изGreenfoot окружења даилуструју ове концепте. Ученици улазе у разумевање да асоцијације дефинишу како класе сарађују, као што је како непријатељ може да утиче на куглу у сценарију игре.

Кроз детаљну дискусију, наставник елаборира различите типове асоцијација унутар Greenfoot -а: један-према-један, један-према-више и много-према-многима. На пример, асоцијација један-на-један може да илуструје како је играч повезан са својим аватаром лика, асоцијација један-према-више може показати како свет садржи више инстанци актера, а асоцијација много-према-више може да прикаже како различити непријатељи комуницирају са више кугли на нивоу игре.

Користећи УМЛ дијаграме класа специфичних за Greenfoot, наставник визуелно демонстрира ове односе, помажући ученицима да разумеју како су асоцијације структурисане и имплементиране у њиховим пројектима игара. На пример, дијаграм би могао да прикаже како је класа непријатеља повезана са вишеструким инстанцама класе Orb, указујући на однос један-према-више где сваки непријатељ утиче на неколико кугли.

Ученици активно учествују у идентификацији и мапирању ових асоцијација у оквиру својих пројеката игара. Они истражују како објекти интерагују на основу ових односа и расправљају о импликацијама на механику и логику игре. Наставник даје конкретне примере из контекста њиховог развоја игре, илуструјући како непријатељске инстанце ступају у интеракцију са инстанцама Орб и како овим интеракцијама управљају асоцијације.

До краја активности, ученици стекну солидно схватање улоге коју асоцијације играју у дизајнирању и имплементацији интерактивних система унутар Greenfoot -а. Они могу идентификовати различите типове асоцијација и применити ово знање за ефикасно моделовање и имплементацију сложених интеракција између Greenfoot објеката. Ово практично разумевање јача њихову способност да дизајнирају кохезивне и интерактивне сценарије игара користећи објектно оријентисане принципе.

2) Задатак 5.7 –Позивање метода Orb.hit(Enemy) из Enemy

Циљ: Наставник води ученике кроз процес позивања методеOrb.hit(Enemy) из класе Enemy .

Концепти за дискусију: Позивање метода, референце објеката.

Активност: Ученици учествују у практичном учењу мењајући метод act() класе Enemy. Они уклањају постојећи код који је одговоран за непотребна понашања као што је ротирање након ударања кугле или одбијања од ивица света. Уместо тога, они имплементирају функционалност за позивањеOrb.hit(Enemy) метод када се инстанца Enemy судара са инстанцом Orb.

Наставник кроз практичне примере и демонстрације илуструје како правилно подесити позив методе. Ученици уче да користе референце објеката (ово) да покрену метод hit() на инстанци Orb након судара са непријатељем. Они истражују ток контроле у Greenfoot-у, разумевајући како позивање метода усмерава путању извршења у оквиру њиховог сценарија игре.

Током читаве активности, наставник даје упутства за отклањање грешака и тестирање имплементације како би се осигурало да позивање Orb.hit(Enemy) функционише како је

предвиђено. Ученици посматрају понашање у окружењу Greenfoot симулације, потврђујући да позив методе ефективно покреће очекиване интеракције између објеката Enemy и Orb.

До краја наставне јединице, ученици стичу вештину у позивању метода и интеракцији објеката у оквиру Greenfoot програмирања. Они разумеју како да користе референце објеката за позивање метода у различитим класама, појачавајући њихово разумевање принципа објектно оријентисаног програмирања у контексту развоја игре.

Commit*:* 63f9c96717d9d2587b60095e3b249b0158c8587b

3) Објашњење кода за методе Greenfoot.stop() и World.getWorldOfType(cls)

Циљ:Наставник објасни функционалностGreenfoot.stop()методе и World.getWorldOfType(_cls_) методе.

Концепти за дискусију: Методе контроле игре, управљање светом**.**

Активност: У овој лекцији ученици се упуштају у разумевање две кључне методе у оквиру Греенфоот оквира**:**Greenfoot.stop() и World.getWorldOfType(_cls_). Наставник почиње разјашњавањем сврхе и употребе сваке методе, наглашавајући њихову улогу у контроли игре и управљању светом.

У Greenfoot.stop()методи је од суштинског значаја за контролу тока извршења Greenfoot сценарија. Када се позове, зауставља симулацију и замрзава све актере и интеракције у свету. Овај метод је посебно користан за имплементацију функције паузирања игре или за покретање одређених догађаја који захтевају привремено заустављање напредовања игре.

С друге стране, World.getWorldOfType(_cls_) метода служи другој сврси везаној за управљање светом. Овај метод омогућава програмерима да преузму инстанце светова који су специфичног типа класе (_cls_). Он пролази кроз све активне светове у Greenfoot окружењу и враћа инстанце светске класе које одговарају наведеном типу. Ова могућност је корисна када програмери треба да комуницирају са световима или да динамички манипулишу световима на основу њихових атрибута класе.

Ученици учествују у практичним демонстрацијама и примерима како би продубили своје разумевање ових метода. Наставник показује како Greenfoot.stop()може се интегрисати у сценарије игре да би се створила функционалност паузе или покренула одређена дешавања у игри. Ученици посматрају како паузирање игре утиче на понашање и интеракције глумаца у окружењу Greenfoot симулације.

Слично, ученици истражују World.getWorldOfType(_cls_) метод кроз практичне вежбе. Они уче како да користе овај метод за динамичко преузимање инстанци специфичних типова света. Наставник демонстрира сценарије у којима је дохватање светова одређеног типа класе неопходно за имплементацију напредног механизма игре или управљање вишеструким истовременим окружењима игре у оквиру Greenfoot-а.

Током читаве активности, наставник подстиче дискусију и даје практичне примере кодирања како би илустровао примену ових метода у сценаријима развоја игара у стварном свету. Ученици активно учествују у експериментисању са методама у оквиру сопствених Greenfoot пројеката, појачавајући своје разумевање кроз директну примену и истраживање.

До краја наставне јединице ученици стичу вештине у коришћењу Greenfoot.stop()за контролу игре и World.getWorldOfType(_cls_) за ефикасно управљање светом у оквиру Greenfoot окружења. Они стичу практичне вештине које побољшавају њихову способност да имплементирају сложена понашања у игри и ефикасно управљају стањима игре користећи ове основне методе.

4) Задатак 5.8 –Имплементација методе Orb.hit(Enemy)

Циљ: Наставник води ученике кроз реализацијуOrb.hit(Enemy) методе.

Концепти за дискусију: Имплементација методе, ажурирање стања објекта, динамика игре.

Активност: Ученици почињу са реализацијомOrb.hit(Enemy) методе, кључни корак у дефинисању интеракције између непријатеља и кугле у оквиру њиховог сценарија игре.

Orb.hit(Enemy)метода игра кључну улогу у одређивању последица када се непријатељ судари са куглом у игри. Ево како ученици могу приступити и применити ову методу. Прво, смањите поене моћикугле. По позиву Orb.hit(Enemy), метода треба да смањи енергију(hp) кугле. Ова акција симулира штету коју кугла нанесе при удару са непријатељем. Након тога, проверите да ли је моћкугле пала на нулу. Акоенергија(hp) кугле достигну нулу или испод, игра би требало да се заврши. То се постиже позивањем Greenfoot.stop()да се заустави извођење игре. Тада је игра готова. Догађа се другачији сценарио ако јеенергијакугле и даље изнад нуле након непријатељског судара, ученици би требало да угради логику да поново покрену непријатеља у арену за континуирано играње.

У току активности наставник олакшава реализацију Orb.hit(Enemy) пружањем смерница о структури метода, коришћењу параметара (Enemy), и како ажурирати стање кугле (hp). Ученици сарађују како би разговарали и одлучили о специфичној динамици игре коју желе да имплементирају, као што је колико штете сваки тип непријатеља наноси кугли и шта се дешава када се поенимоћикугле потроше.

Наставник подстиче ученике да темељно тестирају своје имплементације, обезбеђујући да се метода понаша како се очекује у различитим сценаријима игре. До краја активности, студенти стичу практично искуство у примени методске логике за ефикасно управљање интеракцијама у игри, ојачавајући своје разумевање имплементације метода и механизмаигре у оквиру Greenfoot оквира.

(**Commit***:* 84bcd7c128faaa9313b507f7438f826ae2f47d2c)

6.2.3. СЦЕНАРИО: *Greenfoot* објекти у задатку *Tower*, *Bullet* и стратешке интеракције

Табела 16. Greenfoot објекти у задатку Tower, Bullet и стратешке интеракције

Назив	Greenfoot објекти у задатку Tower, Bullet и стратешке интеракције	
Образовни циљеви и исходи учења	До краја сценарија, ученици ће развити вештину у креирању класаBulletиTower, успостављајући основу за развој стратешке игре. Они ће разумети и применити реалистично кретање Bullet и одредити акције када инстанце Bullet наиђу на Enemy инстанце или ивицу арене. Ученици ће дизајнирати ефикасни механизам пуцања за инстанце Bullet, користећи преношење порука између објеката Tower и других елемената игре како би побољшали динамику играња. Они ће стратешки применити Tower инстанце унутар арене игре и примениће принципе оријентисане на објекте као што су инкапсулација и позивање метода како би се обезбедило стварање робусног механизма игре која се може одржавати. Кроз заједничко решавање проблема, студенти ће се позабавити изазовима у дизајнирању и имплементацији стратешких интеракција између кула, метака и елемената игре, стичући практичан увид у принципе дизајна игре и побољшавајући своје опште разумевање механизма одбране куле у Greenfoot.	
Циљна група	Ученици средњих школа похађају курс OOP4Fun. Основно знање о програмирању укључујући концепте итерације и селекције. Ученике треба упознати са Греенфоот-ом.	
Временски план	1) Задатак 5.9 –Креирање класа Bullet и Tower(10 minutes) 2) Задатак 5.10 – Дискусија о томе како би се инстанца класе Bullet требала да помери и шта треба да се деси када стигне до инстанце класе Enemy или ивице арене (10 минута) 3) Задатак 5.11–Имплементирање кретања инстанце класе Bullet (30 минута) 4) Задатак 5.12 – Дискусија о томе како ће инстанца класе Tower пуцати у инстанцу класе Bullet (15 минута) 5) Задатак 5.13 – Дискусијао томе како инстанца класе Tower треба да комуницира са релевантним објектима користећи поруке приликом имплементације (15 минута) 6) Задатак 5.14 – Имплементација инстанце класе Tower (30 минута) Задатак 5.15 - Towers у Arena (20 минута)	
Материјали и ресурси	Уџбеник из пројекта OOP4Fun. Ресурси из пројекта OOP4Fun. Изворни код пројекта из Github/Gitlab. Интернет ресурси.	
Опис	У овом 130-минутном сценарију учења, средњошколци ће се уронити у динамику интеракције куле и метака унутар Greenfoot окружења. Сесија се фокусира на	

	развијање вештина ученика у дизајнирању и примени стратешких елемената игре како би се створила занимљива и интерактивна игра. Сесија почиње 10-минутним задатком где ученици креирају класеBullet и Tower. Овај темељни корак поставља основу за разумевање и имплементацију интеракција између ових елемената игре. Након тога, следи 10-минутна дискусија о томе како инстанце класе Bullet треба да се крећу и радње које треба да се десе када метак стигне до инстанце класе Enemy или до ивице арене. Ова дискусија поставља терен за примену прецизне и динамичке механике кретања. Ученици затим посвећују 30 минута имплементацији кретања инстанци класе Bullet. Овај задатак изазива ученике да примене своје разумевање Greenfoot's move()методе и руковање догађајима за симулацију реалног понашања метка у окружењу игре. Сесија се наставља 15-минутним задатком о томе како инстанце класе Tower треба да снимају инстанце класе Bullet. Ова дискусија покрива логику и услове за иницирање метака са кула. Након тога, још 15 минута посвећено је дискусији о томе како инстанце класе Tower треба да комуницирају са релевантним објектима користећи поруке приликом снимања. Овај сегмент наглашава важност комуникације објеката и покретања догађаја у развоју игре. Ученици затим проводе 30 минута имплементирајући механизам снимања примерака класе Tower. Овај задатак захтева од ученика да интегришу логику пуцања са интеракцијама објеката, обезбеђујући да се куле ефикасно сарађују са непријатељима или другим елементима игре. Сесија се завршава 20-минутним задатком који се фокусира на управљање и постављање торњева унутар арене. Овај задатак истражује постављање, интеракцију и стратешко позиционирање торњева ради оптимизације динамике играња и ефикасног изазивања играча. На крају сесије, студенти ће стећи практично искуство у дизајнирању и имплементацији стратешких интеракција између кула и метака у Греенфооту. Они ће стећи вештине објектно оријентисаног програмирања, руковања догађајима и стратешког дизајна игара, припремајући их за креирање занимљивих и динамичних игара.	
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе.	
Дељење решења	Да би се њихови резултати дистрибуирали наставницима и колегама студентима, користиће се уобичајено подешавање Github/Gitlab и система за управљање учењем (нпр. Моодле). Ученици могу да наставе дискусију о овој теми на форуму који им је обезбеђен преко алата за управљање учењем.	

6.2.3.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. **Задатак 5.9 –Креирањекласа Bullet и Tower (10 минута)**

   Циљ: Наставник води ученике кроз креирањеBullet и Tower класа.

   Концепти за дискусију: Креирање класе, улоге разреда и почетно подешавање.

   Активност: Лекција почиње прегледом основних концепата који се односе на креирање објеката, кретање и интеракцију унутар Greenfoot окружења. Наставник укључује ученике у дискусију како би разјаснио улоге различитих класа и њихове интеракције у игри.

   Ученици креирају два новекласе у Greenfootокружењу, разумејући сврху сваке у контексту игре. Они уче како да поставе ове класе, припремајући се за сложеније интеракције у каснијим лекцијама.

   Ученици почињу креирањем класе Bullet. Ова класа ће представљати пројектиле које испаљују куле. Отварају Greenfoot, бирају "New Class" из менија и именујте нову класу Bullet.

   Затим ученици креирају класу Tower. Ова класа ће представљати куле које пуцају на непријатеље. Они поново бирају "New Class" из менија и именујте нову класу Tower.

   Током целе сесије, наставник објашњава улоге метка и куле у игри. Класа Bullet представља пројектиле које ће куле испаљивати, док класа Tower представља стационарне објекте који могу испаљивати метке на непријатеље. Наставник се стара да ученици разумеју различите улоге које свака класа игра и како ће бити у интеракцији у игри.

   До краја ове активности, ученици би требало да креирају и поставе основне структуре за класе Bullet и Tower, постављајући основу за детаљнију примену у будућим класама.

   (**Commit***:* ece4df70042c8f60098e14ad2cee55514897d825)

2. **Задатак 5.10- Дискусија о томе како би се инстанца класе Bullet требала да помери и шта треба да се деси када стигне до инстанце класе Enemy или ивице арене (10 минута)**

   Циљ: Наставник води дискусију о очекиваном понашању метка док се креће у игри.

   Концепти за дискусију: логика кретања, детекција судара**.**

   Активност: Ученици размишљају и дискутују о логици кретања метка и руковању сударом.

   Ученици почињу тако што разговарају о томе како инстанца Bullet треба да се креће у игри. Сагласни су да се метак креће праволинијски у правцу у коме је испаљен, без промене правца. Брзина метка треба да буде управљива и конзистентна. Да би ово применили, они могу да користе Greenfoot-ove уграђене методе кретања. Важно је истаћи улогу конструктора за иницијализацију вредности атрибута приликом креирања инстанци објекта.

   Наставник затим води дискусију о томе шта треба да се деси када метак стигне до ивице или се судари са непријатељем. Ученици предлажу да када метак стигне до ивице , треба га уклонити из игре. Такође разговарају о томе да приликом судара са непријатељем, метак треба да нестане, а да непријатељ претрпи штету или да буде уништен.

   Ученици предлажу следећи код за логику кретања метка и судара: прво померите метак напред константном брзином, а затим проверите да ли је метак стигао до ивице. Ако је тачно, уклоните метак са арене. Ако није, проверите да ли се метак сударио са непријатељем. Ако дође до судара, то значи да је метак погодио непријатеља, а ми треба да уклонимо метак и нанесемо штету непријатељу или уклонимо непријатеља.

   Наставник демонстрира како да примени ову логику користећи Greenfoot методе move(int), isAtEdge(), и getOneIntersectingObject(Class cls).

Наставник подстиче ученике да усавршe своје идеје и размисле о додатним детаљима, као што је подешавање брзине на основу тежине игре или додавање визуелних ефеката када метак погоди непријатеља. Ова активност помаже ученицима да разумеју принципе кретања и детекције судара у развоју игре, припремајући их за даљу имплементацију у својим пројектима.

3. **Задатак 5.11–Имплементирање кретања инстанце класе Bullet (30 минута)**

   Циљ: Наставник помаже ученицима да имплементирају логику покрета класе Bullet.

   Концепти за дискусију: Покрет глумаца, границе света.

   Активност: Ученици пишу код за померање метка напред и руковање његовим уклањањем када стигне до ивице арене. Ученици почињу прегледом концепата кретања актера и граница, фокусирајући се на то како се ови концепти примењују на класу Bullet. Наставник подсећа ученике на претходне задатке, наглашавајући знања која треба да примене. Ученици се сећају како да додају код методи act() за руковање интеракцијама на ивици света. Такође, сећају се како да рукују променама смера када глумац уђе у одређене ћелије. Штавише, они се сећају како да користе бројаче и механизме одлагања у методи act() за контролисано кретање.

   Имајући на уму ове концепте, ученици настављају да имплементирају логику покрета класе Bullet. Почињу додавањем move(int) методе у act()метод класе Bullet за непрекидно померање метка напред.

   Commit*:* d372827a831381b2254f838041fa4d9a42e53b82

4. **Задатак 5.12 – Дискусија о томе како ће инстанца класе Tower пуцати у инстанцу класе Bullet (15 минута)**

   Циљ: Наставник дискутује са ученицима о логици гађања мецима са куле.

   Концепти за дискусију: Креирање објеката, позивање метода.

   Активност: Ученици размишљају о томе како ће кула створити и лансирати метке. Ученици почињу дискусијом о општој логици потребној да кула испаљује метке у игри. Они се фокусирају на кључне концепте као што су креирање Bullet објеката и позивање метода за њихово покретање. Наставник наглашава да кула не би требало да испаљује метке при сваком позиву act() методе, слично као што је кретање непријатеља руковано механизмом одлагања.

   Ученици размишљају о корацима потребним да бикула повремено испалила метке. Наставник их наводи да размисле о коришћењу механизма за одлагање снимања. Требало би објаснити која би улога конструктора могла бити имплементација овог механизма.

   Наставник објашњава да ће морати да уведу нови атрибуте, shootDelay, counter, nextShootCounter, у Tower класи. Ови атрибути ће контролисати фреквенцију снимања.

   Наставник разлаже релевантне кораке и методе за Tower класу. Прво дефинишеshootDelay атрибут и иницијализује nextShootCounter на 0 у Tower класи. Након тога, метод act() класе Tower треба модификовати да би управљао логиком снимања. Метода треба да створи и испали метак само када nextShootCounter досеже 0. После пуцања, nextShootCounter треба ресетовати на вредност одshootDelay. Ако nextShootCounter није 0, требало би да се смањи за 1. На крају, треба дефинисати посебан fire() метод за управљање креирањем и лансирањем метака. Овај метод ће инстанцирати Bullet објекат и додати га свету.

   Кроз овај процес, ученици разумеју како да имплементирају логику снимања раздвајањем релевантних корака у методе класе Tower. Наставник се стара да ученици схвате важност позивања метода и креирања објеката, појачавајући њихово разумевање ових концепата у контексту њихове игре.

5. **Задатак 5.13 – Дискусија о томе како инстанца класе Tower треба да комуницира са релевантним објектима користећи поруке приликом имплиментације (15 минута)**

   Циљ: Наставник објашњава како кула комуницира са мецима и другим објектима користећи поруке.

   Концепти за дискусију: Преношење порука, позиви метода.

   Активност: Ученици разговарају о механизму за преношење порука за испаљивање метака.

   Сесија почиње тако што наставник објашњава концепт преношења порука и његов значај у објектно оријентисаном програмирању. Наставник наглашава како објекти у игри међусобно комуницирају користећи методе које служе као поруке.

   Да би то илустровао, наставник користи УМЛ дијаграм секвенце да би описао интеракције између објеката Tower, Bullet и Arena. Дијаграм визуелно представља ток порука и позива метода, помажући ученицима да разумеју редослед интеракција.

   Ученици разговарају о детаљном процесу о томе како класа Tower треба да комуницира са мецима и другим објектима када пуца. Наставник објашњава да када Tower одлучи да пуца, шаље поруку (позив методе) да креира инстанцу метка и дода је у Arena. Ова интеракција се покреће унутар act() методе класе Tower. Наставник демонстрира коришћењем УМЛ дијаграма секвенце како Tower шаље поруку Greenfoot оквиру за додавање новог Bullet објекта у свет. Tower затим шаље поруку инстанци Bullet, постављајући њен смер да одговара тренутној ротацији Tower. Ово осигурава да се метак креће у предвиђеном правцу. Када се Bullet креира и позиционира, он ће ступити у интеракцију са другим објектима у игри, као што су непријатељи или ивице света. Наставник објашњава како се овим интеракцијама рукује методом Bullet's act() , што може укључивати проверу судара и уклањање метка када је потребно.

   Током читаве сесије, наставник наглашава сарадничку природу ових интеракција, показујући како се алгоритам шири међу објектима који сарађују. Ово помаже ученицима да цене модуларни дизајн и јасне комуникационе путеве унутар њихове игре.

6. **Задатак 5.14 – Имплементација инстанце класе Tower (30 минута)**

   Циљ: Наставник води ученике кроз имплементацију механизма гађања класе Tower.

   Концепти за дискусију: Креирање објеката, позиционирање актера.

   Активност: Ученици пишу код како би омогућили кули да испаљује метке.

   Сесија почиње тако што наставник објашњава општи циљ: применити механизам за гађање за класу Toraw. Наставник затим разлаже задатак на кораке којима се може управљати и води ученике кроз сваки од њих.

   Прво, ученици припремају неопходне атрибуте и конструкторе класе Tower. Наставник објашњава да је Tower потребан атрибут да би пратио када може да пуца. Затим ученици креирају две методе: booleanTower.canShoot() и void Tower.fire().У почетку, ове методе могу да врате false и не раде ништа, респективно, тако да се могу користити у act() методи.Theact()метод се затим ажурира да користи ове методе. Наставник објашњава да canShoot()метода треба да врати true ако shootCounter досеже 0. Fire()метода је имплементирана да креира инстанцу Bullet и да је правилно позиционира.

   Наставник се стара да ученици разумеју сваки део кода, наглашавајући стварање објекта Bullet, његово позиционирање у свету и усклађивање његове ротације са торњем.

   Ученици затим тестирају своје решење тако што покрећу игру, постављају инстанцу Tower и проверавају да ли испаљује метке у одговарајућим интервалима. Наставник подстиче ученике да решавају све проблеме, обезбеђујући да се меци креирају и крећу како се очекује.

До краја активности, ученици су имплементирали функционални механизам класе Tower, побољшали њихово разумевање стварања објеката, позиционирања актераи позивања метода у Greenfoot-у.

Commit*:* 62aec085954beacf996865a55bed312a09c675f2

7. **Задатак 5.15 - Towers у Arena (20 минута)**

Циљ: Наставник помаже ученицима да интегришу куле и метке у игру, стварајући функционалну арену.

Концепти за дискусију: Интеграција игара, тестирање.

Активност: Ученици постављају куле у арену и тестирају њихову интеракцију са мецима и непријатељима. Наставник почиње објашњавањем циља: интегрисати куле у арену и осигурати да правилно реагују са мецима и непријатељима. Сесија ће укључивати постављање кула у арену и тестирање њиховог понашања у окружењу игре.

Ученици почињу постављањем примерака класе Tower на различите позиције у арени. Наставник објашњава како да додате куле преко Greenfoot интерфејса, осигуравајући да је сваки торањ правилно позициониран.

Затим наставник уводи концепт преоптерећења конструктора. Ово је посебно корисно за иницијализацију Tower објеката са различитим ротацијама. Наставник затим води ученике кроз ажурирање класе Tower да би укључио преоптерећени конструктор који прихвата целобројни параметар за ротацију. Ово омогућава већу контролу над постављањем и оријентацијом кула унутар арене.

Commit*:* bfb6a271f490c341c760e654b3f86a87111c54cb

6.2.4. СЦЕНАРИО: *Greenfoot* објекти у задацима: *Bullets*, *Enemies* и динамика игре

Табела 17. Greenfoot објекти у задацима: Bullets, Enemies и динамика игре

Назив	Greenfoot објекти у задацима: Bullets, Enemies и динамика игре	
Образовни циљеви и исходи учења	До краја сценарија, ученици ће развити вештину у дизајнирању и имплементацији динамике интерактивне игре користећи класе Bullet и Enemy у оквиру Freenfoot окружења. Они ће разумети како да олакшају интеракцију објеката кроз преношење порука, омогућавајући ефикасну комуникацију између елемената игре. Студенти ће демонстрирати способност да имплементирају прецизне механизме детекције судара и одговора, посебно о томе како инстанце класе Bullet ступају у интеракцију са инстанцама класе Enemy. Они ће стећи практичан увид у основне Greenfoot методе као нпр Greenfoot.showText(String, int, int), Greenfoot.getRandomNumber(int), и World.act(),користећи их за побољшање презентације игре, увођење случајности и управљање ажурирањима стања игре. Штавише, ученици ће савладати имплементацију механике покретања непријатеља и услова на крају игре, обезбеђујући динамично искуство играња. Кроз ревизију асоцијација објеката, ученици ће учврстити своје разумевање о томе како објекти сарађују да би створили привлачну динамику игре, припремајући их да примене ове вештине у будућим пројектима развоја игара.	
Циљна група	Ученици средњих школа похађају курс OOP4Fun. Основно знање о програмирању укључујући концепте итерације и селекције. Ученике треба упознати са Greenfoot.	
Временски план	1) Задатак 5.16 – Дискусија о томе како инстанца класе Bullet треба да комуницира са релевантним објектима користећи поруке (15 минута) 2) Задатак 5.17 – Имплементирње инстанце класе Bullet у класиEnemy (30 минута) 3) Објашњење кода за методе Greenfoot.showText(String, int, int), Greenfoot.getRandomNumber(int) and World.act() (15 минута) 4) Задатак 5.18 - Појава непријатеља и крај игре (30 минута) Провера веза (20 минута)	
Материјали и ресурси	Уџбеник из пројекта OOP4Fun. Ресурси из пројекта OOP4Fun. Изворни код пројекта из Github/Gitlab. Интернет ресурси..	
Опис	У овом сценарију учења од 110 минута, средњошколци ће уронити у замршеност динамике игре која укључује метке и непријатеље унутар Greenfoot окружења. Сесија се фокусира на развијање вештина ученика у креирању интерактивне и динамичне игре кроз ефективне интеракције објеката и механику игре. Сесија почиње 15-минутном дискусијом о томе како инстанце класе Bullet треба да комуницирају са релевантним објектима игре користећи поруке. Ова дискусија	

	поставља основу за разумевање начина на који објекти комуницирају и сарађују да би постигли специфично понашање у игри. Након тога, ученици ће посветити 30 минута имплементацији функционалности где инстанце класе Bullet успешно погађају инстанце класе Enemy. Овај задатак изазива ученике да примене своје разумевање детекције судара објеката и руковања догађајима како би створили утицајне интеракције унутар игре. Следећи сегмент укључује 15-минутно објашњење суштинских Greenfoot метода:Greenfoot.showText(String,int, int), Greenfoot.getRandomNumber(int), and World.act().Ученици ће стећи увид у то како ове методе доприносе приказивању текста, генерисању насумичних бројева за механику игре и управљању циклусом ажурирања света игре. Ученици затим троше 30 минута на задатке који се односе на појаву непријатеља и услове на крају игре. Ово укључује дизајнирање и имплементацију механизама за стварање непријатеља у одговарајућим интервалима и одређивање услова за завршетак игре на основу радњи играча или циљева игре. Следи 20-минутна сесија ревизије, фокусирана на консолидовање разумевања асоцијација објеката и њихове улоге у примени динамике игре. Ученици ће прегледати и побољшати своје разумевање начина на који објекти интерагују и сарађују уGreenfoot окружењу како би постигли жељене ефекте играња. На крају сценарија, ученици ће се појавити са бољим разумевањем како да креирају интерактивну и занимљиву динамику игре која укључује метке, непријатеље и стратешку механику игре унутар Greenfoot-а. Они ће бити оспособљени практичним вештинама у имплементацији интеракција објеката, управљању стањима игре и побољшању искуства играча кроз принципе структурираног дизајна игре.	
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе.	
Дељење решења	Да би се њихови резултати дистрибуирали наставницима и колегама студентима, користиће се уобичајено подешавање Github/Gitlab и система за управљање учењем (нпр. Моодле). Ученици могу да наставе дискусију о овој теми на форуму који им је обезбеђен преко алата за управљање учењем	

6.2.4.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1) **Задатак 5.16 – Дискусија о томе како инстанца класе Bullet треба да комуницира са релевантним објектима користећи поруке (15 минута)**

   Циљ: Наставник води дискусију о томе како меци треба да ступе у интеракцију са другим објектима, посебно непријатељима, кроз преношење поруке.

   Концепти за дискусију: Преношење порука, откривање колизије и интеракција са објектима.

   Активност: Лекција почиње прегледом интеракција објеката унутар Greenfoot окружења, фокусирајући се на то како инстанце различитих класа комуницирају и утичу једна на другу. Наставник укључује ученике у дискусију како би ојачао ове концепте и њихову практичну примену у развоју игара.

   Ученици размишљају и дискутују о логици интеракције метака и порукама које треба да пошаљу. Наставник почиње објашњавањем важности прослеђивања порука у објектно оријентисаном програмирању. Дискусија ће се фокусирати на то како инстанце класе Bullet комуницирају са другим објектима, као што су непријатељи и арена, посебно када метак погоди непријатеља.

   Ученици се подстичу да размишљају и поделе своје идеје о логици интеракције метка. Они разматрају питања попут: Шта би требало да се деси када метак погоди непријатеља? Како би метак требало да пренесе овај догађај другим објектима? Које поруке треба да се пренесу да би се интеракција правилно одвијала?

   Наставник уводи концепт детекције судара, објашњавајући како игра треба да открије када се метак укрсти са непријатељем. Они такође разговарају о накнадним акцијама, као што је смањење здравља непријатеља или уклањање непријатеља из арене.

   Да би визуелизовао ове интеракције, наставник користи УМЛ дијаграм секвенце. Дијаграм илуструје поруке које се размењују између класа Bullet, Enemy и Arena током интеракције.

2) **Задатак 5.17 – Имплементирње инстанце класе Bullet у класи Enemy (30 минута)**

   Циљ: Наставник води ученике кроз реализацију интеракције метак-непријатељ**.**

   Концепти за дискусију: Откривање колизије, позивање метода и промене стања објекта.

   Активност: Ученици пишу код за руковање сударом између метка и непријатеља, укључујући ефекте судара. Наставник почиње објашњавајући концепте детекције судара и позивања метода у Greenfoot-у, наглашавајући како су они кључни за руковање интеракцијама између објеката игре . Ученици ће бити вођени корак по корак да имплементирају логику судара између метака и непријатеља.

   Ученици ће прво припремити неопходне атрибуте и методе класа Bullet и Enemy. Ученици ће написати код у класи Bullet да открију колизију са непријатељском инстанцом и позвати Enemy.hit(Bullet) методу. Ученици ће затим имплементирати Enemy.hit(Bullet) методу да се носи са ефектима судара, као што је смањење моћинепријатеља или његово уклањање из игре. Ученици ће тестирати њихову примену како би осигурали да интеракција метак-непријатељ функционише како је предвиђено.

   До краја сесије, студенти ће имати функционалан механизам за детекцију судара између метака и непријатеља, са одговарајућим позивима метода и променама стања објекта. Ова вежба појачава њихово разумевање детекције судара, позива метода и практичне примене ових концепата у развоју игара.

   Commit*:* dcfe31bc006b7f3dcd8b8b759cc1be901c32913c

3) **Објашњење кода за методе Greenfoot.showText(String, int, int), Greenfoot.getRandomNumber(int) and World.act() (15 минута)**

Циљ: Наставник објашњава употребу специфичних Greenfoot метода које ће бити корисне у игри.

Концепти за дискусију: Приказ текста, генерисање насумичних бројева и метода акта.

Активност: Ученици уче како да прикажу текст на екрану, генеришу насумичне бројеве и имплементирају логику игре у методу акта.

Наставник почиње увођењем Greenfoot метода Greenfoot.showText(String, int, int), Greenfoot.getRandomNumber(int), и World.act().Разматра се сврха ових метода, наглашавајући њихов значај у развоју игре за приказивање информација, стварање случајности и дефинисање понашања.

Метода Greenfoot.showText(String, int, int) се користи за приказивање текста на екрану на одређеним координатама. Наставник објашњава да је ова метода корисна за приказивање информација о игри као што су резултати, моћили упутства директно на екрану игре.

Метода Greenfoot.getRandomNumber(int) генерише насумични број између 0 (укључиво) и наведене вредности (искључиво). Наставник расправља о томе како се овај метод може користити за увођење случајности у игру, као што је стварање непријатеља на насумичним локацијама или генерисање насумичних образаца кретања.

Метода World.act()се више пута позива од Greenfoot оквира да изврши главну логику игре. Наставник наглашава да је метода акта где су смештене главне радње и логика игре, омогућавајући континуирано ажурирање и интеракције унутар игре.

4) Задатак 5.18 - Појава непријатеља и крај игре (30 минута)

Циљ: Наставник помаже ученицима да спроведу стварање непријатеља и услове за крај игре.

Концепти за дискусију: Појава објеката, петља игре и услови на крају игре**.**

Активност: Ученици пишу код за периодично стварање непријатеља и дефинишу услове који покрећу крај игре. Наставник почиње објашњавањем важности стварања непријатеља у редовним интервалима и дефинисањем услова на крају игре када су сви непријатељи поражени. Разматрају се концепти покретања објеката, петље игре и услови на крају игре, дајући ученицима јасно разумевање шта треба да се примени.

Arena.act()метода ће се користити за периодично позивање процеса стварањанепријатеља.Требало би увести механизам одлагања за контролу интервала између непријатеља. Креирати spawn()методу у класи Arena за руковање стварним процесом стварања непријатеља. Овај метод ће креирати инстанцу класе Enemy, доделити својства непријатељу (нпр. положај, атрибуте), додати непријатељску инстанцу у арену.

Ученици ће дефинисати и имплементирати услове за крај игре. Када број непријатеља падне на нулу, игра је готова и играч је победио. Да би то урадили, ученици треба да одржавају атрибут у класи Arena да би пратили број створених непријатеља. Повећајте овај атрибут сваки пут када се непријатељ појави и смањите га када је непријатељ убијен. Методу Arena.kill(Enemy) треба прилагодити да провери да ли су сви непријатељи поражени. Ако су сви непријатељи убијени, зауставите игру Greenfoot.stop()и приказати поруку победе. Стога, позивање Greenfoot.stop()треба да буде последња команда у методи.

На крају тестирајте начине повећања броја непријатеља посматрајући њихово периодично креирање у арени. Уверите се да кашњење између креирања непријатељафункционише исправно. Проверите стање на крају игре тако што ћете симулирати пораз свих непријатеља и проверити да ли се игра зауставља са приказаном поруком победе.

До краја ове сесије, студенти ће имплементирати функционалан систем стварања непријатеља и дефинисати јасне услове за крај игре, ојачавајући њихово разумевање петљи игре, управљање објектима и исходе игре засноване на условима.

Commit*:* d48341a095561500af6032d5c8f56e201060f9a4

5) Провера веза (20 минута)

Циљ: Наставник разматра концепт асоцијација између класа, наглашавајући како објекти интерагују и комуницирајуу Greenfoot.

Концепти за дискусију: Асоцијације, комуникација објеката и преношење порука**.**

Активност: Ученици дискутују и ревидирају своје разумевање асоцијација, цртајући везе између различитих објеката и њихових интеракција**.**

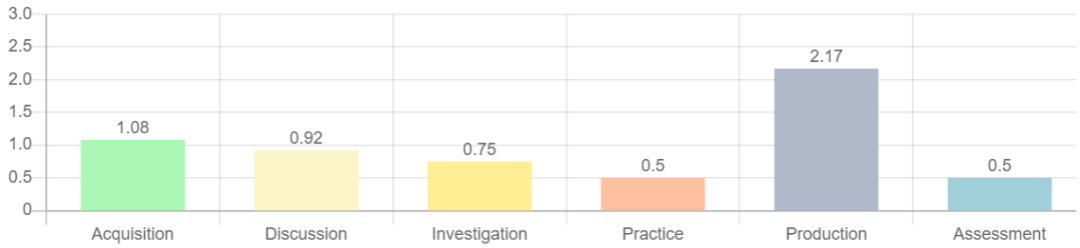
Наставник почиње поновним разматрањем кључних појмова везаних за асоцијације између класа. Ово укључује начин на који објекти комуницирају, комуницирају и преносе поруке једни другима. Да би то олакшао, наставник може да се ослања на примере и задатке обрађене у претходним лекцијама, помажући ученицима да консолидују своје знање и разумеју како се ови концепти примењују у Greenfoot окружењу.

Користите УМЛ дијаграме секвенце да визуелно представите интеракције између различитих објеката. На пример, покажите редослед порука када метак погоди непријатеља и како се Arena носи са стварањем и уклањањем непријатеља.

До краја ове сесије, ученици ће имати појачано разумевање асоцијација и интеракција објеката унутар Greenfoot окружења. Они ће моћи јасно да разумеју како различити објекти у њиховој игри комуницирају и сарађују, примењујући ове концепте на сопствене пројекте развоја игара.

7. НАСЛЕЂИВАЊЕ

7.1. Структура наставних активности и задаци



Слика 11. Расподела ангажовања ученика према типу учења за тематску целину „Наслеђивање“

7.2. Наставни сценарији

Припремљена су четири наставна сценарија за тематску целину „Наслеђивање“.

7.2.1. СЦЕНАРИО: **Увод у наслеђивање**

Табела 18. Увод у наслеђивање

Назив	Увод у наслеђивање
Образовни циљеви и исходи учења	До краја ове области ученици ће бити у стању да разумеју концепте наслеђивања. Разумевање ових концепата ће бити дискутовано у контексту развоја игара, подстицања креативности, тимског рада и ентузијастичног приступа кодирању помоћу Греенфоот алата.
Циљна група	Ученици средњих школа који похађају курс ООП4Фун. Основно знање о програмирању и основно знање о објектно оријентисаном програмирању. Ученике треба упознати са Гринфут-ом.
Временски план	1. Основни појмови о наслеђивању (15 минута) 2. Хијерархија класа и наслеђе (15 минута) 3. Задатак 6.1 и 6.2: Идентификација заједничких својстава (15 минута) и идентификација класе предака (15 минута) 4. Увод у апстрактнекласе (5 минута) 5. Задатак 6.3: Дефиниција апстрактнекласе у игри (10 минута)
Материјали и ресурси	Уџбеник из пројекта ООП4Фун. Ресурси из пројекта ООП4Фун. Изворни код пројекта са Гитхуб/Гитлаб. Интернет ресурси.
Опис	У овом 75-минутном сценарију учења, ученици средњих школа се упознају са принципима објектно оријентисаног програмирања који се односе на наслеђивање кроз развој игара применом Гринфут алата. Сесија почиње 15-минутним уводом од стране наставника о основним концептима наслеђивања, успостављајући контекст у вези са претходним сесијама и будућим развојем игре. Након тога следи 15-минутни сценарио, током којег ученици и наставници дискутују о хијерархији класа у оквиру своје игре. Да би се објаснили концепти у вези са наслеђивањем, посматрају се класе Orb и Direction. Током следећег 15-минутног задатка, истражује се идентификација заједничких својстава за ове класе. Примећено је да ове класе не делују током свог живота; они једноставно реагују на поруке. Сходно томе, идентификован је уобичајени метод деловања, метод act(). Овај метод ће бити дефинисан у обе класе са празним телом. На основу идентификованих заједничких својстава, у наредних 15 минута имплементиран је нови метод класе PassiveActorcontaining*act()*.У следећем сценарију од 5 минута уводе се апстрактнекласе. Апстрактне класе служе као нацрти за друге класе и не могу се инстанцирати. Међутим, они су неопходни у дизајнирању хијерархије класа.

	С обзиром да је класа PassiveActor нацрт за акцију, она је дефинисана као апстрактна класа у наредних 10-минутном сценарију. Поред тога, PassiveActor је успостављен као предак класа Orb и Direction, чинећи Orb и Direction својим потомцима. Пошто је метода act() већ дефинисана у класи PassiveActor, она је уклоњена из класа Orb и Direction. Као резултат тога, до краја сесије, ученици се упознају са новим концептима везаним за наслеђивање.
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе. С обзиром на значај концепта наслеђивања, структура пројекта отвара могућности за даљу дискусију и модификацију. У овом контексту може се размотрити више класа и њихова хијерархија, а могу се увести и додатне класе, методе и атрибути. С друге стране, наставник може прилагодити ову тему да покаже предности наслеђивања и са њим повезане универзалности само на овде предложеним хијерархијама.
Дељење решења	Да би се њихови резултати дистрибуирали наставницима и колегама студентима, користиће се уобичајено подешавање Гитхуб/Гитлаб-а и система за управљање учењем (нпр. Моодле). Ученици могу да наставе дискусију о овој теми на форуму који им је обезбеђен преко алата за управљање учењем.

7.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. **Основни концепти наслеђивања**

   Циљ: Успостављање контекста везаног за претходне сесије, увођење и објашњење концепта наслеђивања кроз примере из стварног живота, и дискусија о његовим предностима.

   Концепти за расправу: Наслеђивање, примери наслеђа из стварног света.

   Активности:У уводном делу се успоставља контекст везан за претходне сесије. Наставник уводи појам наслеђивања. Наставник треба да учини овај концепт ближим ученицима користећи примере из стварног живота (нпр. ако се узме у обзир однос родитељ-дете, деца наслеђују карактеристике од својих родитеља, као што су тип косе, боја очију, итд.). Треба разговарати о предностима наслеђивања. Ови концепти се разматрају у контексту Гринфут окружења и програмског језика Јава.

2. **Хијерархија класа и наслеђивање**

   Циљ: Увођење хијерархије класа у контексту наслеђивања објашњавањем класа предака и потомака, дискусијом о примерима из стварног живота и наслеђивању својстава, наглашавајући предности хијерархије класа.

   Концепти за дискусију: Наслеђивање, хијерархија класа, примери хијерархије класа из стварног света, предности наслеђивања и хијерархије класа.

   Активности:Наставник уводи хијерархију класа у контекст концепта наслеђивања. Наставник уводи класу предака (познату и као: супер класа, класа родитеља) и класе потомака (познате и као: подкласе, класе деце):

   - О претходно обрађеним класама из реалног живота може се дискутовати овде,
   - У овом контексту, треба разговарати о томе да подкласе могу наследити својства (тј. атрибуте и методе) од родитељске класе,
   - Такође, треба разговарати о томе да подкласе могу укључити додатна својства која нису доступна у родитељској класи.

Требало би разговарати о предностима хијерархије класа у контексту концепта наслеђивања. Треба објаснити да у програмском језику Јава свака класа може имати више подкласа, али само једну родитељску класу. Може се дискутовати о улози класе Objeect у контексту хијерархије класа и наслеђивања.

3. **Задатак 6.1 and 6.2 : Идентификација заједничких својстава и идентификација класе предака**

   Циљ: Идентификовање заједничких својстава у класама игре, проналажење класе претка и имплементација нове класе у хијерархији класа.

   Концепти за дискусију: Наслеђивање, хијерархија класа, имплементација наслеђа и хијерархије класа у развоју игре.

   Активности:У контексту развоја игре, разматрају се класе Orb и Direction. Треба приметити да ове класе реагују на поруке. Због тога треба идентификовати заједнички метод за деловање, метод act(). На основу идентификованих заједничких својстава, треба имплементирати нову класу PassivActor која садржи метод act():

   - Ове класе (PassiveActor, Orb, and Direction) треба да послуже за представљање хијерархије класа у концепту наслеђивања,
   - Наставник може визуелно да представи хијерархију разреда користећи хијерархијски дијаграм,
   - Наставник упозорава ученике шта се променило у окружењу Гринфут када је класа Actor замењенаса PassivActor.

Commit*:* afe617814c07a5d885ed06479bf71deda8725f19

4. **Увод у апстрактне класе**

Циљ: Увођење концепта апстрактних класа, разматрање њихове улоге као нацрта у дизајнирању хијерархија класа и истраживање примера из стварног света да би се илустровала њихова примена и специјализација у подкласе.

Концепти за дискусију: Наслеђивање, Хијерархија класа, Апстрактне класе, Примери апстрактних класа из стварног света у контексту наслеђивања и хијерархије класа.

Активности:Појам апстрактне класе уводи наставник. Требало би разговарати о томе да апстрактне класе служе као нацрти за друге класе и да се не могу инстанцирати. Међутим, оне су неопходне у дизајнирању хијерархије класа. Наставник и ученици могу да дискутују о примерима из стварног света који се односе на апстрактне класе и потклассе (нпр., класа Рачунар са основним својствима може се дефинисати као апстрактна класа и може се специјализовати за конзолу, сто, лаптоп и мобилни телефон, сваки са одређеним скупом својстава итд.). Други пример могу бити геометријске фигуре. Правоугаоник или троугао могу бити наслеђени од фигуре апстрактне класе. Приликом израчунавања опсега и површине опште фигуре немамо тачну формулу. Али имамо тачну формулу за правоугаоник и троугао. Квадрат се може наследити од правоугаоника. Ученици треба да разговарају о више примера геометријских фигура и тела.

5. **Задатак 6.3: Дефиниција апстрактне класе у игри**

Циљ : Дискусија о улози класе PassiveActor и имплементација класе као апстрактне.

Концепти за дискусију: Наслеђивање, Хијерархија класа, Апстрактне класе, Имплементација апстрактне класе у развоју игре.

Активности: Концепт апстрактне класе се разматра у Гринфут окружењу и програмском језику Јава. У контексту развоја игара, класа PassiveActor је шаблон за акцију. Стога је дефинисана као апстрактна класа и установљен као предак класа Orb и Direction, чинећи Orb и Direction својим потомцима. Пошто је метода act() већ дефинисана у класи PassiveActor, треба је уклонити из класа Orb и Direction.

Commit*:* f7a5702cae29bf21c9c88620d01ef64e4127c21c

7.2.2. СЦЕНАРИО: **Концепти наслеђивања**

Табела 19. Концепти наслеђивања

Назив	Концепти наслеђивања
Образовни циљеви и исходи учења	До краја ове сесије ученици ће разумети додатне концепте наслеђивања. Испитани концепти ће бити разматрани у контексту развоја игара, подстицања креативности, тимског рада и ентузијастичног приступа кодирању у Гринфут окружењу.
Циљна група	Ученици средњих школа похађају курс ООП4Фун. Основно знање о програмирању и основно знање о објектно оријентисаном програмирању. Ученике треба упознати са Гринфут-ом.
Временски план	1) Задатак 6.4.: Идентификација заједничких својстава везаних за кретање (15 минута) 2) Задатак 6.5.: Дефиниција апстрактне класе која се односи на кретање (15 минута) 3) Задатак 6.6.: Идентификација особина специфичних за класу у вези са кретањем (15 минута) 4) Увод у кључну реч супер у контексту наслеђивања (20 минута) 5) Задатак 6.7. : Рефакторинг кода који се односи на кретање (30 минута)
Материјали и ресурси	Уџбеник из пројекта ООП4Фун. Ресурси из пројекта ООП4Фун. Изворни код пројекта са Гитхуб/Гитлаб. Интернет ресурси.
Опис	Током ове 95-минутне сесије учења, ученици средњих школа се упознају са напредним концептима везаним за насљеђивање примјеном алата Греифут. Сесија почиње са 15-минутним сценаријем који истражује заједничке особине везане за кретање ентитета, фокусирајући се на класе Bullet и Enemy. Ове класе делују слично током живота, крећу се на исти начин и након тога реагују на околину. На основу идентификованих заједничких својстава, у наредном 15-минутном задатку имплементирана је нова апстрактна класа MovingActor која садржи *act()* метод. Ова класа је заједнички предак који ће имплементирати метод *act()* да се креће на исти начин и да се подкласе фокусирају на њихову специфичну сврху. Поред тога, MovingActor је успостављен као предак класа Bullet и Enemy, чинећи Bullet и Enemyсвојим потомцима. У наредном 15-минутном сценарију испитују се својства специфична за класу која се односе на кретање ентитета. У том контексту, *истражује се act()* метод одговарајућих класа, као и атрибути moveDelay и nextMoveCounter. Може се приметити да је код *act()* метода одговорна за кретање иста.

	Након тога следи 20-минутни сценарио, током којег је уведена super кључна реч у контексту наслеђивања. У последњем 30-минутном сценарију извршен је рефакторинг кода који се односи на кретање ентитета. Као резултат тога, претходно идентификовани атрибути *moveDelay* и *nextMoveCounter* из подкласа Bullet и Enemyпремештени су у класу предака MovingActor. Поред тога, параметарски конструктор за иницијализацију ових атрибута је дефинисан у класи MovingActor. Овај конструктор са одговарајућим параметрима је позван из подкласа Bullet и Enemy користећи *super* кључну реч. Осим тога, код одговоран за кретање у *act()* методи подкласа Bullet и Enemy је премештен у *act()* метод класе MovingActor, док остатак имплементације остаје непромењен у подкласама. Коначно , родитељска верзија методе *act()* се позива као прва линија методе *act()* у подкласама Bullet и Enemy. Као резултат тога, до краја сесије, студенти се упознају са новим концептима везаним за наслеђивање.
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе. С обзиром на значај концепата наслеђивања, структура пројекта отвара могућности за даљу дискусију и модификацију. У том контексту, може се размотрити више класа и њихова хијерархија, а могу се увести и додатне класе, методе и атрибути. С друге стране, наставник може да прилагоди ову тему да покаже предности наслеђивања и са њом повезану универзалност само на овде предложеним хијерархијама.
Дељење решења	У циљу ширења њихових резултата наставницима и колегама студентима, користиће се уобичајена поставка Гитхуб/Гитлаб и система за управљање учењем (нпр. Моодле). Ученици могу наставити дискусију о теми на форуму који им је достављен путем алата за управљање учењем.

7.2.2.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. **Задатак 6.4.: Идентификација заједничких својстава везаних за кретање**

   Циљ: Испитивање класа Bullet и Enemy, наглашавајући њихово слично понашање током живота, посебно како се крећу и реагују на своју околину.

   Концепти за дискусију: наслеђивање, хијерархија класа, апстрактне класе.

   Активност: Фокус је наBullet и Enemy класе, које делују слично током живота. Треба приметити да се ове класе крећу на исти начин и након тога реагују на околину.

2. **Задатак 6.5.: Дефиниција апстрактне класе која се односи на кретање**

   Циљ : Испитивање класа Bullet и Enemy, наглашавајући њихово слично понашање током живота, посебно како се крећу и реагују на своју околину.

   Концепти за дискусију: Наслеђивање, хијерархија класа, апстрактне класе, имплементација апстрактне класе у развоју игре.

   Активност:На основу идентификованих заједничких својстава, треба имплементирати нову апстрактну класу MovingActor која садржи *act()* методу. Поред тога, MovingActor је успостављен као предак класа Bullet и Enemy, чинећи Bullet и Enemy својим потомцима. Треба разговарати о томе да подкласе наслеђују заједничке особине од родитељске класе. Класа MovingActor је нацрт за дизајн класе и треба је прогласити апстрактном.

   Commit*:* 43e53b533563ce0a860b294ad9009f77409c48d4

3. **Задатак 6.6.: Идентификација особина специфичних за класу која се односи на кретање**

   Циљ: Испитивање класа Bullet и Enemy, наглашавајући њихово слично понашање током живота, посебно како се крећу и реагују на своју околину.

   Концепти за дискусију : наслеђивање, хијерархија класа, апстрактне класе.

   Активност: Испитују се особине специфичне за класу које се односе на кретање ентитета. Истражује се *метода act()* одговарајућих класа, као и атрибути moveDelay и nextMoveCounter. Може се приметити да је код *act()* методе одговорне за кретање иста.

4. **Увод у*super*кључну реч у контексту наслеђивања**

   Циљ: увођење super кључне речи у контексту наслеђивања, демонстрирање њене употребе за позивање својстава из матичне класе и дискусија о њеним предностима.

   Концепти за дискусију: Наслеђивање, хијерархија класа, супер кључна реч у контексту наслеђивања. Позицијакључне речи super.

   Активности: Наставник уводи *super* кључну реч. Уведенаје супер кључна реч у контексту наслеђивања:

   - *super*кључна реч може да се користи како би се позвао конструктор из родитељске класе,
   - *super*кључна реч се може користити како би се позвао метод из родитељске класе,
   - *super*кључна реч може да се користи како би се позвао атрибут из родитељске класе,
   - *supe*r мора бити прва реч.

   Треба размотрири предности коришћењакључне речи *super* keyword у контексту наслеђивања.

5. **Задатак 6.7. : Рефакторинг код који се односи на кретање**

   Циљ: Рефакторинг код који се односи на кретање објеката од подкласа Bullet и Enemy до класе предака MovingActor.

   Концепти за дискусију: Наслеђивање, хијерархија класа, super кључна реч у контексту наслеђивања.

Aktivnosti:Извршен је рефакторинг кода који се односи на кретањеобјеката. Претходно идентификовани атрибути *moveDelay* и *nextMoveCounter* из подкласа Bullet и Enemy су премештени у класу предака MovingActor. Параметарски конструктор за иницијализацију ових атрибута је дефинисан у класи MovingActor. Овај конструктор са одговарајућим параметрима је позван из подкласа Bullet и Enemy користећи *super* кључну реч. Код одговоран за кретање у *act()* методи подкласа Bullet и Enemy је премештен у *act()* метод класе MovingActor, док остатак имплементације остаје непромењен у подкласама. Коначно , надређена верзија методе *act()* се позива као прва линија методе *act()* у подкласама Bullet и Enemy. Треба разговарати о томе да подкласе могу укључити додатне особине које нису доступне у надређеној класи (тј. другачија имплементација *act()* метода).

Commit*:* ca1f010a63445c1847b74259a1c6cd4817121db3

7.2.3. СЦЕНАРИО: **Концепти наслеђивања (Део 2)**

Табела 20. Концепти наслеђивања (Део 2)

Назив	Концепти наслеђивања (Део 2)
Образовни циљеви и исходи учења	До краја ове сесије, ученици ће разумети додатне концепте наслеђивања. Испитани концепти ће бити разматрани у контексту развоја игара, подстицање креативности, тимског рада и ентузијастичног приступа кодирању у Гринфут окружењу.
Циљна група	Ученици средњих школа који похађају курс OOP4Fun. Основно знање програмирања и основно знање објектно оријентисаног програмирања. Студенти би требали бити упознати са Гринфутом.
Временски план	1) Задатак 6.8. : Стварање прилагођених непријатеља (30 минута) 2) Увод у Лисков принцип супституције (20 минута) 3) Задатак 6.9.: Постављање прилагођених непријатеља (20 минута)
Материјали и ресурси	Уџбеник из пројекта OOP4Fun. Ресурси из пројекта OOP4Fun. Изворни код пројекта из Гитхуб / Гитлаб. Интернет ресурса.
Опис	Током ове 70-минутне сесије учења, ученици средњих школа се упознају са напредним концептима наслеђивања помоћу алата Гринфут. Сесија почиње са 30-минутним сценаријем који се фокусира на класу непријатеља, где ученици креирају додатне подкласе које представљају различите непријатеље (нпр. Жаба и Паук). У том контексту, слике и конструктори без параметара (са одговарајућим позивањем родитељског конструктора) су дефинисани за сваку врсту непријатеља. У следећем 20-минутном сценарију, уводи се Лисков принцип супституције (ЛСП). Овај принцип, део СОЛИД принципа објектно оријентисаног дизајна, наводи да функције које користе показиваче или референце на надређене класе треба да буду у стању да користе објекте подкласа. Након тога, 20-минутни задатак је посвећен повраћају прилагођених непријатеља. Метода *Arena.spawn()* је испитана, а прилагођени непријатељи се креирају кроз различите одлуке и чувају у варијабли типа Enemy. Примећено је да ниједан други код у апликацији не треба мењати, што показује примену ЛСП-а. Као резултат тога, до краја сесије, студенти се упознају са напредним концептима наслеђивања и практичним применама.
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе. С обзиром на значај концепата наслеђивања, структура пројекта отвара могућности за даљу дискусију и модификацију. У том контексту, може се размотрити више класа

	и њихова хијерархија, а могу се увести и додатне класе, методе и атрибути. С друге стране, наставник може да прилагоди ову тему да покаже предности наслеђивања и са њом повезану универзалност само на овде предложеним хијерархијама.
Дељење решења	У циљу ширења њихових резултата наставницима и колегама студентима, користиће се уобичајена поставка Гитхуб/Гитлаб и система за управљање учењем (нпр. Моодле). Ученици могу наставити дискусију о теми на форуму који им је достављен путем алата за управљање учењем.

7.2.3.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. **Задатак 6.8. : Стварање прилагођених непријатеља**

   Циљ: Дефинисање подкласа класе Enemy, свака са сликама и конструкторима без параметара који на одговарајући начин позивају родитељски конструктор.

   Концепти за дискусију: Наслеђивање, хијерархија класа, супер кључна реч у контексту наслеђивања.

   Активности:Фокус на класи непријатеља и дефиницији додатних подкласа које представљају различите непријатеље (нпр. Frog и Spider). Слике и конструктори без параметара (са одговарајућим позивањем родитељског конструктора) треба да буду дефинисани за сваки тип непријатеља.

   Commit*:* b0ac1fbe793548a32f7700c292aed631918c8388

1. **Увод у Лисков принцип супституције**

   Циљ: Увођење Лисковог принципа супституције, расправљање о стварном свету и истраживање предности придржавања овог принципа у контексту наслеђивања.

   Концепти за дискусију: Наслеђивање, хијерархија класа, референтне варијабле, Лисков принцип супституције.

   Активности: Уведен је Лисков принцип замене. Овај принцип је део СОЛИД принципа објектно оријентисаног дизајна. Принцип каже да функције које користе показиваче или референце на надређене класе треба да буду у стању да користе објекте подкласа. Треба дискутовати о примерима стварних речи (нпр. ако је класа рачунара дефинисана као надређена класа, а класе конзоле, десктопа, лаптопа и мобилног телефона дефинисане као подкласе, Лисков принцип замене каже да ће функције које користе класу рачунара такође радити са свим подкласама, без икаквих промена у коду). Треба разговарати о предностима коришћења Лисковог принципа супституције у контексту наслеђивања.

2. **Задатак 6.9.: Постављање прилагођених непријатеља**

   Циљ: Демонстрација примене принципа Лисковљеве супституције стварањем прилагођених непријатеља.

   Концепти за дискусију: Наслеђивање, хијерархија класа, референтне варијабле, Лисков принцип супституције.

   Активности: Задатак је посвећен постављању прилагођених непријатеља. Метода *Arena.spawn()* је обрађена, а прилагођени непријатељи се креирају кроз различите одлуке и чувају у варијабли типа Enemy. Треба напоменути да ниједан други код у апликацији не треба мењати, што показује примену Лисковљевог принципа супституције.

   Commit*:* 8cd4397f585ec957bbc18ca98e01823f434a13a6

7.2.4. СЦЕНАРИО: **Концепти наслеђивања (Део 3)**

Табела 21. Концепти наслеђивања (Део 3)

Назив	**Концепти наслеђивања (Део 3)**
Образовни циљеви и исходи учења	До краја ове сесије, ученици ће разумети додатне концепте наслеђивања. Испитани концепти ће бити разматрани у контексту развоја игара, подстицање креативности, тимског рада и ентузијастичног приступа кодирању у Гринфут окружењу.
Циљна група	Ученици средњих школа који похађају курс OOP4Fun. Основно знање програмирања и основно знање објектно оријентисаног програмирања. Студенти би требали бити упознати са Гринфутом.
Временски план	1. Задатак 6.10.:Дискусија о хијерархијикласе Arena(20 минута) 2. Задатак 6.11 и 6.12.: Направите универзалну класу Arena (30 минута) и креиратиDemoArena (15 минута) 3. Задатак 6.13.: Креирати прилагођенекласе Arena (30 минута) 4. Ревизија теорије наслеђивања (20 минута)
Материјали и ресурси	Уџбеник из пројекта OOP4Fun. Ресурси из пројекта OOP4Fun. Изворни код пројекта из Гитхуб / Гитлаб. Интернет ресурса.
Опис	Током ове 115-минутне сесије учења, ученици средњих школа се упознају са напредним концептима наслеђивања помоћу алата Гринфут. Сесија почиње 20-минутном дискусијом о хијерархији класе Arena. Подкласе Arena класе су одговорне за прилагођене распореде (нпр. позиције Orb и Direction инстанци, величина арене). Ови задаци се обављају у конструкторима подкласа, који постављају и чувају положаје мријеста, ротације и димензије арене. У наредном 30-минутном задатку, уводи се универзална класа Arena. Додатни атрибути (*spawnPositionX*, *spawnPositionY* и *spawnRotation*) су дефинисани, иницијализовани у конструктору и коришћени у методама *spawn()* и *respawn(Enemy).* Атрибути који се односе на димензије арене (*width* и height) су такође дефинисани и иницијализовани у конструктору. Како класа Arena служи као нацрт за дефинисање конкретних арена, она је дефинисана као апстрактна класа. На основу идентификоване класе Arena, следећи 15-минутни задатак уводи подкласу DemoArena. DemoArena конструктор је дефинисан, позивајући се на конструктор родитељске класе, а код одговоран за распоред праваца, кугли и кула се помера из Arena конструктора у DemoArena конструктор. Коначно , креирана је нова инстанца класе DemoArena. У следећем 30-минутном сценарију креиране су и друге иновативне подкласе класе Arena. Код се може делити са другим ученицима у групи.

	Коначно , последњи 20-минутни сценарио покрива теорију у вези са наслеђивањем. Као резултат тога, до краја сесије, студенти се упознају са напредним концептима наслеђивања и практичним применама.
Вредновање	Гамификација представља неформалну процену, али ће повећати интересовање, унутрашњу мотивацију и резултате учења целе групе. С обзиром на значај концепата наслеђивања, структура пројекта отвара могућности за даљу дискусију и модификацију. У том контексту, може се размотрити више класа и њихова хијерархија, а могу се увести и додатне класе, методе и атрибути. С друге стране, наставник може да прилагоди ову тему да покаже предности наслеђивања и са њом повезану универзалност само на овде предложеним хијерархијама.
Дељење решења	У циљу ширења њихових резултата наставницима и колегама студентима, користиће се уобичајена поставка Гитхуб/Гитлаб и система за управљање учењем (нпр. Моодле). Ученици могу наставити дискусију о теми на форуму који им је достављен путем алата за управљање учењем.

7.2.4.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. **Задатак 6.10.: Дискусија о хијерархији класе Arena**

   Циљ: Истраживање хијерархије класе Arena, наглашавање како су подкласе одговорне за дефинисање прилагођених распореда и имплементацију ових распореда у оквиру својих конструктора.

   Концепти за дискусију: Наслеђивање , хијерархија класа, конструктори.

   Активности:Дискусија о хијерархији класа Arena, треба приметити да су подкласе класе Arena одговорне за прилагођене распореде (нпр. позиције Orb и Direction инстанци, величина арене). Ови задаци се обављају у конструкторима подкласа, који постављају и чувају положаје постављања објеката, ротације и димензије арене.

2. **Задатак 6.11. и 6.12.: Креирање универзалних класа Arena и DemoArena**

   Циљ: Увођење универзалне апстрактне класе Arena, дефинисање и иницијализација конкретне подкласе класе Arena, истраживање хијерархије класе Arena.

   Концепти за дискусију: Наслеђивање, хијерархија класа, апстрактне класе, конструктори.

   Активности: На основу претходне дискусије, уведена је универзална класа Arena. Додатни атрибути (*spawnPositionX*,*spawnPositionY* и*spawnRotation*) су дефинисани, иницијализовани у конструктору и коришћени у методама *spawn()*и *respawn(Enemy).* Атрибути који се односе на димензије арене (*width* и height) су такође дефинисани и иницијализовани у конструктору. Како класа Arena служи као нацрт за дефинисање конкретних арена, она је дефинисана као апстрактна класа .

   Commit*:* e9844d7d9b5f19969618b469ebc907d0fe3c1357

   На основу идентификоване класе Arena, дефинисана је подкласа DemoArena. DemoArena конструктор је дефинисан, позивајући се на конструктор родитељске класе, а код одговоран за распоред праваца, кугли и кула се помера из Arena конструктора у DemoArena конструктор. Коначно , креирана је нова инстанца класе DemoArena. Да бисте активирали класу DemoArena, кликните десним тастером миша и изаберите нову класу DemoArena.

   Commit*:* 6a6569774b5735f453a56c7cb2cdbf19d228eae9

3. **Задатак 6.13.:Креирање прилагођенихкласа класе Арена**

   Циљ: Дефинисање и иницијализација customArena подкласа, истраживање хијерархије класе Arena.

   Концепти за дискусију: Наслеђивање, хијерархија класа, апстрактне класе, конструктори.

   Активности:Створене су и друге иновативне подкласе класе Arena. Код се може делити са другим ученицима у групи.

1. **Ревизија теорије наслеђивања**

   Циљ: Разматрање концепта и предности наслеђивања, расправљање о хијерархији класа и апстрактне класе, истраживање супер кључне речи и Лисков принцип супституције заједно са њиховим предностима, и испитивање примера и имплементација наслеђивања у стварном животу и игри.

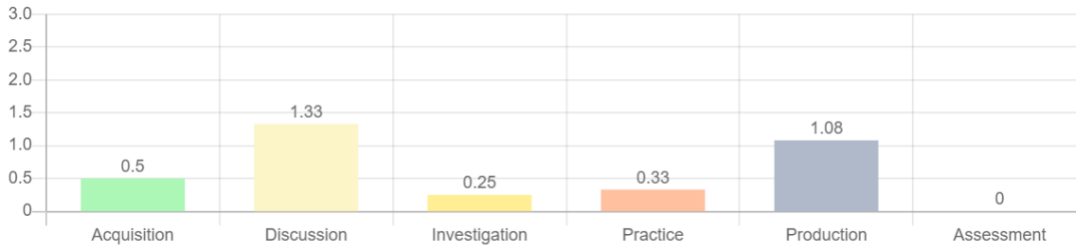
   Концепти за дискусију: Наслеђивање, хијерархија класа, апстрактне класе, супер кључна реч, принцип замене Лисков, примери и имплементације наслеђивања у стварном животу и игри.

   Активности:Разматра се концепт наслеђивања. Предности наслеђивања су сагледане. Разматра се хијерархија класе и њене предности у контексту концепта наслеђивања. Разматра се концепт апстрактне класе. Расправља се о super кључној речи у њеним предностима у

контексту наслеђивања. Лисков Принцип супституције је разматран и користи од коришћења Лисков принцип супституције у контексту наслеђивања су дискутовани. Примери наслеђивања у стварном животу су дискутовани. Расправља се о примерима наслеђивања и имплементацији везаним за игру.

8. ЕНКАПСУЛАЦИЈА И СКРИВАЊЕ ИНФОРМАЦИЈА

8.1. Структура наставних активности и задаци



Слика 12. Расподела ангажовања ученика према типу учења за тематску целину „Енкапсулација и скривање информација“

8.2. Наставни сценарији

Припремљена су два наставна сценарија за тематску целину „Енкапсулација и скривање информација“.

8.2.1. СЦЕНАРИО: **Увод у енкапсулацију**

Табела 22. Увод у енкапсулацију

Назив	Увод у енкапсулацију
Образовни циљеви и исходи учења	Циљ овог сценарија је да ученицима приближи концепт енкапсулације кроз даље унапређење игре `TowerDefense`.
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује и наслеђивање. Потребно је да ученици буду упознати са *Greenfoot* окружењем.
Временски план	Укупно време потребно за реализацију сценарија: **125** минута. 1. Увод (5 минута) 2. Тимски рад и програмирање [Задатак 8.1.] (20 минута) 3. Тимски рад [Задаци 8.2. и 8.3.] (30 минута) 4. Дискусија (35 минута) 5. Тумачење кôда (25 мин)**Error! Reference source not found.** 6. Формирање тимова и пројектни задатак [Задатак 8.4.] (10 минута)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Током ове лекције, ученици ће се упознати са концептом енкапсулације у контексту објектно-оријентисаног програмирања. Лекција почиње концизним уводом од 5 минута, у оквиру којег наставник даје преглед циљева лекције. Ученици крећу са развојем класе `ManualTower`, коју изводе из класе `Tower`. Током ове активности ученици треба да се усредсреде на дефинисање два конструктора, који треба да буду усклађени са конструкторима надкласе, како би се осигурала исправна иницијализација. Потребно је и да ученици имплементирају методу `act()`, која на почетку позива методу `act()` своје надкласе. Након што ученици дефинишу класу, наставник уводи логички атрибут `isManuallyControlled`, који иницијализује са вредношћу `false`. Ученици затим имплементирају методу `changeControl(boolean)` путем које се наизменично мења вредност атрибута `isManuallyControlled` и, у складу са њим, иконица дате куле. Кроз ову активност се демонстрира енкапсулирање тиме што се, путем метода, контролише приступ стању објекта. Сваки ученик би затим требало да ручно

	покрене методу `changeControl()` над инстанцама класе `ManualTower` и обрати пажњу на то како се мењају интерна стања и визуелни прикази кула. Суштина овог сценарија се огледа у имплементацији приватне методе `processUserControl()` чији је задатак да детектује када се мишем кликнуло на инстанцу одређене куле. Када корисник кликне на одређену кулу, метода мења стање куле и ажурира њену оријентацију на основу позиције миша, при чему се користи енкапсулација за сакривање сложеније логика управљања кулом. Ученици треба да имплементирају дату методу и повежу је са методом `act()`, али и да тестирају интеракцију са окружењем игре да би били сигурни у функционалност, утврђујући на тај начин знање о томе како приватне методе омогућавају да се кôд заштити од промена споља.
Вредновање	Активности у оквиру ове лекције ће омогућити наставницима да дају ученицима формативне повратне информације базиране на њиховом учешћу у спроведеним дискусијама, али и на праћењу њиховог рада у окружењу обрнуте учионице (енгл. *flipped classroom*), као и рада у тиму. Међусобно вредновање ће се вршити *online*, као део домаћег задатка. Кроз ову активност ученици ће се подсетити важних аспеката лекције и биће подстакнути да критички процењују рад других ученика. Разматрањем различитих примера, ученици ће лакше усвојити изложено градиво, што доприноси и повећању општег достигнућа постављених образовних циљева и исхода учења целе групе. Исходе учења и стечена знања ће ученици даље применити и током развоја тимског пројекта на којем раде.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

8.2.1.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Увод (5 минута)

   Наставник треба да покрене претходно развијену игру и да са ученицима посматра понашање различитих протагониста. Наставник предлаже да се, у циљу лакшег одстрањивања непријатеља, уведе нова врста куле којом може управљати корисник. Корисник би требало да може да, у једном тренутку управља само једном кулом. Када кликне на кулу, корисник би требало да преузме контролу над њом. Да би се указало на то којом кулом управља корисник, потребно је да кула, којом у датом тренутку управља корисник, има другачији изглед.

2. Тимски рад и програмирање [Задатак 8.1.] (20 минута)

   Циљ: Припрема класе **`ManualTower`**.

   Концепти за дискусију: наслеђивање, класе, конструктори.

   Активности: Будући да ученици већ умеју да дефинишу изведене класе, замолите их да у тимовима дефинишу класу **`ManualTower`** као изведену класу класе **`Tower`**. Ученици треба да имплементирају и конструкторе и методу **`act()`**, и да осигурају да ће конструктори надкласе бити позвани из ових метода. Током овог дела лекције, ученици ће обновити градиво, применити га, и унапредити практично знање о наслеђивању.

   [Commit*:* `63a02fa0c5080165cba8b467da08c4b65f31d0a8`**]**

3. Тимски рад [Задаци 8.2. и 8.3.] (30 минута)

   Циљ: Увиђање потребе за приватним методама кроз дефинисање методе **`changeControl()`**.

   Концепти за дискусију: методе, класе, атрибути, модификатори приступа.

   Активности: Наставник треба да припреми иконице за кулу којом управља корисник. Потребно је објаснити ученицима како да користе методу **`Actor.setImage(String)`**, како би променили иконицу објекта програмским путем. Оставити ученицима пар минута да испробају ову методу.

   Наставник треба да покрене дискусију о томе како се може утврдити да ли кулом управља корисник. Наставник треба да истакне да је, поред тога што је потребно променити стање објекта, једнако важно ажурирати и његову иконицу. Неопходно је нагласити да, уколико корисник жели да промени стање неког **`Tower`** објекта и директно мења искључиво његов атрибут, иконица неће бити промењена. Ова дискусија треба да помогне ученицима да увиде потребу за променом, и вредности атрибута и пратеће иконице, путем методе, али и да разумеју зашто атрибути треба да буду приватни, а не јавни. Објасните ученицима да се овакав приступ назива енкапсулација и сакривање информација, и да он подразумева да интерно стање објекта треба да буде сакривено, а да се јавне методе користе како би се дато стање променило на контролисан начин.

   Пружите ученицима прилику да имплементирају логику саме методе. Замолите их да затим ручно позову своју методу и прате промену интерног стања објекта.

   [Commit*:* `2257746b7dac5eaab7acc55d6493319230338f3a`**]**

4. Дискусија (35 минута)

   Циљ: Разумевање потребе за енкапсулирањем логике унутар засебне методе.

   Концепти за дискусију: методе, гранање.

   Активности: Наставник треба да истакне да се стање куле засада може променити искључиво директним (тј. ручним) позивањем методе **`changeControl()`**. Циљ је да се обезбеди да се ова метода аутоматски позива на клик мишем.

Међутим, потребно је указати на то да се миш може налазити и ван подручја света и да ће тада ће информације о тренутном статусу миша имати вредност `null`. Подсетите ученике и на то да се метода `act()` непрестано извршава током трајања игре и да она заправо треба да провери да ли је корисник кликнуо на кулу и да само у том случају позове методу `changeControl()`. Потребно је нагласити да логика за обраду треба да буде енкапсулирана унутар засебне методе `processUserControl()`.

5. Тумачење кôда (25 мин)

Циљ: Увођење методе неопходне за решавање проблема.

Концепти за дискусију: методе, *Greenfoot* окружење.

Активности: Размотрите на који начин би се могло променити стање инстанце, када се кликне на одређени објекат. Да би овакво понашање могло да се имплементира, потребно је увести и појаснити методу `GreenFoot.mouseClicked(Object)`. Поред тога, потребно је увести и класу `MouseInfo`, чија се инстанца може користити за добијање информација о текућој позицији миша.

6. Формирање тимова и пројектни задатак [Задатак 8.4.] (10 минута)

Циљ: Продубљивање разумевања приватних метода и енкапсулације кроз практичан задатак.

Концепти за дискусију: методе, модификатори приступа, класе.

Активности: Након дефинисања приватне методе `processUserControl()`, замолите ученике да имплементирају њену логику. Када се кликне мишем, потребно је преиначити конторолу. Ако кулом управља корисник, онда она треба да прати миша и да буде усмерена према њему. Подсетите ученике на то да је могуће и да се миш нађе ван подручја света. Након што поделите ученике у тиме, затражите од њих да имплементирају логику методе `processUserControl()`.

Један или два тима би требало да представе своја решења, и да цела група, заједно са наставником, продискутује њихова решења. До краја лекције сви ученици треба да разумеју начин имплементације ове методе.

[Commit*:* `6ec1f489576019a6493490f9e97797920b923869`]

8.2.2. СЦЕНАРИО: **Истраживање енкапсулације кроз развој игара**

Табела 23. Истраживање енкапсулације кроз развој игара

Назив	Истраживање енкапсулације кроз развој игара
Образовни циљеви и исходи учења	
Циљна група	Ученици средњих школа који учествују у *OOP4Fun* курсу. Предуслов је поседовање основног знања о програмирању које укључује и наслеђивање. Потребно је да ученици буду упознати са *Greenfoot* окружењем.
Временски план	1) Изокренута учионица - сесија (30 минута): Ученици треба да идентификују проблем са претходно имплементираном контролом корисника и да истраже како да реше тај проблем. 2) Атрибути класе (5 минута) 3) Задатак 7.6.: Дајте ученицима пример о контролисаном торњу (кули) од стране корисника тако што ћете визуелно и функционално означити тренутно контролисани торањ у игри. На пример, можете променити његову боју, икону или додати визуелни ефекат контролом корисника. (5 минута) 4) Метод класе (10 минута) 5) Задатак 7.7.: Измена кориснички управљаног торња (куле) којим се управља са централног места (20 мин) 6) Задатак 7.8.: Позивање промена торња (куле) којим се управља ос стране корисника (15 мин) 7) Ревизија теорије (10 мин)
Материјали и ресурси	Приручник *OOP4Fun* пројекта. Ресурси *OOP4Fun* пројекта. Изворни кôд пројекта доступан на *Github/Gitlab* платформи. Ресурси на Интернету.
Опис	Сесија почиње дискусијом међу ученицима који проналазе проблеме везане за контролу корисника, као што је немогућност поништавања избора торња(куле) након што је изабран. Ученици треба да буду укључени у дискусију и предложе решења за праћење торња (куле)који се тренутно контролише и модификовање класе ManualTower како би се укључио механизам за поништавање изабраногторња (куле). На крају, треба имплементирати методу класе changeControlledInstance(), која омогућава промену контроле над торњевима(кулама) из централног метода, чиме се побољшава разумевање енкапсулације показујући како методе класе могу управљати заједничким стањем између инстанци. Овај свеобухватни образовни приступ подучава концепту енкапсулације и демонстрира њену важност и корисност у апликацијама у стварном свету, развијајући вештине решавања проблема и сарадње међу ученицима.

Вредновање	Активности у оквиру ове лекције ће омогућити наставницима да дају ученицима формативне повратне информације базиране на њиховом учешћу у спроведеним дискусијама, али и на праћењу њиховог рада у окружењу обрнуте учионице (енгл. *flipped classroom*), као и рада у тиму. Међусобно вредновање ће се вршити *online*, као део домаћег задатка. Кроз ову активност ученици ће се подсетити важних аспеката лекције и биће подстакнути да критички процењују рад других ученика. Разматрањем различитих примера, ученици ће лакше усвојити изложено градиво, што доприноси и повећању општег достигнућа постављених образовних циљева и исхода учења целе групе. Исходе учења и стечена знања ће ученици даље применити и током развоја тимског пројекта на којем раде.
Дељење решења	За дељење решења са наставницима и другим ученицима користиће се уобичајена поставка *Github/Gitlab* платформе и образовне платформе (нпр. *Moodle*). Ученици могу да наставе дискусију на дату тему путем форума који ће им бити расположив у оквиру образовне платформе.

8.2.2.1. *СМЕРНИЦЕ ЗА ПРИПРЕМУ НАСТАВЕ*

1. Сесија у „обрнутој“ учионици (30 минута)

   Циљ: Ученици треба да препознају потребу да се атрибут иницијализује на једном месту

   Концепти за дискусију: атрибути класе.

   Активности: На почетку часа дозволите ученицима да идентификују проблем са корисничком контролом. Тренутно није могуће поништити избор куле. Подстакните ученике да размисле о томе како би овај проблем могао да се реши. Објасните да у игри само један кула треба да буде изабран у било ком тренутку. Ова дискусија треба да доведе ученике до идеје да постоји једно место у програму које се иницијализује само једном и коме могу приступити други делови програма, као и други објекти и актери.

2. Атрибути класе

   Циљ: Упознавање са основама атрибута класе

   Концепти за дискусију: атрибути, атрибути класе, класе

   Активност: Објасните шта су атрибути класе: променљиве које припадају самој класи, а не инстанцама те класе. Повежите овај концепт са сценаријем игре о којем смо раније говорили, где би поседовање централизованог атрибута за управљање тренутно изабраним торњем могло да реши проблем.

3. Задатак 7.6.: Додавање доказа о кориснички контролисаном торњу

   Циљ:Практична примена атрибута класе и вредности null

   Концепти за дискусију: атрибути, атрибути класе, класе, модификатори приступа

   Активност: Да бисте пратили који је торањ тренутно изабран у игри, додајте приватни статички атрибут controlledInstance у класу ManualTower и иницијализујте га на null. Статички атрибут се односи на целу класу, а не на објекат те класе. Дефинисање статичке променљиве ће нам омогућити да одредимо да ли је торањ изабран и ако јесте, који је то торањ, референцом на име класе, без потребе да приступамо одређеном објекту. Наставник треба да нагласи да постоји само један controlledInstance за целу игру. На почетку, controlledInstance треба да буде иницијализован на null, јер није изабран ниједан торањ.

   Преглед интерног стања класе. Овде наставник објашњава разлике између статичких и нестатичких атрибута. Наставник са ученицима разговара о предностима коришћења статичких атрибута у игрицама. Наставник такође треба да помене статичке методе и да разговара са ученицима о томе где је коришћење статичких метода корисно.

   [Commit*:* `c4739460bed583d2126de066acc6b1149d022990`**]**

4. Метода класе

   Циљ: Упознавање са основама метода класе

   Концепти за дискусију: методе класе, класе, објекти

   Активност: Представите концепт метода класе које могу да раде на подацима на нивоу класе. Дискутујте о потреби за методама као што је changeControlledInstance, која управља пребацивањем контроле над тренутно изабрани торањ. Нагласите да се ове методе могу позвати без потребе за инстанцом класе. На пример, школско звоно звони свима у исто време, није важно ко сте, с друге стране, провера домаћег задатка ученика захтева информације о том конкретном ученику.

5. Задатак 7.7.: Промена кориснички контролисаног торња са централизованог места

   Циљ: Практична примена статичких метода.

   Концепти за дискусију: методе, статичке методе класе, статички атрибути.

   Активност: Наставник треба да дода методу changeControlledInstance која мења кориснички контролисан торањ. Параметар методе ће бити торањ који корисник жели да изабере. Прво треба проверити да ли је контролисана инстанца тренутно изабрана. Ако јесте, ништа не би требало да се мења, али ако је прослеђена инстанца другачија, треба променити тренутно контролисану инстанцу (референцу на тренутно контролисану инстанцу). Ручно тестирајте функцију и приметићете да се иконе торњева не мењају. Нагласите да само промена референце на контролисану инстанцу неће променити контролу и да то треба уради корисник. Додајте кôд који ослобађа тренутно контролисану инстанцу и након ажурирања референце, додајте кôд који поставља корисничку контролу ново контролисане инстанце. Истакните потребу за проверавањем null референци које се могу појавити ако тренутно нема контролисане инстанце или ако нема нове контролисане инстанце (када је параметар null).

   [Commit*:* `9dc6d8dd4dcbbd71edb8009c1a72403dea1a0ee0`**]**

6. Задатак **7.8**.: Позивање промене кориснички контролисаног торња

   Циљ: Практична примена статичких метода.

   Концепти за дискусију: методе, статичке методе класе, статички атрибути.

   Активност: Ручно тестирајте функцију да бисте проверили да ли ради исправно. Након тога, дискутујте са ученицима где би ова функција требало да буде позвана. Метод треба позвати унутар функције act() класе Arena и унутар функције processUserControl(). На крају, учините метод ManualTower.changeControl(Boolean) приватним и посматрајте промене инстанце класе ManualTower.

   [Commit*:* `c052bbb6aa4c7e690d4d8cf55d3831028fa2b9e3`**]**

7. Обнављање теоријских концепата (10 минута)

   Дајте резиме лекције, наглашавајући значај статичких атрибута и метода у ефикасном управљању заједничким својствима класе. Подстакните ученике да наставе да истражују тако што ће применити научене концепте у сопственим пројектима.