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

Sveučilište u Zagrebu
Fakultet organizacije i informatike

OOP4FUN: OBJECT ORIENTED PROGRAMMING FOR FUN

VODIČ ZA NASTAVNIKE SREDNJIH ŠKOLA

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

Autori:
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 Devčić, Dijana Peras, Goran Hajdin, Lea Masnec

Varaždin, rujan 2024.

IMPRESUM

Naslov:
OOP4FUN: OBJECT ORIENTED PROGRAMMING FOR FUN
Vodič za nastavnike srednjih škola

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

Autori:
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 Devčić, Dijana Peras, Goran Hajdin, Lea Masnec

Nakladnik:
Sveučilište u Zagrebu
Fakultet organizacije i informatike
Pavlinska 2, 42000 Varaždin

Za nakladnika:
Prof. dr. sc. Marina Klačmer Čalopa

Prevoditelji:
Davor Fodrek, Marko Mijač, Antonela Čižmešija, Lea Masnec, Kristina Takač i Zlatko Stapić

Grafičko oblikovanje i prijelom teksta:
Dora Jovanovska, Zlatko Stapić

Lektura:
Slađan Lipovec

Tisak:
PRAXA Tiskara, Varaždin

ISBN:
ISBN 978-953-6071-80-7 (tiskano izdanje)
ISBN 978-953-6071-81-4 (e-izdanje)

Naklada:
50 kom.

Financirano sredstvima Europske unije.

Ovaj vodič objavljen je na češkom, slovačkom, njemačkom, srpskom i hrvatskom jeziku.

CIP zapis dostupan je u računalnome katalogu Nacionalne i sveučilišne knjižnice u Zagrebu pod brojem 001254089.

PODACI O PROJEKTU

Naziv projekta:
Object Oriented Programming for Fun

Akronim projekta:
OOP4FUN.

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

Projektni koordinator:
Žilinska univerzita v Žiline (Slovačka)

Projektni partneri:
Sveučilište u Zagrebu (Hrvatska)
Srednja škola Ivanec (Hrvatska)
Univerzita Pardubice (Češka Republika)
Gymnázium Pardubice (Češka Republika)
Obchodna akademia Povazska Bystrica (Slovačka)
Hochschule fuer Technik und Wirtschaft Dresden (Njemačka)
Gymnasium Dresden-Plauen (Njemačka)
Univerzitet u Beogradu (Srbija)
Gimnazija Ivanjica (Srbija)

Više o projektu dostupno je na https://www.oop4fun.eu/

Co-funded by the Erasmus+ Programme of the European Union

Sadržaj

Co-funded by the Erasmus+ Programme of the European Union

Popis tablica

Co-funded by the Erasmus+ Programme of the European Union

Predgovor

U posljednjih nekoliko godina primijećen je značajan pad broja učenika koji se zanimaju za tehnički orijentirane studijske programe (STEM) u srednjim školama diljem svijeta. Ovaj trend ukazuje na hitnu potrebu motiviranja učenika i pokazivanja da STEM, a osobito programiranje, nije toliko teško kao što se možda čini. Ključ za uvjeravanje srednjoškolaca u tu činjenicu leži u rukama njihovih učitelja, koji ih vode kroz tehnički orijentirane predmete, a osobito informatiku.

Zbog toga smo 2021. godine odlučili osnovati međunarodni konzorcij od pet visokih škola i pet srednjih škola, predvođenih Sveučilištem u Žilini. Naš cilj bio je razviti i proširiti koncept HOOP, izvorno osmišljen za Slovačku.

Naš projekt „Objektno orijentirano programiranje na zabavan način“ (OOP4FUN – Object Oriented Programming for Fun) osmišljen je s jasnim fokusom na srednjoškolske nastavnike, koji su naša primarna ciljana skupina. Kroz razgovore s profesorima sveučilišta uključenima u projekt, otkriveno je da postoji značajan jaz u razumijevanju osnovnih principa programiranja, osobito u području objektno orijentiranog programiranja (OOP). Svjesni važnosti tih temeljnih znanja, odlučili smo osigurati da srednjoškolski nastavnici dobiju cjelovito razumijevanje OOP-a.

Kako bismo riješili ove probleme, pripremili smo pregled aktualnih metodologija i koncepata u programiranju, posebno prilagođenih nastavnicima srednjih škola. Razvili smo detaljne kurikulume i popratne materijale, uključujući udžbenike i online sadržaje. Naš tim, sastavljen od istraživača i stručnjaka s različitih sveučilišta iz Slovačke, Češke, Hrvatske, Srbije i Njemačke, donio je širok spektar stručnog znanja, znanstvenih spoznaja i profesionalnih postignuća. Jedinstvena misija ovog međunarodnog tima bila je unaprijediti nastavne planove i rezultate projekata usmjerenih na učenike i nastavnike, posebno u kontekstu učenja i poučavanja OOP-a kroz razvoj igara. Ovaj pristup uključuje znanja i vještine vezane uz proces razvoja i suradnje u stvarnim i virtualnim okruženjima.

U ovoj publikaciji nalaze se materijali za pripremu nastavnih sati u okviru različitih predmeta vezanih uz informatiku i programiranje. Mogu se koristiti u cijelosti ili samo kao inspiracija. Pripremljeni primjeri predstavljeni su za rad u Greenfoot okruženju. Prilog publikaciji sadrži opis rješenja konkretnog projekta u Greenfootu.

Želimo da ova publikacija bude korisno pomagalo svima koji žele pružiti kvalitetno obrazovanje iz objektno orijentiranog programiranja u svojim srednjim školama.

Autorski tim

Co-funded by the Erasmus+ Programme of the European Union

Dodatni elektronički izvori

Ovaj vodič na hrvatskom jeziku dostupan je i u elektroničkom obliku i može mu se pristupiti na adresi: https://oop4fun.eu/ i ovdje.

Silabus nastavnog plana i programa poučavanja programiranja temeljem osnovnih načela objektno orijentiranog programiranja na hrvatskom jeziku, a koji se spominje u ovom vodiču često, dostupan je na adresi: https://oop4fun.eu/ i ovdje.

Repozitorij sa izvornim verzijama programskog koda projekta *Tower Defense* koji se spominje u ovom ovidiču, ali i ostali projekti koji se spominju u silabusu dostupni su na: https://gitlab.kicon.fri.uniza.sk/oop4fun.

E-kolegij sa materijalima na engleskom jeziku za podršku ovoj knjizi može se pronaći na platformi Moodle i dostupan je na adresi: https://oop4fun.fon.bg.ac.rs/.

Plan poučavanja i učenja sa raspisanim ishodima učenja, cjelinama i aktivnostima, te načinom izvedbe istih na engleskom jeziku dostupan je na adresi http://learning-design.eu/en/preview/404c42f7d267e15baec68a88/details

Silabus kolegija

GitLab verzije koda

Moodle kolegij

Co-funded by the Erasmus+ Programme of the European Union

1. Uvod

Cilj ove knjige je predstaviti predmet s građom koja će pomoći nastavnicima u pripremi materijala za poučavanje učenika u rješavanju zadataka vezanih uz programiranje, korištenjem osnove objektno orijentiranog programiranja (OOP) te slijedeći laganu OOP paradigmu.

Učenici će naučiti kako podijeliti zadatke među objektima koji surađuju, utvrditi njihove mogućnosti te implementirati dizajnirani model. Poučavanje se provodi pomoću programskog jezika Java u okruženju Greenfoot. To je trenutno vrlo popularan i u praksi široko upotrebljavan programski jezik. Greenfoot predstavlja uređivač izvornog kôda temeljen na okvirima (engl. *frame-based source code editor*) koristeći jezik Stride. To otvara mogućnosti za nastavnike koji će htjeti upotrebljavati tehnike predstavljene u ovom nastavnom planu i programu s učenicima mlađe dobi. Greenfoot je vrlo vizualan i od samog početka omogućuje stvaranje vizualiziranog objekta koji je „živ“ i s kojim se može komunicirati. Stoga je teorijski uvod sveden na minimum, a učenici će krenuti s radom od samog početka.

U predmetu se objašnjavaju osnovni koncepti OOP-a (poput enkapsulacije, nasljeđivanja ili asocijacije) kreiranjem računalnih igara, gdje se spomenuti koncepti koriste na jednostavan i intuitivan način. Proces izrade računalne igre se temelji na timskom radu i praktičnoj primjeni znanja i vještina iz drugih područja informatike i njoj srodnih predmeta (rad s multimedijskim i uredskim programima). Dizajn svake računalne igre je dovoljno otvoren i slobodan, tako da učenici mogu individualno i kreativno proširivati igru pravilno primjenjujući stečeno znanje.

Predmet je usmjeren na uvođenje inovativnog pristupa poučavanju programiranja, temeljenog na rješavanju zadataka korištenjem paradigme objektno orijentiranog programiranja. OOP je danas dominantna

paradigma za razvoj aplikacija, stoga je i primjereno da učenici posjeduju znanja i vještine iz ovog područja. U predmetu je predstavljeno razvojno okruženje koje koristi različite oblike uređivanja izvornog kôda (uređivanje temeljeno na principu okvira – engl. *frame based editing*, ali i na pisanju izvornog kôda), što omogućuje poučavanje učenika s različitim razinama prethodnog tehničkog znanja i aktivnosti. Svojom jednostavnošću i jasnoćom ovaj alat podržava brzo i intuitivno razumijevanje nastavnih tema, što pozitivno utječe na učenike i njihovu motivaciju.

Programiranjem interaktivnih igara u grafičkom okruženju učenici će steći određena znanja i vještine kojima će moći:

- identificirati problem,
- identificirati prikladne objekte za rješavanje identificiranog problema (dekompozicija objekta),
- dizajnirati klase objekata, njihove atribute i metode,
- identificirati i ispravno koristiti veze među objektima (asocijacija, nasljeđivanje),
- dizajnirati algoritam za rješavanje problema i distribuirati ga među objektima koji surađuju,
- koristiti elemente izvornog kôda (grananje, petlje) za implementaciju dizajniranog algoritma,
- učinkovito koristiti sredstva za otklanjanje pogrešaka u izvornom kôdu,
- izraditi jednostavnu aplikaciju s grafičkim sučeljem u okruženju Greenfoot.

Ishodi učenja predmeta su sažeti na sljedeći način:

- razumjeti osnovne principe objektno orijentiranog programiranja,
- razumjeti osnove algoritmizacije,
- razumjeti sintaksu programskog jezika Java,
- analizirati izvršavanje programa na temelju izvornog kôda,
- izraditi vlastiti program korištenjem OOP-a.

Co-funded by the Erasmus+ Programme of the European Union

Suvremen pristup u osmišljavanju predavanja, posebno onih za osnovno i srednjoškolsko obrazovanje, očituje se u definiranju i dijeljenju scenarija poučavanja (engl. *teaching scenario, TS*). Scenariji poučavanja „shvaćaju se kao suvremeni pedagoški pristup koji osnažuje individualizaciju nastavnog procesa uzimajući u obzir različite potrebe učenika. Nastava temeljena na TS-u usmjerena je na relevantna znanja i vještine učenika, uključujući i one koje su potrebne u digitalnom društvu. Pažljivo planiranje scenarija poučavanja može otkloniti potencijalne zamke i nedostatke koji bi mogli utjecati na nastavni proces."

U kontekstu obrazovanja i nastavnog dizajna scenariji poučavanja predstavljaju detaljne opise ili narative koji ocrtavaju specifičnu nastavnu situaciju ili kontekst. Ovi scenariji često se koriste u obuci nastavnika za simulaciju nastavnih situacija iz stvarnog svijeta i stoga se smatraju najboljim alatom za predstavljanje inovativnih ideja poučavanja i učenja. Budući da scenariji poučavanja obično uključuju informacije o ciljevima učenja, sadržaju koji se poučava, karakteristikama učenika, korištenim metodama poučavanja i strategijama evaluacije, oni se također mogu uskladiti s elementima potrebnima za projektom osmišljene artefakte dizajna učenja.

Kako bi se dobio strukturiran pristup u definiranju nekoliko scenarija poučavanja, osmišljen je predložak prikazan u tablici 1, a koji bi trebao biti ispunjen konkretnim podacima vezanim uz određeni scenarij poučavanja. Predložak sadrži kratak opis kako definirati svaki element scenarija poučavanja uključujući naslov, ishode učenja, ciljanu publiku to jest ciljanu učeničku skupinu, trajanje scenarija poučavanja, potrebne materijale i resurse, detaljan opis scenarija, način ocjenjivanja i diseminacije rezultata učenika.

Tablica 1. *Predložak za dokumentiranje scenarija poučavanja*

Naslov	Scenarijima poučavanja dajte opisni naslov koji privlači pozornost.
Ishodi učenja	Jasno navedite planirane ishode učenja. Što bi učenici trebali znati, razumjeti ili moći učiniti do kraja scenarija?
Ciljana publika	Navedite publiku kojoj je namijenjeno, razred i prethodno znanje za koje je scenarij poučavanja dizajniran.
Trajanje scenarija	Procijenite vrijeme potrebno za dovršetak scenarija poučavanja, uključujući sve specifične vremenske okvire za različite aktivnosti, npr. uvod nastavnika (5 min.), samostalno istraživanje učenika (10 min.), programiranje u timu (20 min.), izlaganje/rasprava (10 min.).
Materijali i resursi	Navedite materijale, resurse i alate potrebne i nastavnicima i učenicima. To može uključivati udžbenike, internetske i multimedijske resurse, softver itd.
Opis	• Predstavite scenarij poučavanja, objasnite njegovu svrhu i važnost. • Navedite glavne aktivnosti koje će učenici obavljati kako bi postigli ciljeve učenja. Uključite detalje kao

Co-funded by the Erasmus+ Programme of the European Union

	što su rasprave, praktične aktivnosti, grupni rad, natjecanje itd. • Navedite kako će učenici biti organizirani, tj. hoće li raditi individualno ili u timu te koliko će timovi biti veliki. • Objasnite na kojim će projektima/problemima/zadacima učenici raditi. Preporučuje se korištenje pristupa temeljenog na problemu ili projektu. Oni bi trebali odražavati situacije iz stvarnog života. • Objasnite kako će projekti/problemi/zadaci biti dodijeljeni učenicima (timovima). • Ako se radi u timovima, navedite detalje o tome kako će učenici surađivati. • Navedite više pojedinosti vezanih uz aktivnosti u kojima učenici trebaju sudjelovati. • Ako se koristi npr. obrnuta učionica, odredite koji dio zadane teme učenici trebaju sami istražiti.
Ocjenjiva-nje	Navedite pojedinosti o tome kako će se evaluirati trud i znanje učenika. • Tko će vrednovati učenike: (1) nastavnici, (2) učenici sami svoj rad (samovrednovanje), (3) učenici međusobno (vršnjačko vrednovanje)? • Koji kriteriji vrednovanja će se koristiti? • Koliko često će se provoditi vrednovanje?

Disemina-cija rezultata	Objasnite kako će učenici diseminirati svoje rezultate nastavnicima i drugim učenicima. Npr. učenici (svi ili dio njih) mogu prezentirati svoje rezultate/rješenja pred nastavnikom i svojim vršnjacima, a zatim može uslijediti usporedba i rasprava.

aktivnostima, te načinom izvedbe istih dostupan je u sklopu navedenog alata na adresi http://learning-design.eu/en/preview/404c42f7d267e15baec68a88/details. Materijali za podršku ovoj knjizi mogu se pronaći na platformi Moodle: https://oop4fun.fon.bg.ac.rs/.

Co-funded by the Erasmus+ Programme of the European Union

2. Uvod u okruženje Greenfoot

Greenfoot je vizualni 2D obrazovni softverski alat s uređivačem kôda za izradu igara i simulacija u programskom jeziku Java. Greenfoot je vizualan i interaktivan. Programi su izrađeni u standardnom tekstualnom kôdu Java, pružajući kombinaciju iskustva programiranja u tradicionalnom tekstualnom jeziku s vizualnim izvršavanjem.

2.1. Istraživanje razvoja igara na kreativan način

Tablica 2. *Istraživanje razvoja igara na kreativan način*

Naslov	Istraživanje razvoja igara na kreativan način
Ciljevi učenja	Do kraja nastavne cjeline učenici ne samo da će uspješno instalirati Greenfoot i uvjeriti se u njegove mogućnosti kroz primjere projekata nego će i sudjelovati u zajedničkoj, praktičnoj igri unutar razvojnog okruženja. Ovaj razigrani uvod pobuđuje pozitivno raspoloženje za uzbudljivo istraživanje razvoja igara, potičući kreativnost, timski rad i entuzijastičan pristup kodiranju Greenfootom.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući varijable, funkcije, koncepte iteracije i selekcije.

Trajanje scenarija	1) Uvod (5 min.) 2) "Tko će biti najbrži" (10 min.) 3) Igranje igara s nastavnikom (30 min.) 4) Formiranje tima i dodjela projekta (5 min.) 5) Suradnja u timu i kodiranje (30 min.) 6) Provjera i povratne informacije (10 min.) 7) Domaća zadaća (30 min.) 8) Ocjenjivanje natjecanja (30 min.)
Materijali i resursi	Web stranica Greenfoota i upute za preuzimanje. Primjeri koje je pripremio nastavnik. Internetski izvori za identifikaciju drugih primjera.
Opis	U ovom 90-minutnom scenariju poučavanja učenici će uroniti u svijet Greenfoota pomoću igrifikacije, zabave, istraživanja i timskog rada. Nakon što nastavnik predstavi tematiku koja slijedi, osvrne se na prethodnu lekciju i postavi izazovne ciljeve, počinje izazov "Tko će biti najbrži". Učenici dobivaju igrificirani zadatak da pronađu upute, preuzmu i instaliraju Greenfoot (za njih još nepoznat razvojni alat) na svoja računala. Prva tri učenika dobivaju znakove zahvalnosti (bedževe, bodove, slatkiše itd.). Drugo iznenađenje za njih je to što će sljedećih 30 minuta igrati igre s nastavnikom. On će voditi učenike kroz otvaranje, sastavljanje i pokretanje jednog ili dvaju jednostavnih primjera projekta (od uvodne do srednje razine složenosti). Ovo će učenicima pokazati osnovne elemente razvojnog okruženja Greenfoot i osnovne postupke za rukovanje projektnim datotekama i resursima. Nakon toga učenici će se podijeliti u timove (svaki po 3 ili 4 učenika) i dobiti jednostavan zadatak. Timovi bi trebali promijeniti „nešto" u zadanom primjeru projekta kako bi

Co-funded by the Erasmus+ Programme of the European Union

	igra bila iznenađujuća ili zabavna. Timska suradnja i kodiranje (30 minuta) motivirat će timove na zajednički rad u pokušaju promjene nečega u zadanim primjerima. Ako pokvare kôd izvan granice mogućnosti da ga sami poprave, mogu zatražiti pomoć od nastavnika ili ponovno preuzeti „početnu verziju“. Ovo će biti dobar primjer zašto bi se trebali koristiti sustavi za verzioniranje kôda prilikom kodiranja. Jedan ili dva tima će prezentirati svoj rad za provjeru i povratnu informaciju te razgovarati o rezultatima s nastavnikom. Kod kuće za domaću zadaću svaki bi učenik trebao potražiti primjere igara u Greenfootu, a zatim upoznati razred sa svojim najdražim primjerom učitavajući poveznicu, s opisom onoga što ga čini njegovim najdražim primjerom i s dvije-tri snimke zaslona razvojnog okruženja i pokrenute igre. U sklopu igrifikacije i motivacije putem natjecanja svaki učenik treba glasovati za tri najbolje igre (nije dopušteno glasovati za svoju igru). Pobjednici se proglašavaju i nagrađuju znacima zahvalnosti (bedževi, bodovi, slatkiši itd.).
Ocjenjiva-nje	Ova će aktivnost omogućiti nastavnicima da daju formativne povratne informacije o vrednovanju na temelju rasprava i praćenja obrnute učionice i timskog rada učenika. Igrifikacija predstavlja neformalno vrednovanje, ali će povećati interes, intrinzičnu motivaciju i rezultate učenja cijele grupe. Recenzirana procjena će se provoditi online kao dio domaće zadaće. Ovo će učenike podsjetiti na važne aspekte nastave, motivirat će ih da instaliraju Greenfoot, dati različite

	primjere i podsjetiti ih na ono što je učinjeno tijekom nastave, a to će povećati ukupna postignuća ishoda učenja.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

2.1.1. Priručnik za nastavnike za pripremu lekcije

1. Uvod

Cilj: Upoznati učenike s programskim jezicima (vizualnim i tekstualnim), integriranim razvojnim okruženjima i izvornim kôdom.

Koncepti za raspravu: programski jezici, integrirana razvojna okruženja i izvorni kôd.

Aktivnosti: Nastavnik iznosi kratak uvod za lekciju koja slijedi i uvodi pojmove kao što su:

- programski jezik,
- integrirana razvojna okruženja (alati za programiranje) i
- izvorni kôd.

On predstavlja primjere izvornog kôda u različitim programskim jezicima kao što su Java, C i Python, ali demonstrira i programe u Scratchu ili nekom drugom vizualnom jeziku.

Nastavnik ne ide „u dubinu“, već pokušava objasniti te koncepte koristeći jednostavan primjer, kao što je usporedba učenja programskog jezika s učenjem materinjeg ili stranog jezika.

Co-funded by the Erasmus+ Programme of the European Union

- Prirodni jezici imaju vlastitu gramatiku i pravopis, a isti je slučaj i s programskim jezicima. U pisanju slijedimo neka gramatička i pravopisna pravila, slično kao kad pišemo program u programskom jeziku.
- Kada pišemo priču na materinjem jeziku koristimo bilježnicu i olovku kao alate koji nam u tome pomažu. Slično je i kada pišemo program u nekom programskom jeziku, naš alat za to je integrirano razvojno okruženje.
- Rezultat pisanja može biti priča napisana na papiru, dok je rezultat kodiranja program koji kreiramo, a koji nazivamo izvornim kôdom.

Nastavnik može napraviti uvod vlastitim primjerom.

2. "Tko će biti najbrži"

Cilj: Istražiti Greenfoot i upute za instalaciju.

Koncepti za raspravu: Greenfoot, upute za instalaciju.

Aktivnosti: Nastavnik učenicima daje zadatak da saznaju što je Greenfoot. Od njih traži da pronađu upute za instalaciju Greenfoota. Učenici rade na zadatku, nakon čega im nastavnik prezentira kako pronaći, preuzeti i pokrenuti upute za instalaciju Greenfoota na platformi Moodle.

3. Igranje igara s nastavnikom

Cilj: Pokrenuti različite Greenfootove projekte.

Koncepti za raspravu: Greenfootovi projekti (web projekti i samostalni projekti).

Aktivnosti: Nastavnik daje učenicima zadatak da na internetu pogledaju i pronađu primjere projekata izrađenih u Greenfootu. Oni traže projekte, dok on prati njihov rad.

Nastavnik prezentira učenicima kako mogu pronaći primjere gotovih projekata izrađenih u okruženju Greenfoot. Na primjer, u

tražilici Google projekti se mogu pronaći na temelju ključnih riječi *Greenfoot Java project example* ili slično.

Objašnjava učenicima da postoje dvije vrste Greenfootovih projekata:

- oni koji se mogu pokrenuti u web pregledniku,
- oni koji se mogu preuzeti, a zatim otvoriti i pokrenuti u okruženju Greenfoot.

Nastavnik prezentira projekte:

- koji se mogu pokrenuti direktno u web pregledniku,
- koji se mogu pokrenuti u Greenfootu nakon preuzimanja.

Nastavnik može preuzeti neke primjere projekata ili im pristupiti putem poveznica.

4. **Formiranje tima i dodjela projekata**

Cilj: Uključiti učenike u učenje temeljeno na projektu pomoću jednostavnog zadatka.

Koncepti za raspravu: nije predviđeno za ovu cjelinu.

Aktivnosti: Nastavnik formira timove, priprema zadatak i zadaje projekt na kojem će učenici raditi. Neki od primjera zadataka mogu se pronaći na Moodleu. Nastavnik odabire jedan reprezentativni projekt (ne previše složen) i za njega definira zadatak (problem) koji učenici trebaju riješiti.

5. **Suradnja u timu i kodiranje**

Cilj: Uvesti učenika u izradu projekta u Greenfootu. Učenici rade na dodijeljenom zadatku.

Koncepti za raspravu: kreiranje projekta u Greenfootu.

Aktivnosti: Nastavnik pokreće Greenfoot i pokazuje učenicima kako izraditi projekt te im zadaje zadatak da izrade projekt u Greenfootu.

Co-funded by the Erasmus+ Programme of the European Union

Zadatak 1.1.: Napravite novi projekt, dajte mu odgovarajuće ime (npr. **TowerDefense**) i spremite ga na odgovarajuće mjesto.

Nastavnik naglašava učenicima da projekt (identičan upravo izrađenom projektu) može biti:

- preuzet s Gita[1] unosom odgovarajuće naredbe **Commit**: 9046f5353d857dcc112abd92d7b7170abcc64a80
- preuzet s Gita, ali kao ZIP datoteka sa sljedeće adrese: https://oop4fun.fon.bg.ac.rs/.

Nastavnik naglašava učenicima da na tekućem satu neće raditi na projektu koji su upravo izradili, već će to raditi od sljedećeg sata, a sada će raditi na postojećim projektima.

Učenici preuzimaju zadatak i rješavaju problem.

6. **Provjera i povratne informacije**

Cilj: Rasprava o predloženom rješenju učenika.

Koncepti za raspravu: nije predviđeno za ovu cjelinu.

Aktivnosti: Nastavnik prati rad učenika, a po potrebi im daje smjernice (upute). Nakon završetka bira jedan tim koji će pokazati svoje rješenje. Nastavnik s učenicima raspravlja o predloženom rješenju.

7. **Domaća zadaća**

Cilj: Dati učenicima zadatak da istraže Greenfootove projekte.

Koncepti za raspravu: nije predviđeno za ovu cjelinu.

Aktivnosti: Nastavnik definira zadatak koji učenici trebaju riješiti na jednom od postojećih projekata.

[1]Učenici moraju imati instaliran Git na svojim računalima.

8. Ocjenjivanje natjecanja

Cilj: Uključiti učenike u proces evaluacije učeničkih projekata.

Koncepti za raspravu: nije predviđeno za ovu cjelinu.

Aktivnosti: Učenici prezentiraju svoj projekt. Nastavnik od svih zahtijeva da sudjeluju u evaluaciji prezentiranih projekata slanjem linkova za ispunjavanje Googleovog obrasca. Učenici bodovanjem određuju tri najbolja tima.

Nakon prezentacija nastavnik daje sažetak učeničkih projekata, iznosi svoje dojmove o radu učenika, navodeći je li njima zadovoljan, jesu li ispunili njegova očekivanja ili su ih premašili.

Co-funded by the Erasmus+ Programme of the European Union

3. Definicija klase

U okviru ove tematske cjeline izrađena su dva scenarija poučavanja.

3.1. Istraživanje klasa i objekata uz Greenfoot kroz razvoj igara

Tablica 3. *Istraživanje klasa i objekata uz Greenfoot kroz razvoj igara*

Naslov	Istraživanje klasa i objekata uz Greenfoot kroz razvoj igara
Ciljevi učenja	Do kraja ove nastavne cjeline učenici će moći razumjeti osnovni koncept objekta i klase. Razumijevanje ispitanih koncepata raspravljat će se u kontekstu razvoja igre, poticanja kreativnosti, timskog rada i entuzijastičnog pristupa kodiranju alatom Greenfoot.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući objekt i klasu.
Trajanje scenarija	1) Objekt (10 min.) 2) Identifikacija objekata i njihovih svojstava (15 min.) 3) Klasa, instanca, nasljeđivanje (15 min.) 4) Orijentacija u Greenfootu (10 min.) 5) Konstruktor klase (10 min.) 6) Priprema svijeta (15 min.) 7) Postavljanje slike (10 min.) 8) Priprema grafike svijeta (15 min.)

Materijali i resursi	Web stranica Greenfoota i upute za preuzimanje. Primjeri koje je pripremio nastavnik. Internetski izvori za identifikaciju drugih primjera.
Opis	U ovom 90-minutnom scenariju poučavanja srednjoškolci se upoznaju s načelima programiranja vezanim uz objekt i klasu kroz prizmu razvoja igre pomoću alata Greenfoot. Nastava počinje 10-minutnim uvodom nastavnika, koji upućuje učenike u svijet objektno orijentiranog programiranja objašnjavajući koncept objekta i njegovih svojstava u stvarnom životu. Nakon toga nastavnik postavlja učenicima izazov "Tko će biti najbrži" (10 min.), a oni moraju prepoznati objekte i njihova svojstva na temelju tekstualnog opisa zadatka. Zatim s učenicima osmišljava rješenje zadatka (5 min.). U nastavku nastavnik objašnjava razliku između klase i objekta. Na najvišoj razini apstrakcije objašnjava pojam nasljeđivanja (15 minuta). Nastavnik pokreće okruženje Greenfoot i objašnjava klase **`World`**, **`Actor i MyWord`**. Objašnjava i predstavlja izvorni kôd koji je generiran i stoji iza klasa **`World`**, **`Actor`**, **`MyWorld`** nakon izrade projekata. Nastavnik započinje zadatak 1.2. i pokazuje učenicima kako stvoriti **`World`** (Svijet) za aplikaciju koju trebaju izraditi. Objašnjava kako postaviti sliku za određenu klasu i s učenicima radi na zadatku 1.3..
Evaluacija	Igrifikacija predstavlja neformalno vrednovanje, ali će povećati interes, intrinzičnu motivaciju i rezultate učenja cijele grupe.

Co-funded by the Erasmus+ Programme of the European Union

Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

3.1.1. Priručnik za nastavnike za pripremu lekcije

1. Objekt

Cilj: Upoznati učenike s pojmom objekta na primjerima iz života.

Koncepti za raspravu: objekt i karakteristike objekta.

Aktivnosti: Nastavnik uvodi pojam objekt, a korištenjem primjera iz svakodnevnog života ovaj koncept trebao bi približiti učenicima. Na primjer, nastavnik može pitati učenike o njihovom imenu, visini, datumu rođenja, boji očiju... Postavljajući ova pitanja potaknut će učenike da razmotre po čemu se razlikuju jedni od drugih, pripremajući ih za kasnije pitanje: kako se učenici razlikuju jedni od drugih?

Nastavnik zaključuje da svi (on, učenici) imaju osobine po kojima se međusobno razlikuju te ističe da je svatko od nas zapravo „jedan“ objekt. Objašnjava pojam osobine kao svojstvo koje svatko od nas posjeduje, uz specifičnu vrijednost za svaku osobinu jedinstvenu za svakog pojedinca. Dakle, objekti se razlikuju jedni od drugih na temelju vrijednosti koje posjeduju za svoje karakteristike.

2. Identifikacija objekata i njihovih svojstava

Cilj: Uključiti učenike u prepoznavanje objekata i njihovih svojstava.

Koncepti za raspravu: objekti i njihova svojstva.

Aktivnosti: Nastavnik daje zadatak učenicima da na temelju teksta prepoznaju objekte i njihova svojstva.

3. **Klasa, instanca, nasljeđivanje**

   Cilj: Stjecanje znanja o klasi i instanci. Napraviti razliku između klase i instance.

   Koncepti za raspravu: klasa, instanca.

   Aktivnosti: Kako bi objasnio koncept klase, instanci (objekte) klase i nasljeđivanje, nastavnik daje primjere iz stvarnog života.

   Učenicima postavlja pitanja kako bi uvidjeli razliku između klase i objekta. Nastavnik vodi raspravu o prepoznatim objektima i njihovoj klasifikaciji u klasama.

4. **Orijentacija u Greenfootu**

   Cilj: Predstaviti `World` (Svijet) u Greenfootu.

   Koncepti za raspravu: instanca klase `World` u Greenfootu.

   Aktivnosti: Nastavnik pokreće okruženje Greenfoot i stvara jednostavan projekt, da bi objasnio kako da učenici stvore predstavljeni koncept u Greenfootu.

   Nastavnik ističe da svaki projekt izrađen u Greenfootu sadrži tri klase: `World`, `Actor i MyWorld.`

   Zatim predstavlja klasu `MyWorld` i njezinu ulogu.

   Naglašava da se pozadina svake aplikacije izrađene u Greenfootu sastoji od ćelija koje predstavljaju jednu matricu. Nastavnik pokazuje kako definirati veličinu pozadine (dimenzije matrice) i veličinu svake ćelije matrice.

   Objašnjava da se objekti koji se pojavljuju na ekranu („sceni“) nalaze na jednoj od tih ćelija i pokazuje izvorni kôd svake od ovih klasa.

5. **Konstruktor klase**

   Cilj: Predstaviti konstruktor klase

   Koncepti za raspravu: konstruktor.

Co-funded by the Erasmus+ Programme of the European Union

Aktivnosti: Nastavnik pokazuje izvorni kôd i predstavlja koncept konstruktora.

6. **Priprema svijeta**

   Cilj: Uključiti učenike u rad na projektnom zadatku

   Koncepti za raspravu: svijet u Greenfootu.

   Aktivnosti: Nastavnik učenicima daje zadatak 1.2**.** iz projekta **Tower Defense**: napravite svijet veličine 10x10 ćelija. Svaka ćelija treba biti veličine 75 piksela.

   Nastavnik prati rad učenika i na kraju zamoli jednog od njih da pokaže svoje rješenje i opiše ga.

   Opis rješenja: Uredite izvorni kôd klase **`MyWorld`** (dvaput kliknite na njega) da biste kreirali svijet veličine 10x10 ćelija. Svaka ćelija treba biti veličine 75 piksela.

Commit: a593cd4a92d0fa0db78275614c3e41a2e96b4e57

7. **Postavljanje slike**

   Cilj: Postaviti sliku svijeta u projektu Greenfoot.

   Koncepti za raspravu: postavke slike za klasu **`World`.**

   Aktivnosti: Nastavnik objašnjava učenicima da pozadina aplikacije (ili svijeta) u Greenfootu može biti slika. Pokazuje im da pozadina može biti jedna slika koja pokriva cijelo područje svijeta ili pak slika veličine koja odgovara dimenzijama ćelije.

   Nastavnik pokazuje kako postaviti sliku u klasu **`MyWorld`**, a učenici prate njegove upute i rade zajedno korak po korak.

8. **Priprema grafike svijeta**

   Cilj: Upoznati učenike s pozadinom klase **`World`**.

   Koncepti za raspravu: pozadina klase **`World`**.

Aktivnosti: Nastavnik zadaje učenicima zadatak **1.3. iz** projekta **Tower Defense**: napravite odgovarajuću sliku za pozadinu svijeta i postavite je, pojašnjavajući im što od njih očekuje. Nastavnik može preuzeti završne projekte koje je već izradio i pokazati ih. Učenicima daje poveznicu ili naredbu u Gitu za preuzimanje početnog projekta koji trebaju ažurirati ovim zadatkom (značajkom). Inicijalni projekt se može preuzeti iz repozitorija Git.

Učenici izvršavaju zadatak samostalno ili u skupini, dok nastavnik prati njihov rad.

Nastavnik demonstrira rješenje korak po korak, a učenici prate upute.

Opis rješenja:

- Pronađite ili stvorite odgovarajuću sliku za pozadinu svijeta. Možete koristiti pripremljene slike (odaberite stavku **`Set image…`** iz kontekstnog izbornika klase **`MyWorld`**) ili prilagođenu sliku (kopirajte sliku u podmapu *Images* u mapi svog projekta i odaberite je na isti način kao što je prethodno opisano).
- Kao pozadinu možete koristiti jednu sliku koja će pokrivati cijeli svijet (izračunajte potrebnu veličinu slike s obzirom na veličinu svijeta) ili manju koja će se više puta kopirati (koristite kvadratnu sliku veličine ćelije).

Commit:1184980643db082cfdd6bde9984bceaddf010d49

Co-funded by the Erasmus+ Programme of the European Union

3.2. Stvaranje klasa i objekata uz Greenfoot kroz razvoj igara

Tablica 4. *Stvaranje klasa i objekata uz Greenfoot kroz razvoj igara*

Naslov	Stvaranje klasa i objekata uz Greenfoot kroz razvoj igara
Ishodi učenja	Do kraja ove nastavne cjeline učenici će razumjeti osnovne koncepte objekta, klase, svojstava klase i metoda.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući objekt, klasu, svojstva klase i metode.
Trajanje scenarija	1) Osnovni koncepti (25 min.) 2) Stvaranje klase **`Enemy`** (10 min.) 3) Stvaranje instance klase **`Enemy`** (30 min.) 4) Sučelje objekta (5 min.) 5) Poruka i metoda (15 min.) 6) Slanje poruke instanci klase (30 min.) 7) Revizija teorije (15 min.)
Materijali i resursi	Web stranica Greenfoota i upute za preuzimanje. Primjeri koje je pripremio nastavnik. Internetski izvori za identifikaciju drugih primjera.
Opis	Tijekom nastave učenici će napredovati kroz nekoliko strukturiranih zadataka kako bi produbili razumijevanje koncepata objektno orijentiranog programiranja. U početku će provesti 25 minuta stvarajući klasu pod nazivom

	Enemy, nakon čega će uslijediti fokusirani 15-minutni zadatak za definiranje atributa i metoda unutar ove klase. Zatim će izdvojiti 30 minuta za instanciranje objekta klase **Enemy**, primjenjujući svoje znanje o stvaranju i inicijalizaciji objekta. Poslije toga će u 5 minuta istražiti koncept sučelja objekta, s naglaskom na definiciji operacija koje objekt može izvoditi. Potom će učenici tijekom 15 minuta proniknuti u koncepte poruka i metoda, učeći kako objekti komuniciraju kroz pozivanje metoda. Zatim će se još 30 minuta posvetiti praktičnoj primjeni, pa će učenici slati poruke svojoj instanciranoj instanci **Enemy**, produbljujući svoje razumijevanje ponašanja objekta. Konačno, 15-minutna teorijska revizija će rekapitulirati ključne koncepte kao što su objekti, klase, instance, unutarnje stanje, identitet, poruke i metode, osiguravajući sveobuhvatno razumijevanje ishoda učenja.
Evaluacija	Igrifikacija predstavlja neformalno vrednovanje, ali će povećati interes, intrinzičnu motivaciju i rezultate učenja cijele grupe.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

Co-funded by the Erasmus+ Programme of the European Union

3.2.1. Priručnik za nastavnike za pripremu lekcije

1. Osnovni koncepti

Cilj: Utvrditi znanje o klasi, njezinim svojstvima i objektima.

Koncepti za raspravu: klasa, svojstva i objekti.

Aktivnosti: Tijekom uvoda u ovu cjelinu nastavnik s učenicima ponavlja prethodno obrađene pojmove i kroz raspravu pojašnjava pojmove klasa, svojstva klasa i objekt.

Nastavnik učenicima daje pismeni zadatak, upućujući ih da identificiraju specifične klase, njihova svojstva, pridružene objekte i odgovarajuće vrijednosti unutar ponuđenog teksta.

Na kraju odabire učenika iz razreda koji će prezentirati svoje rješenje. Tijekom ove aktivnosti drugi učenici sudjeluju iznoseći svoja mišljenja.

Nastavnik detaljno opisuje kako kreirati klasu u okruženju Greenfoot. Učenici prate njegove upute i rade zajedno korak po korak.

2. Stvaranje klase Enemy

Cilj: Uključiti učenike u rad na projektu.

Koncepti za raspravu: stvaranje klase.

Aktivnosti: Nastavnik učenicima daje zadatak **1.4.** iz projekta **Tower Defense**: napravite klasu **`Enemy`** i stavite odgovarajuću sliku za nju.

Objašnjava učenicima što od njih očekuje, a pritom može preuzeti završne projekte koje je već izradio i pokazati ih. Daje im poveznicu ili naredbu za Git za preuzimanje početnog projekta koji trebaju ažurirati ovim zadatkom (značajkom). Inicijalni projekt se može preuzeti iz repozitorija Git.

Učenici izvršavaju zadatak samostalno ili u grupi, dok nastavnik prati njihov rad. Nastavnik korak po korak demonstrira rješenje, a učenici prate upute.

Opis rješenja:

Stvorite neprijatelja koji će marširati prema igračevom tornju da ga ošteti i na kraju uništi. Napravite novu podklasu klase **`Actor`** (odaberite stavku **`Actor`** desnim klikom miša i odaberite **`New subclass…`** iz kontekstnog izbornika klase **`Actor`**). Pridružite klasi odgovarajuće ime (**`Enemy`**) i sliku. Nastavnik bi trebao objasniti konvenciju prema kojoj klasa treba započeti velikim slovima i cijelu konvenciju imena u Javi.

Commit: 4981400623729c3d112b54454b6e6151e18426bf

3. **Stvaranje instance klase `Enemy`**

Cilj: Upoznati stanje objekta. Naučiti učenike da kreiraju instancu klase u Greenfootu.

Koncepti za raspravu: stanje objekta, instanca.

Aktivnosti: Nastavnik objašnjava pojam stanja objekta na jednostavnom primjeru. Recimo, nastavnik se nalazi u učionici na određenoj udaljenosti od ulaznih vrata. Ova udaljenost određuje njegov položaj, s koracima naprijed ili natrag koji mijenjaju njegovu blizinu vratima. Stoga se nastavnikov položaj u odnosu na ulazna vrata može kvantificirati varijablom koja se mijenja tijekom vremena. Dakle, pozicija nastavnika definirana je udaljenošću od ulaznih vrata, ilustrirajući kako se stanje objekta – u ovom slučaju, nastavnika – karakterizira vrijednošću njegovog atributa (udaljenosti) u bilo kojem trenutku.

Nastavnik opisuje zadatak **1.5** iz projekta **Tower Defense,** kako stvoriti instancu klase **`Enemy`**. Otvara posljednju verziju projekta, dok učenici slijede njegove upute i rade zajedno korak po korak.

Co-funded by the Erasmus+ Programme of the European Union

Nastavnik stvara instancu klase **`Enemy`** (odabrati desnim klikom miša stavku **`New Enemy()`** iz kontekstnog izbornika klase **`Enemy`**, staviti instancu u svijet lijevim klikom miša na željenu poziciju). Istražite njegovo unutarnje stanje (odaberite mjesto stavke u svijetu desnim klikom miša i odaberite Inspect iz kontekstnog izbornika kreirane instance).

Nastavnik traži od učenika da kreiraju novu instancu i stave je na drugu poziciju kako bi usporedili unutarnja stanja dviju stvorenih instanci.

4. Sučelje objekta

Cilj: Upoznati učenike sa sučeljem kao skupom radnji koje možemo izvesti na nekom objektu.

Koncepti za raspravu: sučelje.

Aktivnosti: Nastavnik kroz jednostavne primjere upoznaje učenike s pojmom sučelja. Na primjer, ako se promatra osobu i aktivnosti koje obavlja tijekom dana (kao što je buđenje, doručak, odlazak na posao) bez ulaženja u određene detalje – poput načina na koji se budi (uz budilicu, telefon ili ga budi roditelj), što i gdje doručkuje ili način prijevoza na posao – zbir ovih aktivnosti može se usporediti sa sučeljem. Ono definira koje radnje objekti određene klase mogu izvesti, ali ne specificira kako se te radnje provode ili izvršavaju. Ne definira se što je osoba jela za doručak, kako je otišla na posao (pješice, autom ili autobusom) ili kako se probudila (je li ju netko zvao ili je budi sat).

5. Poruka i metoda

Cilj: Upoznati učenike s konceptom metode.

Koncepti za raspravu: metoda klase.

Aktivnosti: Nastavnik uvodi pojam metode. Da bi objasnio koncept metode, povezuje pojmove svojstava (karakteristika, atributa) i promjenjivih vrijednosti za ta svojstva.

Primjer 1: ako uzmemo u obzir klasu osoba i svojstvo godine, njena se vrijednost povećava za jedan svake godine, dosljedno na isti dan. S druge strane, ako pogledamo svojstva poput visine i težine, te se karakteristike često mijenjaju – ljudi rastu i njihova težina varira.

Primjer 2: ako promatramo kretanje osobe od točke A do točke B i opisujemo to kretanje u koracima – npr. korak naprijed, skretanje ulijevo za 45 stupnjeva, 8 koraka naprijed, skretanje udesno za 30 stupnjeva i još 5 koraka naprijed – ove pojedinačne radnje mogu se grupirati u ono što nazivamo metodom.

Nastavnik pokazuje učenicima u Greenfootu kako mogu pogledati metode koje klasa ima i koje se mogu pozvati na određenom objektu.

Pokazuje metode definirane u klasi **`Actor`** te kako pozvati metode na objektu.

6. Slanje poruke instanci klase

Cilj: Upoznati učenike kako pozvati metodu na objektu.

Koncepti za raspravu: pozivanje metode.

Aktivnosti: Nastavnik opisuje zadatak **1.6.** iz projekta **Tower Defense**, slanje poruke instanci klase **`Enemy`** (pozivanje metode). Otvara posljednju verziju projekta, dok učenici slijede njegove upute i rade zajedno korak po korak.

Nastavnik šalje poruke instanci klase **`Enemy`** tako da će se pomaknuti na poziciju [12, 6] i bit će okrenuta prema dolje (desnim klikom na objekt u svijetu, odabrati **`Inherited from Actor`** i zatim odabrati metodu **`setLocation(int, int)`**. Opisat će učenicima što

Co-funded by the Erasmus+ Programme of the European Union

će se dogoditi s instancom i kako je to utjecalo na unutarnje stanje dotične instance.

Nastavnik demonstrira učenicima kako se definiraju metode. Stvara metodu **`setPosition(int x, int y)`** za postavljanje Aktera na određene koordinate. Naglašava da je ova metoda ekvivalentna metodi **`setLocation(int x, int y)`** i ističe važnost provjere postoji li ekvivalentna metoda prije definiranja nove kako bi se izbjeglo dupliciranje. Napominje da nazivi metoda trebaju jasno označavati njihovu svrhu i funkcionalnost, omogućujući trenutno razumijevanje iz samog naziva. Nastavnik također naglašava da nazivi metoda trebaju biti sažeti i daje primjere dobro definiranih, loše definiranih i nepravilno definiranih metoda. Metode koje korisnik može izvesti (pozvati) vidljive su desnim klikom na objekt.

Nastavnik opisuje kako definirati metodu, a učenici slijede njegove upute i rade zajedno korak po korak.

Nastavnik od učenika traži da definiraju način na koji se Akter spušta (smanjuje Y-koordinatu) i način na koji se podiže (povećava Y-koordinatu). Prati učenike kako rade na zadatku i, ako je potrebno, daje upute individualno.

7. Revizija teorije

Cilj: Rezimiranje koncepata koji su obrađivani na nastavi.

Koncepti za raspravu: objekt, klasa, instanca, unutarnje stanje, identitet, poruka, metoda.

Aktivnosti: Nastavnik rezimira koncepte koji su obrađeni na nastavi.

4. Algoritam

U ovoj tematskoj cjelini kreirana su dva nastavna scenarija koja se odnose na algoritme u Greenfootu.

4.1. Uvod u algoritme u okruženju Greenfoot

Tablica 5. *Uvod u algoritme i algoritamsko razmišljanje*

Naslov	Uvod u algoritme i algoritamsko razmišljanje
Ishodi učenja	Na kraju lekcije učenici bi trebali dubinski poznavati algoritme i algoritamsko razmišljanje, biti vješti u dizajniranju i implementaciji osnovnih algoritama te sposobni primijeniti algoritamske koncepte kako bi učinkovito rješavali različite probleme.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući varijable, funkcije, koncepte iteracije i selekcije, koji su vješti u logičkom razmišljanju i rješavanju problema. Učenici bi trebali biti upoznati i s Greenfootom.
Trajanje scenarija	1) Uvod u osnovne algoritme kao slijed koraka (15 min.) 2) Zadatak pisanja jednostavnog algoritma(20 min.) 3) Algoritam i njegova svojstva (15 min.) 4) Zadatak pisanja općenitog algoritma (25 min.) 5) Algoritmizacija (15 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s platforme GitHub/GitLab.

Co-funded by the Erasmus+ Programme of the European Union

	Internetski resursi.
Opis	U ovom scenariju poučavanja učenici srednjih škola će istražiti svijet algoritama i algoritamskog razmišljanja. Lekcija započinje 15-minutnim uvodom, s ciljem upoznavanja učenika s osnovnim konceptima algoritama, ističući njihovu važnost u rješavanju problema. Nakon uvoda učenici će sudjelovati u 20-minutnom zadatku u kojem će trebati napisati jednostavan algoritam za rješavanje određenog problema. Ova praktična aktivnost omogućava učenicima da primijene ranije uvedene koncepte i usavrše svoje algoritamske vještine. Slijedi 15-minutni segment posvećen raspravi o algoritmima i njihovim svojstvima. Obradit će se teme poput ispravnosti, učinkovitosti i skalabilnosti, s naglaskom na važnosti jasnih i preciznih uputa u dizajnu algoritama. Nadovezujući se na stečeno znanje, učenici će sljedećih 25 minuta izrađivati općenitiji algoritam za nešto složeniji problem. Ovaj zadatak potiče učenike na apstraktno i kritičko razmišljanje te na primjenu algoritamskih principa za rješavanje stvarnih situacija. U završnom, 15-minutnom segmentu učenici će se baviti algoritmizacijom, analizirajući i poboljšavajući svoje algoritme. Ovaj proces uključuje identificiranje mogućih poboljšanja, optimiziranje učinkovitosti i osiguravanje robusnosti algoritama. Tijekom lekcije učenici će raditi pojedinačno ili u malim grupama, čime se potiče suradnja i učenje kroz interakciju s vršnjacima. Aktivnim sudjelovanjem u pisanju i analizi

	algoritama učenici će razviti vještine kritičkog razmišljanja i sposobnosti rješavanja računalnih problema. Na kraju lekcije učenici će bolje razumjeti algoritme i algoritamsko razmišljanje, što će ih opremiti ključnim vještinama za sustavan i učinkovit pristup složenim problemima.
Ocjenjivanje	Igrifikacija predstavlja neformalnu evaluaciju, ali će povećati interes, unutarnju motivaciju i ishode učenja cijele grupe.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

4.1.1. Priručnik za nastavnike za pripremu lekcije

1. Uvod u osnovne algoritme kao slijed koraka

Cilj: Uvesti pojam algoritma.

Koncepti za raspravu: osnove algoritma.

Aktivnosti: Nastavnik uvodi pojam algoritma kroz primjere iz stvarnog života. Primjerice, pita učenike što rade ujutro od trenutka kada se probude do trenutka kada dođu u školu. Zatim ih pita znaju li napraviti pizzu ili topli sendvič, ili znaju li recept za pripremu nekog jela ili kolača.

Nastavnik povezuje proces pripreme obroka ili kolača s izradom programa te naglašava da, baš kao što postoji recept za pripremu obroka, postoji i „recept“ za izradu programa koji se zove algoritam.

Co-funded by the Erasmus+ Programme of the European Union

Nastavnik zaključuje da je algoritam skup koraka koji definiraju kako se program izvodi.

Zatim objašnjava da taj skup koraka ne mora nužno uvijek biti izvršen sekvencijalno, jedan za drugim, te da postoje neki koraci u algoritmu koji se mogu izvršiti ovisno o nekom uvjetu. Nastavnik traži od učenika da navedu primjer takvog slučaja (recimo, ako pišemo algoritam za pripremu toplog sendviča, no nemamo neki sastojak, primjerice šunku, ali imamo sličan sastojak, možemo ga zamijeniti ili otići u trgovinu i nabaviti nedostajući sastojak).

Nastavnik pojašnjava učenicima da se neki koraci u algoritmu mogu ponoviti nekoliko puta i traži od njih da navedu primjer algoritma u kojem se koraci ponavljaju više puta.

2. Zadatak pisanja jednostavnog algoritma

Cilj: Uvesti učenike u pisanje jednostavnog algoritma.

Koncepti za raspravu: algoritam.

Aktivnosti: Nastavnik zadaje učenicima zadatak**,** da na papiru navedu postupak kojim opisuju kako pješak prelazi ulicu.

Na početku nastavnik ne daje dodatne upute učenicima, već prati kako razmišljaju i rade. Ako netko postavi pitanje koje je važno za opis uputa za prelazak pješaka preko ulice, pohvaljuje ga i naglašava zašto je ta informacija važna.

Nakon nekog vremena nastavnik pita učenike jesu li obratili pažnju na to gdje pješak prelazi ulicu, je li to označeno mjesto za prelazak ili nije. Pita ih i jesu li razmišljali o tome postoji li na pješačkom prijelazu semafor ili ne.

Na kraju nastavnik odabire nekoliko učenika koji će pročitati svoje upute za prelazak ulice.

3. **Algoritam i njegova svojstva**

   Cilj: Upoznati učenike sa svojstvima algoritma.

   Koncepti za raspravu: svojstva algoritma.

   Aktivnost: Nastavnik dalje objašnjava učenicima svojstva algoritma i tumači im da se algoritmi također mogu prikazati grafički te iznosi primjer algoritama koji su grafički prikazani.

4. **Zadatak pisanja općenitog algoritma**

   Cilj: Napisati općenitiji algoritam.

   Koncepti za raspravu: pisanje algoritma.

   Aktivnosti: Napišite općenit algoritam za pripremu toplog napitka. Razmislite o tome koji su ulazni podaci potrebni za takav algoritam kako bi bio općenit.

5. **Algoritmizacija**

   Cilj: Pružiti učenicima više primjera i uključiti ih u definiranje vlastitih algoritama.

   Koncepti za raspravu: algoritam.

   Aktivnosti: Nastavnik objašnjava učenicima da čak i u matematici postoje određeni algoritmi koje koristimo u rješavanju problema i pita ih mogu li se sjetiti kojeg primjera.

   Primjer koji nastavnik objašnjava učenicima je primjer izračuna vrijednosti aritmetičkog izraza koji sadrži nekoliko matematičkih operacija, pri čemu treba poštovati pravila prioriteta izvođenja matematičkih operacija.

   Također, nastavnik daje i neke druge primjere, poput sastavljanja novog namještaja koji je kupljen i dolazi s uputama kako sastaviti taj namještaj. Primjer mogu biti i upute koje primamo putem GPS uređaja kada želimo ići od točke A do točke B pomoću navigacije.

   Nastavnik traži od učenika da na papiru napišu vlastiti algoritam, nakon čega neki od njih predstavljaju svoj rezultat.

Co-funded by the Erasmus+ Programme of the European Union

4.2. Avanture u Greenfootu: razotkrivanje pozivanja metoda u Javi, dokumentacije i kontrole aplikacije

Tablica 6. *Avanture u Greenfootu: razotkrivanje pozivanja metoda u Javi, dokumentacije i kontrole aplikacije*

Naslov	Avanture u Greenfootu: razotkrivanje pozivanja metoda u Javi, dokumentacije i kontrole aplikacije
Ishodi učenja	Na kraju ove nastavne cjeline učenici bi trebali solidno razumijevati pozivanje metoda u Javi, s posebnim naglaskom na metode **`act()`** i **`move()`** u okruženju Greenfoot. Trebali bi biti vješti u učinkovitoj upotrebi ključne riječi **`this`** za referenciranje trenutnih objekata unutar konteksta klase. Osim toga, trebali bi pokazati sposobnost pozivanja metoda unutar klase, razumijevajući sintaksu i parametre potrebne za pozivanje metoda. Trebali bi također demonstrirati vještinu primjene tehnika pozivanja metoda kako bi učinkovito riješili zadatke razvoja interaktivnih igara. Nadalje, učenici bi trebali shvatiti važnost dokumentacije kôda i biti sposobni učinkovito dokumentirati Java kôd, osiguravajući jasnoću i čitljivost. Na kraju, trebali bi savladati tehnike kontrole aplikacije u Greenfootu, omogućujući preciznu manipulaciju i interakciju s elementima igre kako bi stvorili zanimljive i funkcionalne mehanike igre.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući koncepte iteracije i selekcije, koji poznaju i okruženje Greenfoot.

Trajanje scenarija	1) Objašnjenje metode **`act()`** (10 min.) 2) Objašnjenje metode **`move()`** (20 min.) 3) Uvođenje ključne riječi **`this`** (5 min.) 4) Pozivanje metode (10 min.) 5) Objašnjenje funkcionalnosti automatskog dovršavanja (5 min.) 6) Važnost dokumentacije koda (15 min.) 7) Uređivanje dokumentacije metode (5 min.) 8) Uređivanje dokumentacije klase (5 min.) 9) Istraživanje dokumentacije (10 min.) 10) Istraživanje kontrola aplikacije Greenfoot (20 min.) 11) Rasprava: Algoritam, svojstva, algoritmizacija, gumbi u Greenfootu (5 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	U ovom scenariju poučavanja u trajanju od 105 minuta učenici srednjih škola će započeti putovanje kako bi savladali pozivanje metoda u Javi, dokumentaciju kôda i kontrolu aplikacije unutar okruženja Greenfoot. Lekcija započinje 15-minutnim istraživanjem metode **act()**, nakon čega slijedi 10-minutno istraživanje metode **move()**, ključnih komponenti pri programiranju u Greenfootu.

Co-funded by the Erasmus+ Programme of the European Union

	Učenici će potom provesti 5 minuta istražujući važnost ključne riječi **this** za referenciranje trenutnih objekata unutar konteksta klase. Zatim učenike čeka 10-minutni zadatak u kojem će vježbati pozivanje metoda unutar klase, primjenjujući sintaksu i parametre potrebne za pozivanje metoda. Nakon toga slijedi 5-minutno objašnjenje funkcionalnosti automatskog dovršavanja u okruženju Greenfoot, s naglaskom na poboljšanju učinkovitosti kodiranja. U sljedećih 15 minuta učenici će shvatiti važnost dokumentacije kôda, razumijevajući kako jasna i sažeta dokumentacija poboljšava čitljivost i održivost kôda. U 5-minutnom zadatku učenici će dodati dokumentaciju svom kôdu, osiguravajući jasnoću i razumljivost za sebe i druge. Nadovezujući se na ovaj zadatak, učenici će provesti dodatnih 5 minuta dodajući detaljniju dokumentaciju svom kôdu. U 10-minutnom zadatku koji slijedi učenici će istraživati i čitati dokumentaciju koju su dodali njihovi kolege, stječući uvid u različite stilove kodiranja i pristupe. Na kraju, učenici će provesti 20 minuta istražujući tehnike kontrole aplikacije u Greenfootu, omogućujući preciznu manipulaciju i interakciju s elementima igre kako bi stvorili zanimljive i funkcionalne mehanike igre. Tijekom cijele lekcije učenici će raditi individualno ili u malim grupama, što potiče suradnju i učenje od kolega. Aktivnim sudjelovanjem u pisanju i analizi koda učenici

	će razvijati kritičko razmišljanje i sposobnosti rješavanja problema u računarstvu. Na kraju lekcije učenici će imati dublje razumijevanje pozivanja metoda u Javi, dokumentacije kôda i kontrole aplikacije u Greenfootu i bit će opremljeni ključnim vještinama za daljnji razvoj igara.
Evaluacija	Igrifikacija predstavlja neformalnu procjenu, ali će povećati interes, unutarnju motivaciju i ishode učenja cijele grupe.
Disemina-cija rezultata	Za diseminaciju svojih rezultata nastavnici i učenici koristit će uobičajeni sustav GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle). Učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

4.2.1. Priručnik za nastavnike za pripremu lekcije

1. Objašnjenje metode `act()`

Cilj: Objasniti upotrebu metode **`act()`** i pozivanje metode iz druge metode.

Koncepti za raspravu: metoda **`act().`**

Aktivnosti: Nastavnik otvara zadnju verziju projekta **TowerDefense** i postavlja instance klase **`Enemy`** na poziciju 0,0. Pita učenike što će se dogoditi kada pozovemo metodu **`act()`** na instanci klase **`Enemy`** i je li to očekivano ponašanje.

Zatim od učenika traži da mu pomognu izvršiti zadatak: pomicanje objekta klase **`Enemy`** za dvije ćelije u trenutnom smjeru kada pozovemo metodu **`act().`**

Co-funded by the Erasmus+ Programme of the European Union

Nastavnik pokazuje kako izvršiti ovaj zadatak (dodavanjem poziva metode `move(int)` unutar metode `act()`), dok učenici prate upute nastavnika i iznose mu ideje.

2. **Objašnjenje metode `move()`**

   Cilj: Objasniti metodu `move()` i kako se kretati naprijed i nazad.

   Koncepti za raspravu: metode za kretanje.

   Aktivnost: Nastavnik objašnjava metodu `move(int)` te postavlja instance klase `Enemy` na različite pozicije i poziva metodu unoseći pozitivne vrijednosti, na primjer 1 ili 3. Zatim pita učenike što misle da će se dogoditi ako pozovu metodu `move()` i unesu negativne vrijednosti, na primjer -1.

   Zadaje im zadatak da napišu metodu `backward()` koja pomiče objekt za 1 korak unatrag. Je li moguće pomaknuti objekt za 2 koraka unatrag i za X koraka? Što trebamo učiniti?

3. **Uvođenje ključne riječi `this`**

   Cilj: Uvođenje ključne riječi `this.`

   Koncepti za raspravu: ključna riječ `this.`

   Aktivnosti: Nastavnik uvodi ključnu riječ `this.`

4. **Pozivanje metode**

   Cilj: Naučiti učenike kako napisati metodu za pomicanje objekta vertikalno mijenjajući vrijednost osi Y.

   Koncepti za raspravu: pozivanje metode.

   Aktivnost: Nastavnik pita učenike kako pomaknuti objekt vertikalno, gore ili dolje i traži od njih da pronađu prikladnu metodu za pomicanje gore/dolje desnim klikom na objekt i odabirom opcije `Inherited from Actor`. Nastavnik učenicima daje zadatak 2.4. iz projekta **Tower Defense.** Učenici trebaju prepoznati sljedeće metode: `turn`, `setRotation`, `setLocation`, `getLocation`, `getRotation.`

Potom im zadaje zadatak da napišu metode **`up() i down() te`** prati njihov rad i na kraju zajedno s njima objašnjava kako implementirati te metode. Metode **`up() i down()`** mijenjaju vrijednost koordinata na osi Y, povećavajući je i smanjujući. Ovdje nastavnik može uvesti parametre metode, ali bez detalja.

5. **Objašnjenje funkcionalnosti automatskog dovršavanja**

   Cilj: Uvesti učenike u funkcionalnost automatskog dovršavanja u okruženju Greenfoot.

   Koncepti za raspravu: automatsko dovršavanje (CTRL+SPACE).

   Aktivnosti: Nastavnik opisuje učenicima kako pronaći metodu na objektu koju mogu pozvati ako su zaboravili naziv ili ako tek počinju s kodiranjem te žele „zamoliti“ Greenfoot da im pomogne brže završiti pisanje kôda.

6. **Važnost dokumentacije kôda**

   Cilj: Uvesti učenike u komentare i dokumentaciju.

   Koncepti za raspravu: komentari i dokumentacija, istraživanje prozora dokumentacije.

   Aktivnost: Nastavnik uvodi učenike u naredbe u kôdu koje nisu dio izvršavanja (komentari, komentari dokumentacije), pri čemu treba naglasiti razliku između komentara i dokumentacije (kao posebne vrste komentara).

   Prikazuje im izvorni kôd klase **`Enemy`**, a zatim generirani HTML dokument koji opisuje dokumentaciju klase **`Enemy`**. Nastavnik ovdje ističe da postoje pravila koja trebamo slijediti prilikom pisanja dokumentacije za svoju klasu ili metode.

   Zadaje im zadatak da istraže kako napisati dokumentaciju za klasu i metode u Javi.

7. **Uređivanje dokumentacije metode**

   Cilj: Uvesti učenike u oznaku dokumentacije metode.

   Koncepti za raspravu: oznaka dokumentacije metode.

Co-funded by the Erasmus+ Programme of the European Union

Aktivnosti: Nastavnik zadaje učenicima zadatak da dodaju dokumentacijski komentar za metodu **`act()`.** Nastavnik učenicima objašnjava zadatak **2.4.** iz projekta **Tower Defense.**

Commit*:* 68b1c82c7df2c7826f2d3f78373498569adab7e9

8. **Uređivanje dokumentacije klase**

   Cilj: Uvesti učenike u oznaku dokumentacije klase.

   Koncepti za raspravu: oznaka dokumentacije klase.

   Aktivnosti: Nastavnik učenicima objašnjava zadatak **2.5.** iz projekta **Tower Defense.** Uredite dokumentacijski komentar klase **`Enemy`**. Dodajte verziju klase i njenog autora te pogledajte promjene u generiranoj HTML stranici.

Commit*:* 1a7a9f83c5271a7c0dfa46ce3b1ee65682b0c5e5

9. **Istraživanje dokumentacije**

   Cilj: Naučiti učenike kako da istražuju dokumentaciju klase da bi se upoznali s metodama.

   Koncepti za raspravu: istraživanje dokumentacije.

   Aktivnosti: Nastavnik učenicima objašnjava zadatak 2.6. iz projekta **Tower Defense.** Nastavnik zadaje učenicima zadatak da istraže prozor dokumentacije i pročitaju dokumentaciju za klase **`Actor`** **`i`** **`Window`**, pri čemu naglašava važnost čitanja dokumentacije kako bi se otkrile metode koje mogu biti korisne.

10. **Istraživanje kontrola aplikacije Greenfoot**

    Cilj: Uvesti učenike u kontrole aplikacija u Greenfootu.

    Koncepti za raspravu: kontrole aplikacija u Greenfootu.

    Aktivnosti: Nastavnik učenicima objašnjava zadatak **2.7.** iz projekta **Tower Defense.** Nastavnik nastavlja raditi na zadnjoj verziji projekta, tražeći od učenika da dodaju dvije instance objekta klase **`Enemy`** **`i`** pozovu metodu **`act()`** na svakoj instanci.

Zatim opisuje gumb **`GreenfootAct`** i klikne na njega na glavnom prozoru. Nakon toga navodi učenike da zaključe i objasne što se dogodilo.

Također, nastavnik ih upućuje da kliknu na gumb **`Run`** i da zaključe što se dogodilo.

Potom traži od učenika da kliknu na gumb **`Reset`** i objasne što se dogodilo, a zatim objašnjava što treba napraviti kako bi se svaki put kada se klikne na gumb **`Reset`** dva objekta klase **`Enemy`** pojavila na ploči na pozicijama (0,3) i (3,3).

Nastavnik treba objasniti učenicima da, ako žele da se objekti pojavljuju na ekranu svaki put kada se klikne na gumb **`Reset`**, treba promijeniti konstruktor klase **`World`**, kako bi se ti objekti kreirali u konstruktoru i postavili na željene pozicije.

11. Rasprava: Algoritam, svojstva, algoritmizacija, kontrole aplikacije u Greenfootu

Cilj: Nastavnik sažima lekciju.

Koncepti za raspravu: metode za kretanje, kontrole aplikacije u Greenfootu, dokumentacija.

Aktivnosti: Nastavnik sažima lekciju, pri čemu može naglasiti važnost dokumentacije kôda te pravila za imenovanje metoda i klasa.

Co-funded by the Erasmus+ Programme of the European Union

5. Grananje

Unutar ove tematske jedinice izrađena su dva scenarija poučavanja.

5.1. Istraživanje grananja kroz razvoj igara Greenfootom – nepotpuno grananje kôda

Tablica 7. *Istraživanje grananja kroz razvoj igara Greenfootom – nepotpuno grananje kôda*

Naslov	Istraživanje grananja kroz razvoj igara Greenfootom – nepotpuno grananje kôda
Ishodi učenja	Tema pokriva nepotpuno grananje (namjerno se izostavlja višestruko grananje). Uvode se osnove percepcije svijeta aktera. Učenici će biti sposobni pisati kôd koristeći uvjete. Po završetku ove tematske cjeline učenici će naučiti kako koristiti jednostavne naredbe `if-else` i donositi odluke u svom kôdu kontrolirajući kako se njihova igra ponaša. Steći će osnovno znanje o programskom jeziku Java, naučiti potrebnu sintaksu te analizirati i razumjeti kôd, što će im pomoći da shvate zašto se njihova igra ponaša na određeni način i kako ispraviti probleme. Učenici će moći kreirati vlastite projekte igara koristeći ono što su naučili o objektno orijentiranom programiranju.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujuć, uključujući varijable, funkcije, iteraciju i koncepte selekcije koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Uvod (5 min.) 2) Objašnjenje kôda (15 min.)

	3) Nepotpuno grananje (10 min.) 4) Promatranje stanja igrača (10 min.) 5) Dodavanje detekcije ruba svijeta (10 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHub/GitLab. Internetski resursi.
Opis	Lekcija započinje 15-minutnim objašnjenjem kôda metode `turn()`, što postavlja temelje za razumijevanje nepotpunog grananja kôda. Ovu raspravu nastavnik vodi s ciljem da dovede učenike do zajedničkog razumijevanja uloge metode u svijetu aktera. Nakon toga 10-minutni segment posvećen je stjecanju osnovnih pojmova vezanih uz nepotpuno grananje. Ova faza usvajanja ključna je za učenike kako bi shvatili temeljna načela prije nego što se upuste u pisanje kôda. Zatim učenici sudjeluju u 10-minutnom istraživačkom zadatku, pri čemu promatraju stanje igrača unutar svog kôda, produbljujući razumijevanje toka programa na temelju izvornog kôda. Sljedeća, 10-minutna faza uključuje dodavanje detekcije ruba svijeta u njihov projekt, kao i ponašanje te postavljanje igre.
Procjena	Ova aktivnost omogućit će nastavnicima davanje povratne informacije u obliku formativne procjene na temelju rasprava i praćenja učenika u obrnutoj učionici.

Co-funded by the Erasmus+ Programme of the European Union

	Vrednovanje od strane vršnjaka bit će provedeno online u okviru domaće zadaće. To će podsjetiti učenike na važne aspekte vježbe, omogućiti im kritičko vrednovanje rada drugih učenika, dati im uvid u dobra ili manje dobra rješenja njihovih kolega te će povećati ukupni uspjeh ishoda učenja. Rad na timskom projektu na kojem učenici rade također će koristiti ove ishode učenja i znanja.
Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

5.1.1. Priručnik za nastavnike za pripremu lekcije

1. Uvod

Cilj: Nastavnik razgovara s učenicima o pojmovima koji su obrađeni na prethodnoj i uvodi ciljeve za ovu nastavnu lekciju.

Koncepti za raspravu: algoritam, dokumentacija kôda.

Aktivnosti: U uvodnom dijelu nastavnik zajedno s učenicima pregledava pojmove usvojene na prethodnoj lekciji. Zatim uvodi novi materijal koji će se proučavati na današnjem satu. Kako bi ilustrirao današnju lekciju, nastavnik prikazuje primjer algoritma s jednostavnim grananjem. Jedan od primjera može biti algoritam za prelazak ulice na pješačkom prijelazu gdje nema semafora. Pješak ne prelazi ulicu odmah, već prvo provjerava dolaze li vozila s lijeve ili desne strane. Ako nema vozila, tada prelazi ulicu.

2. **Objašnjenje kôda**

   Cilj: Objasniti učenicima pojedine dijelove kôda.

   Koncepti za raspravu: nije predviđeno za ovu cjelinu.

   Aktivnosti: Nastavnik preuzima najnoviju verziju projekta s: platforme Moodle i repozitorija Git.

   Nastavnik kreira i postavlja objekt klase **`Enemy`** negdje na ploču. Objašnjava neke metode klase **`Actor`**:

   - **`Move(int)`**
   - **`turn(int)`**
   - **`setRotation()`.**

   Dok objašnjava metode, nastavnik također pokazuje kako se mijenjaju određena svojstva klase (na primjer, položaj objekta na ploči, tj. vrijednosti X i Y). Razgovara s učenicima o tome kako dopuniti metodu **`act()`** tako da svaki put kada se pozove metoda **`act()`** objekt klase **`Enemy`** pomakne dva koraka naprijed.

3. **Nepotpuno grananje**

   Koncepti za raspravu: grananje, nepotpuno grananje.

   Aktivnosti: Nastavnik nastavlja raditi na projektu. Postavlja objekt klase **`Enemy`** na ploču.

   Objašnjava učenicima kako mogu provjeriti nalazi li se objekt u gornjoj polovici ploče i prikazuje poruku „Found“. Nastavnik učenicima tumači metodu **`showText()`**, koja se koristi za prikazivanje teksta.

4. **Promatranje stanja igrača**

   Koncepti za raspravu: unutarnje stanje instance.

   Aktivnosti: Nastavnik kreira instancu klase **`Enemy`** i postavlja je u središte ploče. Otvara prozor s unutarnjim stanjem instance i postavlja ga tako da bude vidljiv dok aplikacija radi. Zatim pokreće

Co-funded by the Erasmus+ Programme of the European Union

aplikaciju i promatra kako se vrijednosti atributa X, Y i **`rotation`** u klasi **`Enemy`** mijenjaju prilikom pozivanja različitih metoda. Kako se te vrijednosti mijenjaju kada se kreće (gore, dolje, lijevo i desno) i okreće?

5. **Dodavanje detekcije ruba svijeta**

Cilj: Objasniti što se događa kada objekt dosegne rub svijeta.

Koncepti za raspravu: detekcija ruba svijeta.

Aktivnosti: Nastavnik razgovara s učenicima o tome kako mogu odrediti je li objekt na rubu ili nije. Na primjer, budući da znamo dimenzije ploče, na temelju položaja (X, Y) može se odrediti je li objekt na rubu ploče ili nije.

Nastavnik postavlja instancu negdje u svijetu (ali ne na rub) i poziva metodu **`isAtEdge()`**. Razgovara s učenicima o tome što se dogodilo. Potom premješta neprijatelja na rub i ispituje rezultat metode **`isAtEdge()`**, koja bi sada trebala vratiti **`true.`**

Nastavnik objašnjava metodu **`isAtEdge().`**

Zadatak 3.2. iz projekta **Tower Defense:** Nastavnik učenicima dodjeljuje zadatak da dodaju kôd u tijelo metode **`act()`** kako bi neprijatelja okrenuli za 180° pozivom svojstva **`setRotation()`**, kada dosegne rub svijeta.

Nastavnik razgovara s učenicima o tome kako objekt koji je dosegao rub može nastaviti svoje kretanje:

- unatrag (bez okretanja)
- unatrag (s okretanjem).

Zajedno s učenicima, nastavnik dovršava programski kod metode **`act().`**

Sada pokrećite metodu **`act()`** i razgovarajte o tome što se događa s neprijateljem kada dosegne rub svijeta.

Commit*:* 4927c3ff7eb39b51ba2738f2ab500fd6c32e3bb4

5.2. Istraživanje grananja kroz razvoj igre Greenfootom – potpuno grananje kôda

Tablica 8. *Istraživanje potpunog grananja kroz razvoj igre Greenfootom*

Naslov	Istraživanje potpunog grananja kroz razvoj igre Greenfootom
Ishodi učenja	Ova tema obuhvaća nepotpuno i potpuno grananje (višestruko grananje je namjerno izostavljeno). Uvode se osnove percepcije svijeta Aktera. Učenici će biti sposobni pisati kôd koristeći uvjete. U ovoj tematskoj cjelini učenici će naučiti kako koristiti jednostavne naredbe **`if-else`** te kako donositi odluke u svom kôdu kontrolirajući ponašanje svoje igre. Steći će osnovno znanje programskog jezika Java, naučiti potrebnu sintaksu te analizirati i razumjeti kôdu, što će im pomoći da shvate zašto se njihova igra ponaša na određeni način i kako popraviti probleme. Učenici će biti sposobni kreirati vlastite projekte igara koristeći ono što su naučili o objektno orijentiranom programiranju.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući varijable, funkcije, iteracijske i selekcijske koncepte, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Dodavanje klasa **`Direction`** i **`Orb`** (30 min.) 2) Objašnjenje detekcije sudara (30 min.) 3) Dodavanje detekcije sudara 4) Predviđanje kretanja neprijatelja (1. dio) 5) Predviđanje kretanja neprijatelja (2. dio) 6) Objašnjenje kôda: Potpuno grananje (15 min.) 7) Korištenje potpunog grananja – detekcija sudara (20 min.)

Co-funded by the Erasmus+ Programme of the European Union

	8) Predviđanje kretanja objekata(30 min.) 9) Revizija (5 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHub/GitLaba. Internetski resursi.
Opis	Lekcija započinje 30-minutnom vježbom za dodavanje klasa **`Direction`** i **`Orb`**. Ovo pomaže učenicima da shvate objektno orijentiranu prirodu Jave i važnost pravilnog strukturiranja kôda. Slijedi 20-minutno objašnjenje kôda, fokusirano na koncept detekcije sudara, što pomaže učenicima da razumiju interakcije objekata unutar njihovog okruženja igre. Zatim učenici provode 10 minuta dodajući detekciju sudara svom projektu u još jednom produkcijskom zadatku, koristeći svoje razumijevanje grananja. Potom slijede istraživački zadaci od po 15 minuta, gdje se od učenika traži da predviđaju kretanje neprijatelja u prilagođenom i zahtjevnom okruženju, poboljšavajući svoje vještine rješavanja problema i analitičke sposobnosti. Još jedna 15-minutna rasprava usmjerena je na potpuno grananje, osiguravajući da učenici mogu razlikovati nepotpune i potpune strukture kôda. 20-minutni produkcijski zadatak angažira učenike u korištenju punog grananja s detekcijom sudara, s ciljem konsolidacije njihovog znanja primjenom složenih koncepata u praktičnom okruženju. Lekcija završava izazovnim 30-minutnim istraživačkim zadatkom u kojem učenici ponovno predviđaju kretanje neprijatelja, ovaj put s dodanim iskustvom iz prethodnih zadataka, čime primjenjuju svoje kumulativno znanje.

	Lekcija završava 5-minutnom revizijom teorijskih koncepata.
Ocjenjivanje	Ova aktivnost omogućit će nastavnicima da daju formativnu povratnu informaciju na temelju rasprava i praćenja učeničke obrnute učionice i timskog rada. Online ocjenjivanje od strane vršnjaka bit će provedeno u okviru domaće zadaće. To će učenike podsjetiti na važne aspekte vježbe, omogućit će im kritičku procjenu rada drugih učenika, dati im uvid u dobra ili ne tako dobra rješenja njihovih vršnjaka te povećati ukupno postizanje ishoda učenja. Rad na timskom projektu također će koristiti ove ishode učenja i znanje.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

5.2.1. Priručnik za nastavnike za pripremu lekcije

1. **Dodavanje klasa `Direction i Orb`**

   Cilj: Razumjeti kako dodati novu klasu u projekt.

   Koncepti za raspravu: klasa.

   Aktivnosti: Nastavnik zadaje učenicima zadatak **3.3**. iz projekta **Tower Defense**. Nastavnik prati aktivnosti učenika, a na kraju od jednog od njih traži da predstavi svoj rad. Učenik opisuje i predstavlja svoj rad.

Co-funded by the Erasmus+ Programme of the European Union

Zadatak 3.3.: Stvorite dvije nove klase, potomke klase **`Actor`**. Prva će biti klasa **`Direction`**, a druga **`Orb`**. Pripremite odgovarajuće slike (maks. 50x50 piksela) u grafičkom uređivaču. Zatim dodijelite te slike novostvorenim klasama.

Commit*:* 4ed6b37e6d481181d8b340639aa03391406b6c2e

2. **Rasprava: Objašnjenje detekcije sudara**

   Cilj: Objasniti detekciju sudara.

   Koncepti za raspravu: detekcija sudara.

   Aktivnosti: Nastavnik postavlja instancu klase **`Enemy`** u **`World`** te instancu klase **`Direction`** u isti red. Dodaje kôd u metodu **`act()`** kako bi se objekt pomaknuo jedan korak naprijed.

   Objašnjava učenicima kako odrediti jesu li dva ili više objekata ("likova") u **`Worldu`** na istoj poziciji (na istom polju), a zatim tumači metodu: **`isTouching()`.**

   Nastavnik i učenici modificiraju metodu **`act()`** klase **`Enemy`** kako bi se osiguralo da se neprijatelj okreće za 90° u smjeru kazaljke na satu kada se nalazi na istom polju koje sadrži instancu klase **`Direction`.**

   Zajedno s učenicima, nastavnik promatra što se događa s atributom **`rotation`.**

3. **Dodavanje detekcije sudara**

   Cilj: Razumijevanje učenika kako da odrede jesu li dva objekta na istom polju.

   Koncepti za raspravu: Objasniti da sudar znači trenutak kada se dva objekta na ekranu preklapaju ili nalaze na istom polju.

   Aktivnosti: Nastavnik učenicima dodjeljuje zadatak **3.4**. iz projekta **Tower Defense.** Prati aktivnosti učenika, a na kraju traži od jednog od njih da predstavi svoj rad. Učenik opisuje i predstavlja svoj rad.

Zadatak 3.4.: Dodajte kod u metodu `act()` klase `Enemy` kako biste osigurali da se:

- igrač okrene za 90° suprotno od smjera kazaljke na satu kada uđe u polje koje sadrži instancu klase `Orb`.

Commit*:* 968e6f195e3def25e11bc41b664ba1715f7da11d

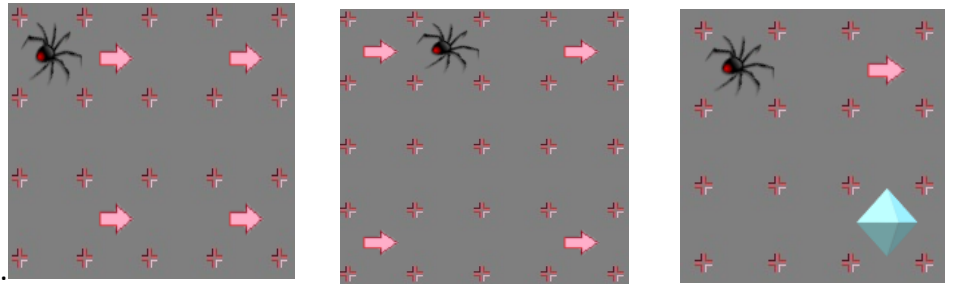
4. **Predviđanje kretanja neprijatelja (1. dio)**

Cilj: Razumjeti kretanje objekata.

Koncepti za raspravu: kretanje objekata.

Aktivnosti: Nastavnik zadaje učenicima zadatak **3.5.** iz projekta **Tower Defense** i prati njihove aktivnosti, a na kraju traži od jednog učenika da predstavi svoj rad. On opisuje i predstavlja svoj rad.

Zadatak 3.5.: Pripremite različite konfiguracije, inspiraciju možete pronaći u donjim slikama. Pogodite kako će se neprijatelj kretati? Pokrenite aplikaciju. Odgovaraju li vaša predviđanja onome što promatrate? Što je uzrokovalo razlike između predviđanja i stvarnosti?



a) b) c)

Slika 1: Konfiguracije prilagođenih postavki instanci za predviđanje kretanja instance klase `Enemy`*.*

5. **Predviđanje kretanja neprijatelja (2. dio)**

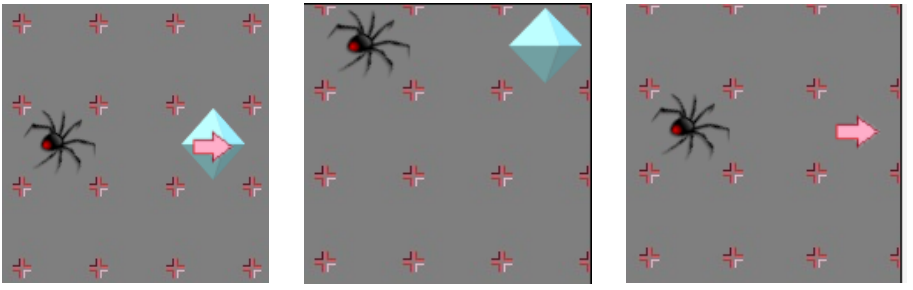
Cilj: Razumijevanje kretanja objekata.

Co-funded by the Erasmus+ Programme of the European Union

Koncepti za raspravu: kretanje objekata.

Aktivnosti: Nastavnik zadaje učenicima zadatak **3.6.** iz projekta **Tower Defense** i prati njihove aktivnosti, a na kraju od jednog od njih traži da predstavi svoj rad, koji to i čini.

Zadatak 3.6.: Pripremite situaciju kao što je prikazano na donjoj slici. Pogodite kako će se neprijatelj kretati? Pokrenite aplikaciju. Odgovaraju li vaša predviđanja onome što promatrate? Što je uzrokovalo razlike između predviđanja i stvarnosti?



a) b) c)

Slika 2: Konfiguracije složenih postavki instanci za predviđanje kretanja instance klase **`Enemy`**.

6. Objašnjenje kôda: Potpuno grananje

Cilj: Poučiti učenike kada i kako koristiti naredbe **`if-then-else`** i **`switch`**.

Koncepti za raspravu: naredba **`if-then-else`**, ugniježđena naredba **`if`**, naredba **`switch`**.

Aktivnosti: Nastavnik traži od učenika da na papiru opišu kako pješak prelazi cestu i prati kako oni rješavaju zadatak. U određenom trenutku nastavnik naglašava učenicima da obrate pažnju na to postoji li na mjestu gdje se prelazi cesta semafor ili ne. Ako u međuvremenu, odnosno prije nego što nastavnik to naglasi, jedan

od učenika postavi to pitanje ili nešto slično, nastavnik ga javno pohvaljuje i naglašava da je prije programiranja važno uvijek prvo analizirati problem i identificirati sve slučajeve koji mogu nastati. Sada bi učenici trebali razumjeti potpuno i ugniježđeno grananje.

Nastavnik će zamoliti učenike da postave objekte klase **`Orb`** i **`Direction`** na lijevu, desnu ili pravu stranu – na različite rubove Svijeta **`(World).`** Učenici će opisati „loše“ ponašanje neprijatelja. Nastavnik bi trebao objasniti da su u jednoj situaciji zadovoljena dva uvjeta. Neprijatelj dodiruje klasu **`Orb/Direction`** i dodiruje rub. Crtanjem situacije na tabli i papiru učenici će prepoznati da su potrebna potpuna i ugniježđena grananja, pa će zatim promijeniti kôd ponašanja neprijatelja.

7. **Korištenje potpunog grananja – detekcija sudara**

   Cilj: Razumjeti potpuno grananje, ugniježđenu naredbu **`if`** i naredbu **`switch`**.

   Koncepti za raspravu: ugniježđena naredba **`if`**, naredba **`switch`** .

   Aktivnosti: Nastavnik daje učenicima zadatak **3.7.** iz projekta **Tower Defense** i prati njihove aktivnosti, a na kraju traži od jednog od njih da predstavi svoj rad. Učenik to i čini.

   Zadatak 3.7.: Promijenite kôd metode **`act()`** klase **`Enemy`** tako da koristi potpuno grananje. Kreirat ćete ugniježđene uvjete. Neka detekcija ruba bude najvažnija provjera, zatim provjerite dodir s instancom klase **`Direction`**, a na kraju provjerite dodir s instancom klase **`Orb`**.

Commit*:* f017de8b49d4fc77f62afac4d842429560bcfb8b

8. **Predviđanje kretanja objekata**

   Cilj: Predvidjeti kretanje neprijatelja u prilagođenim postavkama.

   Koncepti za raspravu: ponašanje objekata.

Co-funded by the Erasmus+ Programme of the European Union

Aktivnosti: Nastavnik daje učenicima zadatak **3.8.** iz projekta **Tower Defense**. Nastavnik postavlja objekte proizvoljno u Svijet, a učenici objašnjavaju njihovo kretanje i ponašanje (samostalno ili u parovima).

Zadatak 3.8.: Ponovite zadatke 3.5. i 3.6. Što se promijenilo?

9. Revizija

Cilj: Sažeti lekciju.

Koncepti za raspravu: koncept grananja.

Aktivnosti: Nastavnik sažima lekciju.

6. Varijable i izrazi

Unutar ove tematske jedinice stvoreno je pet scenarija poučavanja.

6.1. Uvod u varijable i tipove podataka u okruženju Greenfoot

Tablica 9. *Uvod u varijable i tipove podataka u okruženju Greenfoot*

Naslov	Uvod u varijable i tipove podataka u okruženju Greenfoot
Ishodi učenja	Na kraju ove lekcije učenici će moći razumjeti tipove podataka i varijable. Razumijevanje obrađenih koncepata raspravljat će se u kontekstu razvoja igara, čime se potiče kreativnost, timski rad i entuzijastičan pristup kodiranju koristeći alat Greenfoot.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući varijable i tipove podataka, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Uvod (10 min.) 2) Identifikacija varijabli (5 min.) 3) Tipovi podataka (15 min.) 4) Deklaracija varijabli (10 min.) 5) Inicijalizacija varijabli (5 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHub/GitLaba.

Co-funded by the Erasmus+ Programme of the European Union

	Internetski resursi.
Opis	U ovom 45-minutnom scenariju poučavanja učenici srednjih škola uvode se u programske principe povezane s tipovima podataka i varijablama kroz prizmu razvoja igara koristeći alat Greenfoot. Lekcija započinje 10-minutnim nastavnikovim uvodom, kojim se uspostavlja kontekst povezan s prethodnim lekcijama, što stvara osnovu za uvođenje i definiranje varijabli. Nakon toga slijedi 5-minutni scenarij tijekom kojeg učenici i nastavnici istražuju i identificiraju varijable za svoju igru. Uzimajući u obzir da svaka varijabla mora biti određenog tipa, u sljedećih 15 minuta bit će predstavljeni različiti tipovi podataka. Ključne aktivnosti uključuju 10-minutnu cjelinu pod vodstvom nastavnika tijekom koje učenici deklariraju varijable za svoju igru, naglašavajući važnost imena i tipova varijabli. Zatim slijedi 5-minutni dio vezan uz inicijalizaciju varijabli, u kojoj se definiraju vrijednosti varijabli. U ovom kontekstu ponašanje objekata u igri može se promijeniti (npr. rotacija objekta, kretanje objekta). Učenici će nastaviti raditi na igri koja je objašnjena i započeta u prethodnim lekcijama. Kao rezultat, na kraju lekcije uvode se novi koncepti povezani s varijablama i tipovima podataka.

Ocjenjivanje	Igrifikacija predstavlja neformalnu procjenu, ali će povećati interes, unutarnju motivaciju i rezultate učenja cijele grupe.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

6.1.1. Priručnik za nastavnike za pripremu lekcije

1. **Uvod**

   U uvodnom dijelu uspostavlja se kontekst povezan s prethodnim lekcijama. Nastavnik uvodi pojam varijable.

2. **Identifikacija varijabli**

   Cilj: Identifikacija varijabli kroz raspravu, s naglaskom na ulogama varijabli u programiranju.

   Koncepti za raspravu: varijable i vrijednosti.

   Aktivnosti: Nastavnik uvodi pojam varijable i potiče učenike da istraže i identificiraju varijable za svoju igru, o kojima zatim mogu raspravljati i nastavnik i učenici. U ovom scenariju tip varijable može biti izostavljen (ili općenito raspravljan).

3. **Tipovi podataka**

   Cilj: Razumijevanje koncepta tipova podataka, prepoznavanje njihove primjene u stvarnom svijetu, s naglaskom na tipovima varijabli potrebnima za razvoj igara.

   Koncepti za raspravu: varijable, tipovi podataka, primjeri tipova podataka iz stvarnog svijeta, tipovi varijabli za igru.

Co-funded by the Erasmus+ Programme of the European Union

Aktivnosti: Nastavnik uvodi pojam tipa podataka, potičući raspravu o primjerima iz stvarnog svijeta (npr. cijeli brojevi mogu se odnositi na broj prisutnih učenika, decimalni brojevi mogu se odnositi na cijenu proizvoda, tekstualni tip može se odnositi na tekst trenutnih poruka itd.). Tipovi podataka razmatraju se u kontekstu okruženja Greenfoot i programskog jezika Java, a zatim slijedi detaljna rasprava vezana uz tipove varijabli potrebnih za igru.

4. Deklaracija varijabli

Cilj: Primjena koncepta tipova podataka i varijabli te deklaracija varijabli potrebnih za razvoj igara.

Koncepti za raspravu: varijable, tipovi podataka, tipovi varijabli za igru.

Aktivnosti: Tipovi podataka razmatraju se u kontekstu okruženja Greenfoot i programskog jezika Java. Nastavnik bi trebao objasniti razliku između deklaracije i inicijalizacije varijabli: pri deklaraciji varijable deklarira se varijabla određenog tipa podataka, a vrijednost može (ili ne mora) biti prisutna. Može se uvesti analogija (npr. označena kutija za određeni tip keksa, ali bez keksa u toj kutiji). Potom se deklariraju varijable potrebne za igru, uz razmatranje dodatnih primjera. Npr., ako se razmatra metoda `act()`, može se deklarirati varijabla za prikaz teksta.

5. Inicijalizacija varijabli

Cilj: Primjena koncepta tipova podataka i varijabli te inicijalizacija varijabli potrebnih za igru.

Koncepti za raspravu: varijable, tipovi podataka, vrijednosti varijabli za igru.

Aktivnosti: Na temelju prethodno predstavljenih tipova podataka uvode se vrijednosti podataka i rasponi podataka. Te vrijednosti i rasponi podataka razmatraju se u kontekstu okruženja Greenfoot i programskog jezika Java. Nastavnik bi trebao objasniti razliku

između deklaracije i inicijalizacije varijabli: pri inicijalizaciji varijable deklarira se varijabla određenog tipa podataka i može se inicijalizirati u isto vrijeme (to se zove: početna vrijednost), a može se mijenjati u sljedećem kôdu. Pritom se može se uvesti analogija (npr. keks određenog tipa keksa stavlja se u prethodno definiranu označenu kutiju). Zatim slijedi inicijalizacija varijabli potrebnih za igru. Mogu se razmotriti dodatni primjeri. Npr., ako se razmatra metoda **`act()`**, može se inicijalizirati varijabla za prikaz teksta.

6.2. Uvod u operatore i izraze u okruženju Greenfoot

Tablica 10. *Uvod u operatore i izraze u okruženju Greenfoot*

Naslov	Uvod u operatore i izraze u okruženju Greenfoot
Ishodi učenja	Do kraja ove nastavne cjeline učenici će moći razumjeti koncept operatora. Lekcija predstavlja različite tipove operatora (npr. aritmetičke operatore, Booleove operatore, relacijske operatore), kao i njihove odgovarajuće izraze. Dodatno će se uvesti objektni izraz i referentna varijabla. O ispitanim konceptima raspravljat će se u kontekstu razvoja igre, poticanja kreativnosti, timskog rada i entuzijastičnog pristupa programiranju u okruženju Greenfoot.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući koncepte poput iteracije i odabira, koji poznaju i okruženje Greenfoot.

Co-funded by the Erasmus+ Programme of the European Union

Trajanje scenarija	1) Operatori (15 min.) 2) Aritmetički operatori i izrazi (10 min.) 3) Booleovi operatori (15 min.) 4) Relacijski operatori (10 min.) 5) Booleovi izrazi (10 min.) 6) Objektni izrazi (5 min.) 7) Referentna varijabla i njezina nulta vrijednost (15 min.) 8) Korištenje varijabli, tipova podataka i operatora za upravljanje objektima u igri (15 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	Tijekom ove 95-minutne cjeline srednjoškolci će se upoznati s programskim konceptima vezanim uz operatore i izraze u kontekstu razvoja igara pomoću alata Greenfoot. Lekcija započinje 15-minutnim uvodom nastavnika, koji izlaže pozadinske informacije iz prethodnih lekcija i postavlja temelje za uvođenje i definiciju operatora. Zatim slijedi 10-minutni dio u kojem se raspravlja o aritmetičkim operatorima i izrazima koji se koriste za aritmetičke operacije. U tom se kontekstu praktično objašnjavaju i raspravljaju različiti operatori i izrazi unutar okruženja Greenfoot. Sljedećih 15 minuta posvećeno je Booleovim operatorima, logičkim operatorima koji se koriste za upravljanje vrijednostima **`boolean`**. Nakon toga u 10 minuta obrađuju se relacijski operatori koji služe za usporedbu vrijednosti. Na temelju prethodnih dijelova, u narednih 10 minuta raspravlja se o Booleovim izrazima u kontekstu Greenfoota. U sljedećih 5 minuta lekcija fokusira se na

	izražavanje objekata, dok se u narednih 15 minuta raspravlja o referentnim varijablama. Konačno, slijedi 15 minutni dio pod vodstvom nastavnika, tijekom kojeg učenici rješavaju zadatak vezan uz igru „Okreni se u smjeru“. U ovu lekciju uključene su povratne informacije koje su iznijeli i učenici i nastavnik. Učenici će potom nastaviti s radom na projektu igre započetim u prethodnim lekcijama, pa će do kraja predavanja biti upoznati s pojmovima vezanim uz operatore i izraze.
Ocjenjiva-nje	Igrifikacija predstavlja neformalno ocjenjivanje, ali povećava interes, intrinzičnu motivaciju i ishode učenja cijele grupe. U ovom stadiju projekta otvaraju se mogućnosti za domaće zadatke. U tom kontekstu mogu se uvesti dodatne klase, izrazi i vrijednosti kako bi se postigla nova ponašanja (npr. teleporti, tuneli itd.). O tim konceptima može se razgovarati s učenicima, a njihova implementacija može se zadati kao domaća zadaća.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

6.2.1. Priručnik za nastavnike za pripremu lekcije

1. Operatori

Cilj: Razumijevanje koncepta operatora, njihovo povezivanje sa scenarijima iz stvarnog života i učenje o različitim vrstama operatora.

Co-funded by the Erasmus+ Programme of the European Union

Koncepti za raspravu: varijable, tipovi podataka, operatori, primjeri operatora iz stvarnog svijeta.
Aktivnosti: Nastavnik uvodi pojam operator, a ovaj koncept trebao bi približiti učenicima korištenjem primjera iz stvarnog života (npr. kupnja proizvoda na tržnici), nakon čega nastavnik predstavlja različite vrste operatora.

2. Aritmetički operatori i izrazi

Cilj: Razumijevanje koncepta aritmetičkih operatora, njihovo povezivanje sa scenarijima iz stvarnog života i učenje o različitim aritmetičkim operatorima.

Koncepti za raspravu: varijable, tipovi podataka, operatori, primjeri aritmetičkih operatora iz stvarnog svijeta.

Aktivnosti: Nastavnik može objasniti operatore koji su već poznati iz drugih predmeta (npr. matematika i matematički aritmetički operatori), u kontekstu okruženja Greenfoot i programskog jezika Java. Raspravlja se o različitim pojmovima: operator, operand, prioritet operatora, a u obzir se mogu uzeti i drugi primjeri. Dodatni primjer može uključivati definiranje lokalnih varijabli za dohvaćanje i manipuliranje X-položaja i Y-položaja entiteta, čime se mijenja njegov položaj povećanjem vrijednosti varijable.

3. Booleovi operatori

Cilj: Razumijevanje koncepta Booleovih operatora, njihovo povezivanje sa stvarnim životnim scenarijima i učenje o raznim Booleovim operatorima.

Koncepti za raspravu: varijable, tipovi podataka, operatori, primjeri Booleovih operatora iz stvarnog svijeta.

Aktivnosti: Nastavnik može objasniti operatore koji su već poznati iz drugih predmeta (npr. matematika i Booleovi operatori), u kontekstu okruženja Greenfoot i programskog jezika Java. Raspravlja se o različitim pojmovima, kao što su: operator, operand, prioritet operatora, a dodatni primjeri mogu uključivati definiranje lokalnih varijabli za provjeru je li X-položaj entiteta

jednak njegovom Y-položaju, koristeći Booleov operator za određivanje je li entitet na dijagonali.

4. **Relacijski operatori**

   Cilj: Razumijevanje koncepta relacijskih operatora, njihovo povezivanje sa scenarijima iz stvarnog života i učenje o raznim relacijskim operatorima.

   Koncepti za raspravu: varijable, tipovi podataka, operatori, primjeri operatora iz stvarnog svijeta.

   Aktivnosti: Nastavnik može objasniti operatore koji su već poznati iz drugih predmeta (npr. matematike i matematičkih relacijskih operatora), u kontekstu okruženja Greenfoot i programskog jezika Java. U raspravu o različitim pojmovima, kao što su : operator, operand, prioritet operatora, mogu se ubaciti dodatni primjeri definiranja lokalnih varijabli za provjeru je li Y-pozicija jednog entiteta ispod Y-pozicije drugog entiteta, koristeći relacijske operatore za određivanje pozicijskih odnosa između entiteta.

5. **Booleovi izrazi**

   Cilj: Razumijevanje koncepta Booleovih izraza, njihovo povezivanje sa scenarijima iz stvarnog života i učenje o raznim Booleovim izrazima.

   Koncepti za raspravu: varijable, tipovi podataka, operatori, primjeri Booleovih izraza iz stvarnog svijeta.

   Aktivnosti: Nastavnik može objasniti Booleove izraze u kontekstu prethodno predstavljenih operatora i u okruženju Greenfoot i programskog jezika Java. Raspravljajući o operatorima, operandima i prioritetu operatora u kontekstu Booleovih izraza, mogu se razmotriti i dodatni primjeri, poput korištenja Booleovih izraza za provjeru je li pozicija entiteta unutar definirane dimenzije arene u igri.

Co-funded by the Erasmus+ Programme of the European Union

6. **Objektni izrazi**

   Cilj: Razumijevanje koncepta izraza objekta, njihovo povezivanje sa scenarijima iz stvarnog života i učenje o različitim izrazima objekta.

   Koncepti za raspravu: varijable, tipovi podataka, operatori, primjeri objektnih izraza iz stvarnog svijeta.

   Aktivnosti: Nastavnik može objasniti izraze objekta u kontekstu objektno orijentiranog dizajna, a u kontekstu okruženja Greenfoot i programskog jezika Java. Raspravlja se o operatorima, operandima, prioritetu operatora i odabiru klasa u kontekstu objektnih izraza. Mogu se istražiti dodatni primjeri, poput usporedbe referenci dvaju objekata kako bi se provjerilo preklapaju li se.

7. **Referentna varijabla i njezina nulta vrijednost**

   Cilj: Razumijevanje koncepta referentnih varijabli, njihovo povezivanje sa scenarijima iz stvarnog života i njihova primjena u razvoju igara.

   Koncepti za raspravu: varijable, tipovi podataka, operatori, primjeri referentnih varijabli iz stvarnog svijeta.

   Aktivnosti: Nastavnik može objasniti referentne varijable u kontekstu objektno orijentiranog dizajna, i to u kontekstu okruženja Greenfoot i programskog jezika Java, a zatim treba protumačiti i pojam nulte referentne vrijednosti.

8. **Korištenje varijabli, tipova podataka i operatora za upravljanje objektima u igri**

   Cilj: Razumijevanje koncepta varijabli, tipova podataka, operatora i izraza te njihovo korištenje u razvoju igre.

   Koncepti za raspravu: varijable, tipovi podataka, operatori, izrazi.

   Aktivnosti: Nastavnik daje učenicima zadatak **4.1.** iz projekta **Tower Defense** („Okreni se u smjeru“). Nastavnik bi trebao

raspraviti metodu **`act()`** klase **`Enemy`** – neprijatelj – i protumačiti kako koristiti lokalne varijable u kôdu, primjerice varijablu „rotacija“, Mora objasniti razliku između **`this.rotation`** rotacije i **`rotation`** (rotacije općenito) te opisati ponašanje metode **`getOneIntersectingObject(_cls_)`**, koja koristi instancu i pohranjuje je u odgovarajuću lokalnu varijablu (uz kasting klase). Ako ne postoji objekt koji se presijeca, metoda vraća nultu vrijednost. Na temelju izvršene Booleove procjene provodi se odgovarajuće djelovanje (npr. rotiranje ili okretanje). O rezultatima raspravljaju i nastavnik i učenici.

Commit*:* 97dddc4beba40ac785c7413bb245ba849cd956d2

6.3. Uvod u konstruktore u okruženju Greenfoot

Tablica 11. *Uvod u konstruktore u okruženju Greenfoot*

Naslov	Uvod u konstruktore u okruženju Greenfoot
Ishodi učenja	Do kraja ove lekcije učenici će moći razumjeti koncept konstruktora, jer sadrži osnovne teorijske pojmove vezane uz konstruktore, kao i različita objašnjenja kôda i zadatke. O razmatranim konceptima raspravljat će se u kontekstu razvoja igre, poticanja kreativnosti, timskog rada i entuzijastičnog pristupa kodiranju u Greenfootu.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući koncepte poput iteracije i odabira, koji poznaju i okruženje Greenfoot.

Co-funded by the Erasmus+ Programme of the European Union

Trajanje scenarija	1) Osnovni pojmovi o konstruktorima (10 min.) 2) Objašnjenje kôd (20 min.) 3) Promjena naziva klase (5 min.) 4) Izrada izgleda arene (30 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internet resursi.
Opis	Tijekom ove 65-minutne lekcije srednjoškolci se upoznaju s konceptima vezanim uz konstruktore u kontekstu razvoja igara pomoću alata Greenfoot. Lekcija počinje 10-minutnim uvodom nastavnika, koji izlaže osnovne teorijske koncepte konstruktora. Nakon toga slijedi 20-minutni dio tijekom kojeg se praktično objašnjavaju i raspravljaju različiti primjeri kôda u Greenfootu. Sljedeći, 5-minutna dio odnosi se na zadatak. Prethodno definiranu klasu **`MyWorld`** treba preimenovati u **`Arena`**. Važno je napomenuti da se i konstruktor klase također treba preimenovati. Na kraju slijedi 30-minutni dio pod vodstvom nastavnika, tijekom kojeg učenici rješavaju zadatak vezan uz izgled arene. Povratne informacije koje su dali nastavnik i vršnjaci uključene su u ovu lekciju. Učenici će nastaviti s radom na projektu igre započetim u prethodnim lekcijama. Stoga će se do kraja lekcije upoznati s pojmovima vezanim uz konstruktore.
Ocjenjivanje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe. I u ovom dijelu projekta otvaraju se mogućnosti za domaće zadatke. U tom kontekstu mogu se

	uvesti dodatne klase, izrazi i vrijednosti kako bi se postiglo dodatno ponašanje (npr. teleporti, tuneli itd.). Ovi se koncepti mogu raspraviti s učenicima, a dotična implementacija može se zadati kao domaća zadaća.
Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

6.3.1. Priručnik za nastavnike za pripremu lekcije

1. **Osnovni koncepti konstruktora**

   Cilj: Razumijevanje koncepta konstruktora.

   Koncepti za raspravu: konstruktori, klase, objekti; ključne riječi: `super, new, this.`

   Aktivnosti: Nastavnik uvodi pojam konstruktora u kontekstu koncepata klase i objekta u objektno orijentiranom programiranju. Konstruktori se koriste za inicijalizaciju konkretne instance (tj. objekta) klase.

2. **Objašnjenje kôda**

   Cilj: Razumijevanje koncepta konstruktora i povezivanje sa scenarijima iz stvarnog života.

   Koncepti za raspravu: konstruktori, klase, objekti, metode, parametri; ključne riječi: `super, new, this`, primjeri konstruktora iz stvarnog svijeta.

   Aktivnosti: Nastavnik bi trebao pojasniti konstruktore u kontekstu OOP koncepata klase i objekta. Konstruktori se koriste za inicijalizaciju konkretnih instanci klase, a uvijek se pozivaju i mogu

Co-funded by the Erasmus+ Programme of the European Union

se definirati implicitno ili eksplicitno. Postoje zadani konstruktori (koji su implicitno definirani) te parametrizirani i neparametrizirani konstruktori (koji se eksplicitno definiraju). Također treba raspraviti razlike između parametriziranih i neparametriziranih konstruktora. Da bi ovaj koncept bio bliži učenicima, nastavnik bi trebao koristiti primjere iz stvarnog života.

3. **Promjena naziva klase**

   Cilj: Preimenovanje klase **`MyWorld.`**

   Koncepti za raspravu: konstruktori, klase, objekti.

   Aktivnosti: Nastavnik daje učenicima zadatak 4.2. iz projekta Tower Defense. Prethodno definiranu klasu **`MyWorld`** treba preimenovati u klasu **`Arena.`** Također, konstruktor klase treba preimenovati iz **`MyWorld()`** u **`Arena().`**

   Commit: aaf73c9bfd9f76a2a1e504f5e78d2976f1cada12

4. **Izrada izgleda arene**

   Cilj: Razumijevanje koncepta konstruktora i njihovo povezivanje sa scenarijem igre.

   Koncepti za raspravu: konstruktori, klase, objekti, metode, parametri; ključne riječi*:* **`super, new, this`**.

   Aktivnosti: Nastavnik daje učenicima zadatak **4.3.** iz projekta **Tower Defense.** U ovoj fazi treba izraditi prilagođeni izgled za arenu, tako da bude definiran unutar konstruktora klase **`Arena.`** Potrebno je dodati jednu instancu **`Enemy`**, jednu instancu **`Orb`** i barem jednu instancu **`Direction`**. Nakon deklariranja i inicijaliziranja varijabli, svojstva se trebaju dodijeliti pozivanjem odgovarajućih metoda. Na kraju ovi objekti trebaju biti uključeni u arenu pozivanjem metode **`addObject(Actor).`**

   Commit: 8b105ea2eaf697f08c321efe687ddd31e2d0a041

6.4. Uvod u atribute u okruženju Greenfoot

Tablica 12. *Uvod u atribute u okruženju Greenfoot*

Naslov	Uvod u atribute u okruženju Greenfoot
Ishodi učenja	Do kraja ove lekcije učenici će moći razumjeti koncept atributa. U njoj se uvode osnovne teorijske pojmove vezane uz atribute, kao i različita objašnjenja kôda te zadaci. O razmatranim konceptima raspravljat će se u kontekstu razvoja igre, poticanja kreativnosti, timskog rada i entuzijastičnog pristupa kodiranju u Greenfootu.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući osnovno objektno orijentirano znanje, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Problem i rješenje vezano uz kretanje objekata (30 min.) 2) Atributi (10 min.) 3) Parametri konstruktora (10 min.) 4) Atribut **`Enemy.moveDelay`** (20 min.) 5) Kretanje neprijatelja poštujući odgodu (30 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	Tijekom ove 100-minutne lekcije srednjoškolci će se upoznati s konceptima vezanim uz atribute u kontekstu razvoja igara pomoću alata Greenfoot. Lekcija počinje 30-

Co-funded by the Erasmus+ Programme of the European Union

	minutnim zadatkom koji se odnosi na probleme vezane uz kretanje i potencijalna rješenja. Na temelju prethodnog zadatka, u sljedećoj cjelini od 10 minuta uvode se pojmovi vezani uz atribute. Nakon toga slijedi 10-minutni segment u kojem se objašnjavaju i raspravljaju parametri konstruktora. U sljedećem, 20-minutnom dijelu pod vodstvom nastavnika, usredotočuje se na zadatak koji uključuje definiciju novog atributa za kretanje u klasi neprijatelja, kao i definiciju parametarskog konstruktora. Povratne informacije nastavnika i vršnjaka bit će uključene u ovaj dio. Konačno, u posljednjoj, 30-minutnoj cjelini pod vodstvom nastavnika, implementira se kretanje neprijatelja. U ovom kontekstu metoda **`act()`** bit će ažurirana, a povratne informacije nastavnika i vršnjaka također će biti uključene u ovu lekciju. Učenici će nastaviti rad na projektu igre započetom u prethodnim lekcijama. Shodno tome, do kraja predavanja upoznat će se s pojmovima vezanim uz atribute.
Ocjenjiva-nje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe. I u ovom dijelu projekta otvaraju se mogućnosti za domaće zadatke. U tom kontekstu mogu se uvesti dodatne klase, izrazi i vrijednosti kako bi se postiglo dodatno ponašanje (npr. teleporti, tuneli itd.). Ovi se koncepti mogu raspraviti s učenicima, a dotična implementacija može se zadati kao domaća zadaća.

Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

6.4.1. Priručnik za nastavnike za pripremu lekcije

1. **Problem i rješenje vezano uz kretanje objekta**

   Cilj: Razumijevanje problema vezanih uz kretanje i mogućih rješenja u kontekstu konstruktora i atributa.

   Koncepti za raspravu: konstruktori, atributi, metode.

   Aktivnosti: Nastavnik učenicima zadaje zadatak 4.4 iz projekta **Tower Defense**. Nastavnik bi trebao objasniti da neprijatelj trenutno pomiče dvije ćelije odjednom, što uzrokuje probleme s njegovim kretanjem. Da bi se to riješilo, brzina neprijatelja može se drugačije modelirati. Instanca neprijatelja sada će se uvijek pomicati ćeliju po ćeliju. Dodatno, može se definirati novi atribut pod nazivom `moveDelay`, koji će uzrokovati pomicanje instance neprijatelja tek nakon što prođe određeni broj poziva metode `act()`.

2. **Atributi**

   Cilj: Razumijevanje koncepta atributa.

   Koncepti za raspravu: konstruktori, klase, objekti, atributi.

   Aktivnosti: Nastavnik uvodi koncept atributa u kontekstu klasa i objekata u objektno orijentiranom programiranju.

3. **Parametri konstruktora**

   Cilj: Razumijevanje pojma parametara konstruktora.

Co-funded by the Erasmus+ Programme of the European Union

Koncepti za raspravu: konstruktori, klase, objekti, atributi, parametri; ključne riječi: **`super, new, this.`**

Aktivnosti: Nastavnik uvodi koncept parametara konstruktora u kontekstu klasa i objekata u objektno orijentiranom programiranju.

4. **Atribut** Enemy.moveDelay

Cilj: Razumijevanje koncepta atributa i parametara konstruktora.

Koncepti za raspravu: konstruktori, klase, objekti, atributi, parametri; ključne riječi: **`super, new, this.`**

Aktivnosti: Nastavnik učenicima zadaje zadatak **4.5.** iz projekta **Tower Defense**. Novi atribut nazvan **`moveDelay`** tipa **`int`** bit će dodan u klasu **`Enemy`**. Također će se definirati parametarski konstruktor za inicijalizaciju ovog atributa, pri čemu će atribut biti postavljen na vrijednost koju pruža parametar. Kôd u klasi **`Arena`** bit će odgovarajuće prilagođen.

Commit: 6092489ce57541e77ae4e2ee886b20853df9f8a4

5. **Kretanje neprijatelja poštujući odgodu**

Cilj: Razumijevanje koncepta atributa i parametara konstruktora.

Koncepti za raspravu: konstruktori, klase, objekti, atributi, parametri; ključne riječi: **`super, new, this.`**

Aktivnosti: Nastavnik učenicima zadaje zadatak **4.6.** iz projekta **Tower Defense**. Metoda **`act()`** klase **`Enemy`** bit će ažurirana tako da se neprijatelj pomiče tek nakon određenog broja poziva metode **`moveDelay`**. Također će se uvesti novi atribut nazvan **`nextMoveCounter`** tipa **`int`**, koji će biti inicijaliziran na 0. Metoda **`act()`** će biti modificirana tako da poziva **`this.move(1)`** samo kada **`nextMoveCounter`** dosegne 0. Nakon pomicanja, **`nextMoveCounter`** će biti resetiran na vrijednost **`moveDelay`**. Ako instanca **`Enemy`** ne

može pomaknuti neprijatelja jer **`nextMoveCounter`** još nije postigao 0, **`nextMoveCounter`** će se smanjiti za 1.

Commit: bf26e6ed23911ccb712fae3e243cdedff3a89a7f

6.5. Uvod u preopterećenje konstruktora u okruženju Greenfoot

Tablica 13. *Uvod u preopterećenje konstruktora u okruženju Greenfoot*

Naslov	Uvod u preopterećenje konstruktora u okruženju Greenfoot
Ishodi učenja	Do kraja ove lekcije učenici će moći razumjeti koncept preopterećenja konstruktora. Lekcija uvodi osnovne teorijske koncepte vezane uz preopterećenje konstruktora, uključujući različita objašnjenja kôda i zadatke. Razmatrani koncepti raspravljat će se u kontekstu razvoja igre, poticanja kreativnosti, timskog rada i entuzijastičnog pristupa kodiranju u Greenfootu.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući koncepte poput iteracije i odabira, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Osnovni koncepti preopterećenja konstruktora (5 min.) 2) Parametarski konstruktor klase **`Direction`** (25 min.) 3) Preopterećenje konstruktora u klasi **`Direction`** (25 min.) 4) Revizija teorije (20 min.)

Co-funded by the Erasmus+ Programme of the European Union

Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	Tijekom ove 75-minutne lekcije učenici će se upoznati s konceptima koji se odnose na preopterećenje konstruktora u kontekstu razvoja igre pomoću alata Greenfoot. Lekcija počinje 5-minutnim uvodom u osnovne koncepte preopterećenja konstruktora. Sljedeći, 25-minutni dio, pod vodstvom nastavnika, odnosi se na zadatak u kojem se definira parametarski konstruktor u klasi **`Direction`**. Nakon toga slijedi zadatak od 25 minuta u kojem se definira preopterećeni konstruktor u istoj klasi. Povratne informacije koje su dali nastavnik i vršnjaci uključene su u ove cjeline. U posljednjoj 20-minutnoj cjelini, pod vodstvom nastavnika, provodi se revizija teorije koja se odnosi na prethodno raspravljene koncepte (tj. varijable, izraze, operatore, konstruktore, atribute i preopterećenje konstruktora).
Ocjenjiva-nje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe. I u ovom dijelu projekta otvaraju se mogućnosti za domaće zadatke. U tom kontekstu mogu se uvesti dodatne klase, izrazi i vrijednosti kako bi se postiglo dodatno ponašanje (npr. teleporti, tuneli itd.). Ovi se koncepti mogu raspraviti s učenicima, a dotična implementacija može se zadati kao domaća zadaća.

Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem

6.5.1. Priručnik za nastavnike za pripremu lekcije

1. **Osnovni koncepti preopterećenja konstruktora**

   Cilj: Razumijevanje koncepta preopterećenja konstruktora.

   Koncepti za raspravu: konstruktori.

   Aktivnosti: Raspravlja se o konceptima preopterećenja konstruktora.

2. **Parametarski konstruktor klase Direction**

   Cilj: Definiranje parametarskog konstruktora klase **`Direction`** u kontekstu razvoja igre.

   Koncepti za raspravu: konstruktori, parametri, atributi, parametarski konstruktori.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **4.7.** iz projekta **Tower Defense**. U ovom zadatku definira se parametarski konstruktor za klasu **`Direction`** s jednim parametrom**`, rotation`**, tipa **`int`**. Unutar tijela konstruktora, stvorena instanca treba biti rotirana na temelju vrijednosti ovog parametra. Kôd u klasi **`Arena`** treba biti ažuriran u skladu s tim.

Commit: 3c4b9ef57ab17bac2a0abc7fc5e76ea4b6e27e4b

3. **Preopterećenje konstruktora u klasi** Direction

   Cilj: Definiranje preopterećenog konstruktora klase **`Direction`** u kontekstu razvoja igre.

   Koncepti za raspravu: konstruktori, preopterećenje konstruktora.

Co-funded by the Erasmus+ Programme of the European Union

Aktivnosti: Nastavnik učenicima zadaje zadatak **4.8.** iz projekta **Tower Defense**. U ovom zadatku definira se preopterećeni konstruktor u klasi **`Direction`**. Dodaje se konstruktor bez parametara, a unutar njegova tijela poziva se parametarski konstruktor s argumentom **`rotation`** postavljenim na 0. Kôd u klasi **`Arena`** treba biti ažuriran u skladu s tim, koristeći verziju konstruktora klase **`Direction`** bez parametara gdje je to moguće.

Commit: 1e67e67523c66acea4e93363c9a3173302f424c8

4. **Revizija teorije**

Cilj: Revizija teorije vezane uz prethodno raspravljane koncepte.

Koncepti za raspravu: varijable, izrazi, operatori, konstruktori, atributi, preopterećenje konstruktora.

Aktivnosti: Tijekom ove lekcije provedena je revizija prethodno raspravljanih koncepata.

7. Varijable i asocijacije

U ovoj tematskoj cjelini izrađena su četiri nastavna scenarija.

7.1. Greenfootovi objekti na zadatku – upoznavanje s metodama i asocijacijama

Tablica 14. *Greenfootovi objekti na zadatku – upoznavanje s metodama i asocijacijama*

Naslov	Greenfootovi objekti na zadatku – upoznavanje s metodama i asocijacijama
Ishodi učenja	Na kraju ove nastavne cjeline učenici bi trebali steći solidno razumijevanje načina na koji objekti mogu međusobno komunicirati. Primjer klase **`Enemy`** interagira s drugim objektima, posebno s klasom **`Orb`**, unutar Greenfoota. Trebali bi pokazati vještinu u stvaranju i pozivanju metoda unutar klasa u Javi, posebno implementirajući i testirajući metode **`Arena.respawn(Enemy)`** i **`Orb.hit(Enemy)`**. Osim toga, moći će razumjeti i učinkovito upravljati atributima klasa, uključujući definiranje i korištenje atributa **`Enemy.attack`** i **`Orb.hp`**. Trebali bi biti vješti u enkapsulaciji podataka unutar klase, što se demonstrira stvaranjem getter metoda, kao što je **`getEnemy.attack()`**, te razumjeti važnost enkapsulacije podataka za sigurni i održivi kôd. Učenici će moći razumjeti i implementirati prijenos poruka između objekata, osiguravajući da instance klasa učinkovito komuniciraju kako bi izvele radnje u igri. Nadalje, znat će primijeniti tehnike pozivanja metoda za rješavanje

Co-funded by the Erasmus+ Programme of the European Union

	interaktivnih zadataka u razvoju igre, učinkovito koristeći sintaksu i parametre potrebne za pozivanje metoda.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući koncepte poput iteracije i odabira, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Raspraviti što bi se trebalo dogoditi kada neprijatelj dosegne **`Orb`** (10 min.) 2) Komunikacija instance klase **`Enemy`** s objektima pri udaru u **`Orb`** (15 min.) 3) Atributi **`Enemy.attack i Orb.hp`** (10 min.) 4) Metoda (15 min.) 5) Getter atributa **`Enemy.attack`** (5 min.) 6) Kreirati i testirati metodu **`Arena.respawn(Enemy)`** (10 min.) 7) Kreirati i testirati metodu **`Orb.hit(Enemy)`** (10 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	U ovoj 75-minutnoj lekciji učenici će se usredotočiti na interakciju objekata, kreiranje metoda i upravljanje atributima unutar okruženja Greenfoot. Cilj lekcije je unaprijediti razumijevanje učenika o tome kako objekti

	komuniciraju i interagiraju, što je ključan aspekt objektno orijentiranog programiranja. Lekcija počinje 10-minutnom raspravom o dinamici između objekata neprijatelja i **`Orba`**, istražujući što bi se trebalo dogoditi kada neprijatelj dosegne **`Orb`**. Ova rasprava postavit će temelje za razumijevanje interakcije objekata u kontekstu igre. Nakon toga učenici će se u 15-minutnom zadatku baviti raspravom o tome kako bi instanca klase **`Enemy`** trebala komunicirati s relevantnim objektima koristeći poruke kada se sudari s instancom klase **`Orb`**. Ova rasprava naglasit će važnost komunikacije između objekata i slanja poruka. Sljedećih 10 minuta bit će posvećeno atributima **`Enemy.attack`** i **`Orb.hp`**. Učenici će definirati i razumjeti ove atribute, koji su ključni za upravljanje mehanikom igre. U narednih 15 minuta posvetit će se metodama, učenju kako ih kreirati i implementirati unutar svojih klasa. Ovaj dio lekcije učvrstit će njihovo razumijevanje stvaranja i pozivanja metoda. Zatim će u 5-minutnom zadatku trebati kreirati getter za atribut **`Enemy.attack,`** čime će povećati svoje znanje o enkapsulaciji i pristupu podacima. Narednih 10 minuta bit će posvećeno kreiranju i testiranju metode **`Arena.respawn(Enemy).`** Učenici će implementirati ovu metodu kako bi obradili logiku ponovnog pojavljivanja objekata neprijatelja, čime će osigurati razumijevanje funkcionalnosti metoda i testiranja.

Co-funded by the Erasmus+ Programme of the European Union

	Nakon toga, još 10 minuta bit će posvećeno kreiranju i testiranju metode **`Orb.hit(Enemy)`**, koja će upravljati logikom interakcije kada neprijatelj pogodi **`Orb`**. Tijekom lekcije učenici će raditi pojedinačno ili u malim grupama, čime se potiče suradnja i učenje među vršnjacima. Aktivnim sudjelovanjem u raspravama, kodiranju i testiranju metoda, razvit će vještine kritičkog razmišljanja i rješavanja računalnih problema. Na kraju lekcije učenici će imati sveobuhvatno razumijevanje interakcije objekata, kreiranja metoda i upravljanja atributima u Greenfootu. Ove ključne vještine opremit će ih za daljnje projekte razvoja igara i unaprijediti njihovu ukupnu programersku stručnost.
Ocjenjiva-nje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe.
Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

7.1.1. Priručnik za nastavnike za pripremu lekcije

1. Raspraviti što bi se trebalo dogoditi kada neprijatelj dosegne Orb

Cilj: Nastavnik započinje raspravu o očekivanom ponašanju kada neprijatelj dosegne **`Orb`**. Učenici su potaknuti da razmišljaju o dinamici igre i ishodima, kao što su oštećenje **`Orba`** ili završavanje igre.

Koncepti za raspravu: oštećenje **`Orba`**, uklanjanje neprijatelja, pokretanje događaja u igri (npr. smanjenje zdravlja, reprodukcija zvučnih efekata, završavanje igre).

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.1.** iz projekta **Tower Defense**. Lekcija počinje pregledom prethodno obrađenih koncepata kako bi se osiguralo da učenici učvrste osnovu za novu građu. Nastavnik uključuje učenike u raspravu kako bi razjasnio pojmove interakcije objekata, poruka i metoda u kontekstu Greenfoota. Učenici surađuju u grupama kako bi razmislili o ishodima kada neprijatelj dosegne **`Orb`** u njihovoj igri. Istražuju algoritme kao što je smanjenje zdravstvenih bodova (HP) **`Orba`** nakon kontakta s neprijateljem, raspravljajući o scenarijima u kojima HP **`Orba`** može pasti na nulu, što rezultira završetkom igre. Alternativno, ako HP **`Orba`** ostane iznad nule, dogovaraju se da ponovo postave neprijatelja na drugu lokaciju u areni. Također razmatraju integraciju dodatnih događaja u igri koji se aktiviraju ovom interakcijom, kao što su zvučni efekti ili poruke na ekranu. Tijekom lekcije nastavnik vodi raspravu, potičući učenike da usklade svoje predložene algoritme s određenim scenarijem: osiguranje da kada neprijatelj dosegne **`Orb`**, HP **`Orba`** opada kako je prethodno navedeno. Ovo pomaže učenicima da praktično primijene svoje ideje, učvršćujući njihovo razumijevanje dinamike igre i interakcija unutar okvira njihove igre.

Co-funded by the Erasmus+ Programme of the European Union

2. **Komunikacija instance klase** Enemy **s objektima pri udaru u** Orb

   Cilj: Nastavnik vodi učenike kroz proces razumijevanja kako instanca klase **`Enemy`** treba komunicirati s drugim objektima, konkretno: kada udari instancu klase **`Orb`**, koristeći poruke.

   Koncepti za raspravu: pozivanje metoda, prosljeđivanje parametara, reference na objekte.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **5.2.** iz projekta **Tower Defense**. Učenici surađuju u parovima da bi istražili kako instanca klase **`Enemy`** treba komunicirati s instancom klase **`Orb`** koristeći poruke unutar njihovog scenarija igre. Analiziraju i mapiraju redoslijed poruka i akcija koje bi se trebale odviti kada neprijatelj pogodi **`Orb`**. Ova vježba ima za cilj produbiti njihovo razumijevanje pozivanja metoda, prosljeđivanja parametara i referenci na objekte u kontekstu razvoja igre.

   Dok učenici raspravljaju i usavršavaju svoje ideje, nastavnik im pomaže im da osiguraju da redoslijed interakcija među objektima slijedi algoritam raspodijeljen među suradničkim objektima, prema ciljevima lekcije. Da bi se olakšao ovaj proces, nastavnik može uvesti i koristiti UML dijagram sekvenci kako bi vizualno prikazao interakcije među klasama **`Enemy`**, **`Orb`**, **`Arena i Greenfoot`**. Ovaj vizualni alat pomaže učenicima da bolje razumiju tijek poruka i poziva metoda, jačajući njihovo razumijevanje koncepata objektno orijentiranog programiranja i njihove primjene u programiranju u Javi unutar okruženja Greenfoot.

3. **Atributi Enemy.attack i Orb.hp**

   Cilj: Nastavnik uvodi atribute **`Enemy.attack i Orb.hp`**, objašnjavajući njihovu važnost u određivanju ishodnih interakcija.

   Koncepti za raspravu: atributi klase, enkapsulacija.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.3.** iz projekta **Tower Defense**. Nastavnik uvodi koncept atributa klase i enkapsulacije, objašnjavajući kako atributi poput **`Enemy.attack`** i **`Orb.hp`** mogu predstavljati bitne karakteristike objekata unutar igre. Učenici uče kako definirati i koristiti ove atribute u svom kôdu da bi modelirali snagu napada neprijatelja i zdravstvene bodove **`Orba`**.

Započinju dodavanjem novog cijelog broja atributa nazvanog **`Attack`** u klasu **`Enemy`**, uključujući parametar u konstruktoru za inicijalizaciju ovog atributa. Slično tome, dodaju cijeli broj za atribut nazvan **`HP`** u klasu **`Orb`**, uz parametarski konstruktor za postavljanje ove vrijednosti prilikom kreiranja objekta. Nastavnik vodi učenike kroz prilagodbu kôda u klasi **`Arena`** kako bi se prilagodili novi atributi.

Ovo praktično iskustvo pomaže učenicima da razumiju ulogu atributa klase u interakcijama objekata i važnost enkapsulacije u očuvanju integriteta i sigurnosti kôda.

Commit: 4ca1e9f25685990d2bdfe5b610c28422e0944f95

4. Metode

Cilj: Nastavnik pruža pregled metoda, objašnjavajući kako se koriste za enkapsulaciju radnji i ponašanja unutar klasa.

Koncepti za raspravu: definicija metoda, pozivanje metoda, parametri, povratne vrijednosti.

Aktivnosti: Nastavnik započinje objašnjavanjem koncepta metoda kao enkapsuliranih radnji ili ponašanja unutar klase. Navodi praktične primjere te demonstrira sintaksu i strukturu definicije metoda, pokazujući kako se metode pozivaju na objektima. Učenici uče o različitim vrstama metoda, uključujući one koje obavljaju

Co-funded by the Erasmus+ Programme of the European Union

radnje (metode bez povratne vrijednosti) i one koje vraćaju vrijednosti (metode s povratnom vrijednošću).

Nastavnik objašnjava kako se parametri prenose metodama, naglašavajući važnost tipova parametara i njihovog redoslijeda. Kroz vođene vježbe kodiranja učenici vježbaju definiranje metoda s različitim tipovima parametara i povratnim vrijednostima te pozivanje tih metoda na instancama objekata. Istražuju scenarije u kojima metode obavljaju radnje, mijenjaju stanje objekata ili vraćaju specifične vrijednosti, učvršćujući svoje razumijevanje funkcionalnosti metoda unutar klase.

5. Getter atributa Enemy.attack

Cilj: Nastavnik objašnjava koncept metoda za pristup **`Getteru`** i njihov cilj u pristupu vrijednostima atributa.

Koncepti za raspravu: Metode za pristup **`Getteru`**, enkapsulacija.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.4.** iz projekta **Tower Defense**. Nastavnik započinje objašnjavanjem svrhe metoda za pristup, naglašavajući kako one omogućuju kontroliran pristup vrijednostima atributa dok održavaju enkapsulaciju. Učenici uče važnost korištenja **gettera** za dohvaćanje privatnih vrijednosti atributa, čime se jača koncept zaštite podataka unutar klase.

Nastavnik zatim vodi učenike kroz proces stvaranja metode za pristup atributu **`Attack`** u klasi **`Enemy`**. Kroz praktičan primjer učenici implementiraju metodu za pristup, osiguravajući da vraća vrijednost atributa **`Attack`**. Nastavnik demonstrira ispravnu sintaksu i strukturu za definiranje **gettera** te kako se koristi unutar kôda za pristup vrijednosti atributa.

Na kraju aktivnosti učenici bi trebali moći stvoriti i koristiti metode za pristup kako bi na kontroliran način pristupili vrijednostima atributa, čime će poboljšati svoje razumijevanje enkapsulacije i zaštite podataka u objektno orijentiranom programiranju.

Commit*:* 72b7456ea4cc11416c57d72c89b6a7f7e9266e3e

6. **Kreirati i testirati metodu Arena.respawn(Enemy)**

Cilj: Nastavnik vodi učenike kroz izradu i testiranje metode **`Arena.respawn(Enemy)`**, koja upravlja **`Respawnom`** neprijatelja u igri.

Koncepti za raspravu: implementacija metoda, testiranje, mehanika igre.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.5.** iz projekta **Tower Defense**. Nastavnik započinje uvođenjem koncepta implementacije metoda i njegove važnosti u definiranju specifičnih ponašanja unutar klase. Naglašavajući praktičnu primjenu, učenike vodi da kreiraju metodu **`respawn`** u klasi **`Arena`**. Ova metoda, koja ne vraća vrijednost, prima jedan parametar tipa **`Enemy`**. Učenike se upućuje da postave lokaciju i rotaciju neprijatelja unutar ove metode kako bi odgovarale vrijednostima koje su prvotno postavljene u konstruktoru. Nastavnik demonstrira ispravnu sintaksu i strukturu za definiranje ove metode, naglašavajući ključne koncepte implementacije metoda i prijenosa parametara.

Nakon toga učenici testiraju svoju metodu kako bi osigurali da pravilno funkcionira. Kreiraju instancu **`Arena i Enemy`**, ali ne pokreću aplikaciju odmah. Umjesto toga, povuku instancu **`Enemy`** na novu lokaciju, zatim pristupaju kontekstnom izborniku instance **`Arena`** kako bi pozvali metodu **`respawn`**. Nastavnik objašnjava proces osiguravanja da je aplikacija pauzirana i da je polje parametara aktivno, vodeći učenike da desnom tipkom miša kliknu na instancu **`Enemy`** kako bi pravilno ispunili polje parametara. Učenici

Co-funded by the Erasmus+ Programme of the European Union

promatraju izraz koji se gradi u prozoru, a zatim kliknu na gumb OK kako bi vidjeli učinak svoje metode **`respawn`**.

Kroz ovu aktivnost učenici stječu praktično iskustvo u pisanju i testiranju metoda, razumijevajući kako programski manipulirati objektima igre. Nastavnik osigurava da učenici razumiju svaki korak, pružajući pomoć i pojašnjenja prema potrebi, čime se jača njihovo razumijevanje implementacije metoda i mehanike igre.

Commit*:* 43a221876b8acb4fd507175ec4c8f520121d1ab1

7. Kreirati i testirati metodu Orb.hit(Enemy)

Cilj: Nastavnik upućuje učenike u izradu i testiranje metode **`Orb.hit(Enemy)`**, koja definira interakciju kada neprijatelj udari u **`Orb`**.

Koncepti za raspravu: interakcija metoda, ažuriranje stanja objekta.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.6.** iz projekta **Tower Defense**.Nastavnik započinje objašnjavanjem svrhe metode **`Orb.hit(Enemy)`**, naglašavajući kako ona enkapsulira logiku interakcije između **`Orba`** i neprijatelja. Učenike zatim vodi da dodaju ovu metodu u klasu **`Orb`**. Ova metoda, koja ne vraća vrijednost, prima jedan parametar tipa **`Enemy`**.

Za testiranje metode učenici slijede postupak korak po korak, sličan onom korištenom za testiranje metode **`respawn`**. Kreiraju instancu klase **`Orb`** i instancu klase **`Enemy`.** Ne pokrećući aplikaciju odmah, pristupaju kontekstnom izborniku instance **`Orb`** i odabiru metodu **`Hit`**. Nastavnik osigurava da učenici razumiju kako ispuniti polje parametara desnim klikom na instancu **`Enemy`** dok je aplikacija

pauzirana. Ova radnja gradi izraz za poziv metode, koji učenici zatim izvršavaju klikom na gumb OK.

Nastavnik naglašava važnost promatranja izraza koji se gradi u prozoru kako bi se verificirao poziv metode. Ova vježba pomaže učenicima da razumiju interakciju metoda i proces ažuriranja stanja objekata unutar konteksta igre. Kroz ovu praktičnu aktivnost, učenici stječu iskustvo u implementaciji i testiranju metoda, čime se jača njihovo razumijevanje interakcije metoda i ponašanja igre. Nastavnik pruža podršku i pojašnjenja prema potrebi, osiguravajući da učenici uspješno završe zadatak i razumiju njegov značaj.

Commit*:* fe03d520260f172066be35055a901487bf7c2ff7

7.2. Greenfootovi objekti na zadatku – istraživanje asocijacija i naprednih poziva metoda

Tablica 15. *Greenfootovi objekti na zadatku – istraživanje asocijacija i naprednih poziva metoda*

Naslov	Greenfootovi objekti na zadatku – istraživanje asocijacija i naprednih poziva metoda
Ishodi učenja	Do kraja lekcije učenici bi trebali razviti duboko razumijevanje kako objekti u različitim klasama mogu uspostaviti asocijacije i učinkovito komunicirati unutar Greenfoota. Moći će pokazati vještinu u pozivanju metode **`Orb.hit(Enemy)`** iz klase **`Enemy`**, demonstrirajući svoju sposobnost iniciranja naprednih poziva metoda i olakšavanja komunikacije između objekata. Znat će

Co-funded by the Erasmus+ Programme of the European Union

	objasniti funkcionalnost ključnih metoda u Greenfootu kao što su **`Greenfoot.stop()`** i **`World.getWorldOfType(_cls_)`**, razumijevajući njihovu ulogu u kontroliranju izvođenja igre i učinkovitom upravljanju instancama objekata. Također, učenici će znati uspješno implementirati metodu **`Orb.hit(Enemy)`** unutar svojih projekata, integrirajući interakcije objekata i funkcionalnosti metoda kako bi stvorili interaktivne i dinamične mehanike igre. Učenici bi trebali primijeniti osnovne principe objektno orijentiranog programiranja, uključujući enkapsulaciju i pozivanje metoda, kako bi razvili sofisticirane interakcije i funkcionalnosti igre te stekli praktična saznanja o različitim aspektima razvoja igara, uključujući spawnanje neprijatelja, upravljanje stanjem igre i stvaranje privlačnih iskustava za igrače kroz strukturirane interakcije objekata.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući koncepte poput iteracije i odabira, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Asocijacija (10 min.) 2) Pozivanje metode **`Orb.hit(Enemy)`** iz klase **`Enemy`** (15 min.) 3) Objašnjenje koda za metode **`Greenfoot.stop()`** i **`World.getWorldOfType(_cls_)`** (15 min.) 4) Implementacija metode **`Orb.hit(Enemy)`** (30 min.)

Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	U ovoj 70-minutnoj nastavnoj cjelini učenici će se detaljno upustiti u asocijacije između objekata i napredne pozive metoda unutar Greenfoota. Lekcija ima za cilj produbiti razumijevanje učenika o konceptima objektno orijentiranog programiranja i poboljšati njihovu sposobnost implementacije složenih interakcija u razvoju igara. Lekcija počinje 10-minutnom raspravom o asocijacijama između klasa, s fokusom na to kako objekti mogu međusobno komunicirati i surađivati unutar softverskog sustava. Ovo osnovno razumijevanje postavlja temelje za istraživanje složenijih interakcija. Nakon toga učenici će sudjelovati u 15-minutnom zadatku koji se odnosi na pozivanje metode **`Orb.hit(Enemy)`** iz klase **`Enemy`**. Ovaj zadatak naglašava praktičnu primjenu poziva metoda i prijenosa poruka između objekata. Sljedeći segment uključuje detaljno objašnjenje kôda za metode **`Greenfoot.stop()`** i **`World.getWorldOfType(_cls_),`** u trajanju od 15 minuta. Učenici će steći uvid u način na koji ove metode funkcioniraju unutar Greenfootovih okvira, omogućujući preciznu kontrolu nad elementima igre i upravljanje svijetom. Osnova lekcije posvećena je 30-minutnom zadatku u kojem će učenici implementirati metodu **`Orb.hit(Enemy)`**. Ovaj zadatak izaziva učenike da primijene svoje razumijevanje

Co-funded by the Erasmus+ Programme of the European Union

	implementacije metoda, prijenosa parametara i interakcije objekata kako bi stvorili funkcionalnu mehaniku igre unutar svojih projekata. Na kraju nastavne cjeline učenici će steći dublje razumijevanje asocijacija između objekata, vještine u naprednim pozivima metoda poput **`Orb.hit(Enemy)`** iz klase **`Enemy`**, te uvide u implementaciju ključnih metoda u Greenfootu. Ove vještine omogućit će im stvaranje interaktivnijih i dinamičnijih igara, koristeći puni potencijal principa objektno orijentiranog programiranja u razvoju igara.
Ocjenjivanje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

7.2.1. Priručnik za nastavnike za pripremu lekcije

1. Asocijacija

Cilj: Nastavnik objašnjava pojam asocijacija između objekata, naglašavajući kako objekti različitih klasa mogu međusobno komunicirati.

Koncepti za raspravu: asocijacije, interakcije objekata, odnosi među klasama.

Aktivnosti: Lekcija počinje kratkim pregledom asocijacija između klasa u objektno orijentiranom programiranju. Nastavnik potiče učenike na raspravu o tome kako objekti međusobno komuniciraju kroz asocijacije, koristeći praktične primjere iz Greenfoota za ilustraciju tih pojmova. Učenici istražuju kako asocijacije definiraju suradnju između klasa, kao što je primjerice utjecaj klase **`Enemy`** na klasu **`Orb`** u igri.

Kroz detaljnu raspravu nastavnik objašnjava različite vrste asocijacija unutar Greenfoota: „jedan na jedan“, „jedan na više“ i „više na više“. Na primjer, asocijacija „jedan na jedan“ može ilustrirati kako je igrač povezan sa svojim avatarom, asocijacija „jedan na više“ može prikazivati kako jedan svijet sadrži više instanci klase **`Actor`**, dok asocijacija „više na više“ može prikazivati kako različiti neprijatelji interagiraju s više **`Orba`** u nivou igre.

Koristeći UML dijagrame klasa specifične za Greenfoot, nastavnik vizualno prikazuje ove odnose, pomažući učenicima da razumiju kako su asocijacije strukturirane i implementirane u njihovim projektima igre. Na primjer, dijagram može prikazivati kako se klasa **`Enemy`** povezuje s više instanci klase **`Orb`**, što ukazuje na odnos „jedan na više“, kad svaki neprijatelj utječe na nekoliko **`Orba`**.

Učenici aktivno sudjeluju u identificiranju i mapiranju tih asocijacija unutar svojih projekata igre. Istražuju kako objekti međusobno djeluju na temelju tih odnosa i raspravljaju o posljedicama za

Co-funded by the Erasmus+ Programme of the European Union

mehaniku i logiku igre. Nastavnik pruža konkretne primjere iz konteksta razvoja igre, ilustrirajući kako instance **`Enemy`** interagiraju s instancama **`Orb`** i kako su te interakcije regulirane asocijacijama.

Na kraju aktivnosti učenici stječu dubinsko razumijevanje uloge asocijacija u dizajniranju i implementiranju interaktivnih sustava unutar Greenfoota. Mogu identificirati različite vrste asocijacija i primijeniti ovo znanje za modeliranje i implementaciju složenih interakcija između objekata u Greenfootu. Ovo praktično razumijevanje jača njihovu sposobnost dizajniranja kohezivnih i interaktivnih scenarija igara koristeći principe objektno orijentiranog programiranja.

2. Pozivanje metode Orb.hit(Enemy) iz klase Enemy

Cilj: Nastavnik vodi učenike kroz proces pozivanja metode **`Orb.hit(Enemy)`** unutar klase **`Enemy.`**

Koncepti za raspravu: pozivanje metoda, reference na objekte.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.7.** iz projekta **Tower Defense.** Učenici se aktivno upuštaju u izmjenu metode **`act()`** klase **`Enemy.`** Uklanjaju postojeći kôd koji je odgovoran za nepotrebne radnje kao što su rotacija pri kontaktu s **`Orbom`** ili odbijanje od rubova svijeta. Umjesto toga implementiraju funkcionalnost za pozivanje metode **`Orb.hit(Enemy)`** kada instanca klase **`Enemy`** dođe u kontakt s instancom klase **`Orb.`**

Kroz praktične primjere i demonstracije nastavnik pokazuje kako pravilno postaviti poziv metode. Učenici uče koristiti reference na objekte (npr. **`this)`** za aktiviranje metode **`hit()`** na instanci **`Orb`** prilikom sudara s **`Enemy`**. Istražuju tok kontrole u Greenfootu, razumijevajući kako pozivanje metoda usmjerava izvršni put unutar njihove igre.

Tijekom aktivnosti nastavnik pruža smjernice za ispravljanje pogrešaka i testiranje implementacije kako bi se osiguralo da pozivanje `Orb.hit(Enemy)` funkcionira prema planu. Učenici promatraju ponašanje u simulacijskom okruženju Greenfoot, provjeravajući aktivira li poziv metode ispravno interakcije između objekata `Enemy i Orb.`

Na kraju lekcije učenici stječu vještine u pozivanju metoda i interakciji između objekata unutar programiranja u Greenfootu. Razumiju kako koristiti reference na objekte za pozivanje metoda u različitim klasama, čime produbljuju svoje razumijevanje principa objektno orijentiranog programiranja u kontekstu razvoja igara.

Commit*:* 63f9c96717d9d2587b60095e3b249b0158c8587b

3. **Objašnjenje kôda za metode Greenfoot.stop() i World.getWorldOfType(_cls_)**

   Cilj: Nastavnik objašnjava funkcionalnost metoda `Greenfoot.stop()` i `World.getWorldOfType(_cls_).`

   Koncepti za raspravu: metode za kontrolu igre, upravljanje svjetovima.

   Aktivnosti: U ovoj lekciji učenici se usmjeravaju na razumijevanje dviju ključnih metoda unutar Greenfoota: `Greenfoot.stop()` i `World.getWorldOfType(_cls_).` Nastavnik započinje objašnjavanjem svrhe i upotrebe svake metode, naglašavajući njihove uloge u kontroli igre i upravljanju svjetovima.

   Metoda `Greenfoot.stop()` je ključna za kontrolu toka izvršenja scenarija u Greenfootu. Kada se pozove, zaustavlja simulaciju i zamrzava sve likove i interakcije unutar svijeta. Ova metoda je posebno korisna za implementaciju funkcionalnosti pauze u igri ili pokretanje specifičnih događaja koji zahtijevaju privremeno zaustavljanje napredovanja igre.

Co-funded by the Erasmus+ Programme of the European Union

S druge strane, metoda **`World.getWorldOfType(_cls_)`** ima drugačiju svrhu vezanu uz upravljanje svjetovima. Ona omogućava programerima da dobiju instance svjetova koji su specifične klase **`(_cls_)`**. Pretražuje sve aktivne svjetove u Greenfootu i vraća instance klase svijeta koje odgovaraju navedenom tipu. Ova sposobnost je korisna kada programeri trebaju dinamički manipulirati svjetovima ili međusobno djelovati s njima, na temelju njihovih atributa klase.

Učenici sudjeluju u praktičnim demonstracijama i primjerima kako bi produbili svoje razumijevanje ovih metoda. Nastavnik pokazuje kako **`Greenfoot.stop()`** može biti integriran u scenarije igre da bi se stvorila funkcionalnost pauze ili pokrenuli specifični događaji u igri. Učenici promatraju kako pauza u igri utječe na ponašanje likova i interakcije unutar simulacijskog okruženja Greenfoot.

Slično tome, učenici istražuju metodu **`World.getWorldOfType(_cls_)`** kroz praktične vježbe. Uče kako koristiti ovu metodu za dinamičko dohvaćanje instanci specifičnih tipova svjetova. Nastavnik demonstrira scenarije u kojima je važno dohvatiti svjetove određenog tipa za implementaciju naprednih mehanika igre ili upravljanje više istodobnih okruženja igre unutar Greenfoota.

Tijekom aktivnosti nastavnik potiče raspravu i pruža praktične primjere kodiranja kako bi ilustrirao primjenu ovih metoda u stvarnim scenarijima razvoja igara. Učenici aktivno sudjeluju u eksperimentiranju s metodama unutar vlastitih projekata u Greenfootu, učvršćujući svoje razumijevanje kroz izravnu primjenu i istraživanje.

Na kraju lekcije učenici stječu vještine u korištenju metode **`Greenfoot.stop()`** za kontrolu igre i **`World.getWorldOfType(_cls_)`** za učinkovito upravljanje svjetovima unutar Greenfoota. Stječu

praktične vještine koje poboljšavaju njihovu sposobnost implementacije složenih ponašanja igre i upravljanja stanjem igre koristeći ove ključne metode.

4. **Implementacija metode Orb.hit**

Cilj: Nastavnik vodi učenike kroz implementaciju metode `Orb.hit(Enemy)`.

Koncepti za raspravu: implementacija metoda, ažuriranje stanja objekta, mehanika igre.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.8.** iz projekta **Tower Defense.** Učenici počinju implementaciju metode `Orb.hit(Enemy)`, koja je ključna za definiranje interakcije između neprijatelja i `Orba` unutar njihove igre. Metoda `Orb.hit(Enemy)` igra ključnu ulogu u određivanju posljedica kada neprijatelj udari u `Orb` u igri. Učenici će prvo smanjiti zdravlje `Orba`. Kada se pozove `Orb.hit(Enemy)`, metoda bi trebala smanjiti broj zdravstvenih bodova `Orba` (HP), simulirajući štetu koju `Orb` pretrpi pri sudaru s neprijateljem.

Nakon toga treba provjeriti je li zdravlje `Orba` palo na nulu ili ispod nje. Ako jest, igra bi trebala završiti, što se postiže pozivom metode `Greenfoot.stop()` za zaustavljanje igre. To označava kraj igre.

Ako zdravlje `Orba` ostane iznad nule nakon sudara s neprijateljem, učenici trebaju implementirati logiku za ponovni ulazak neprijatelja u arenu kako bi se omogućio nastavak igre.

Tijekom aktivnosti nastavnik pruža smjernice o strukturi metode, korištenju parametra `(Enemy)` i ažuriranju zdravstvenog stanja `Orba (HP).` Učenici surađuju u raspravi i odlučivanju o specifičnim mehanikama igre, kao što su količina štete koju neprijatelj nanosi `Orbu` i što se događa kada zdravlje `Orba` bude iscrpljeno.

Co-funded by the Erasmus+ Programme of the European Union

Nastavnik potiče učenike da temeljito testiraju svoje implementacije kako bi osigurali da metoda funkcionira ispravno u različitim scenarijima igre. Na kraju aktivnosti učenici stječu praktično iskustvo u implementaciji metode za učinkovito upravljanje interakcijama igre, što poboljšava njihovo razumijevanje mehanike igre unutar Greenfoota.

Commit*:* 84bcd7c128faaa9313b507f7438f826ae2f47d2c

7.3. Greenfootovi objekti na zadatku – tornjevi, meci i strateške interakcije

Tablica 16. *Greenfootovi objekti na zadatku – tornjevi, meci i strateške interakcije*

Naslov	Greenfootovi objekti na zadatku – tornjevi, meci i strateške interakcije
Ishodi učenja	Do kraja nastavne cjeline učenici će razviti vještine u kreiranju klasa **`Bullet i Tower`**, postavljajući temelje za strateški razvoj igara. Shvatit će i implementirati realističan pokret metaka te odrediti radnje kada se instance metaka susretnu s instancama neprijatelja ili s rubovima arene. Učenici će dizajnirati učinkovite mehanike pucanja za instance **`Tower`**, koristeći prijenos poruka između objekata **`Tower`** i drugih elemenata igre kako bi poboljšali dinamiku igranja. Strategijski će postaviti instance **`Tower`** unutar igre i primijeniti principe objektno orijentiranog programiranja, poput enkapsulacije i poziva metoda, kako bi osigurali kreiranje robusnih i održivih mehanika igre. Kroz suradničko rješavanje problema učenici će se suočiti s izazovima u dizajniranju i implementaciji strateških interakcija između tornjeva, metaka i ostalih elemenata igre, stječući praktične

	uvide u principe dizajna igara i unapređujući svoje ukupno razumijevanje mehanike obrane tornja u Greenfootu.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući koncepte poput iteracije i odabira, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Kreirajte klase **`Bullet i Towe`**r (10 min.) 2) Raspravite kako bi se instanca klase **`Bullet`** trebala kretati i što bi se trebalo dogoditi kada dosegne instancu klase **`Enemy`** ili rub arene (10 min.) 3) Implementirajte kretanje instance klase **`Bullet`** (30 min.) 4) Raspravite kako će instanca klase **`Tower`** ispaljivati instancu klase **`Bullet`** (15 min.) 5) Raspravite kako bi instanca klase **`Towe`**r trebala komunicirati s relevantnim objektima pomoću poruka prilikom ispaljivanja (15 min.) 6) Implementirajte ispaljivanje instance klase **`Tower`** (30 min.) 7) Tornjevi u areni (20 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN Resursi iz projekta OOP4FUN Izvorni kôd projekta s GitHuba/GitLaba Internetski resursi.
Opis	U ovom 130-minutnom scenariju učenja učenicićе se potpuno posvetiti dinamici interakcija između tornjeva i

Co-funded by the Erasmus+ Programme of the European Union

	metaka unutar Greenfoota. Lekcija je usmjerena na razvijanje vještina učenika u dizajnu i implementaciji strateških elemenata igre kako bi se stvorila angažirajuća i interaktivna igrivost. Lekcija započinje zadatkom od 10 minuta u kojem učenici kreiraju klase **`Bullet`** i **`Tower.`** Ovaj osnovni korak postavlja temelje za razumijevanje i implementaciju interakcija između tih elemenata igre. Nakon toga slijedi 10-minutna rasprava o tome kako bi se instance klase **`Bullet`** trebale kretati i što bi se trebalo dogoditi kada metak dosegne instancu klase **`Enemy`** ili rub arene. Ova rasprava postavlja temelje za implementaciju preciznih i dinamičnih mehanika kretanja. Učenici zatim posvećuju 30 minuta implementaciji kretanja instanci klase **`Bullet.`** Ovaj zadatak motivira učenike da primijene svoje razumijevanje metode **`move()`** iz Greenfoota i rukovanja događajima za simulaciju realističnog ponašanja metaka unutar okruženja igre. Lekcija se nastavlja 15-minutnim zadatkom koji raspravlja o tome kako bi instance klase **`Tower`** trebale ispaljivati instance klase **`Bullet`**. Ova rasprava pokriva logiku i uvjete za pokretanje ispaljivanja metaka iz tornjeva. Zatim će 15 minuta biti posvećeno raspravi o tome kako bi instance klase **`Tower`** trebale komunicirati s relevantnim objektima putem poruka prilikom ispaljivanja. Ovaj segment naglašava važnost komunikacije između objekata i pokretanja događaja u razvoju igre. Učenici potom provode 30 minuta implementirajući mehanizam ispaljivanja instanci klase **`Tower`**. Ovaj zadatak

	zahtijeva od učenika da integriraju logiku ispaljivanja s interakcijama objekata, osiguravajući da tornjevi učinkovito reagiraju na neprijatelje ili druge elemente igre. Lekcija se završava 20-minutnim zadatkom koji se fokusira na upravljanje i postavljanje tornjeva unutar arene. Ovaj zadatak istražuje postavljanje, interakciju i strateško pozicioniranje tornjeva kako bi se optimizirala dinamika igranja i učinkovito izazvali igrači. Na kraju lekcije učenici će steći praktično iskustvo u dizajniranju i implementaciji strateških interakcija između tornjeva i metaka u Greenfootu. Steći će vještine u objektno orijentiranom programiranju, rukovanju događajima i strateškom dizajnu igara, pripremajući ih za stvaranje angažirajućih i dinamičnih igara.
Ocjenjiva-nje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe.
Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

Co-funded by the Erasmus+ Programme of the European Union

7.3.1. Priručnik za nastavnike za pripremu lekcije

1. Kreirajte klase `Bullet` i `Tower`

Cilj: Nastavnik vodi učenike kroz proces kreiranja klasa **`Bullet`** i **`Tower`**.

Koncepti za raspravu: kreiranje klasa, uloge klasa i početna konfiguracija.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.9.** iz projekta **Tower Defense.** Lekcija počinje pregledom osnovnih pojmova vezanih za kreiranje objekata, njihovo kretanje i interakciju unutar Greenfoota. Nastavnik potiče učenike na raspravu kako bi razjasnio uloge različitih klasa i njihove interakcije u igri.

Učenici kreiraju dvije nove klase u Greenfootu, razumijevajući svrhu svake od njih u kontekstu igre. Uče kako postaviti ove klase, pripremajući se za složenije interakcije u budućim lekcijama.

Započinju kreiranjem klase **`Bullet`**. Ova klasa će predstavljati projektile koje će ispaljivati tornjevi. Otvaraju Greenfoot, odabiru **`New Class`** iz izbornika i pozivaju novu klasu **`Bullet.`** Nakon toga učenici kreiraju klasu **`Tower`**. Ona će predstavljati tornjeve koji ispaljuju metke na neprijatelje. Ponovno odabiru **`New Class`** iz izbornika i pozivaju novu klasu **`Tower.`**

Tijekom lekcije, nastavnik objašnjava uloge klasa **`Bullet i Tower`** unutar igre. Klasa **`Bullet`** predstavlja projektile koje će tornjevi ispaljivati, dok klasa **`Tower`** predstavlja stacionarne objekte koji mogu ispaljivati metke na neprijatelje. Nastavnik osigurava da učenici razumiju različite uloge svake klase i kako će one međusobno djelovati u igri.

Na kraju ove aktivnosti učenici bi trebali imati osnovne strukture za klase **`Bullet`** i **`Tower`**, postavljajući temelje za detaljniju implementaciju u budućim lekcijama.

Commit*:* ece4df70042c8f60098e14ad2cee55514897d825

2. **Raspravite kako bi se instanca klase `Bullet` trebala kretati i što bi se trebalo dogoditi kada dosegne instancu klase `Enemy` ili rub arene**

   Cilj: Nastavnik vodi raspravu o očekivanom ponašanju metka dok se kreće u igri.

   Koncepti za raspravu: Logika kretanja, detekcija sudara. **Aktivnosti:** Učenici provode metodu oluje mozgova i raspravljaju o logici kretanja metka i upravljanju sudarima. Nastavnik učenicima zadaje zadatak **5.10.** iz projekta **Tower Defense**.

   Počinju raspravom o tome kako bi se instanca klase **`Bullet`** trebala kretati unutar igre. Slažu se da se metak treba kretati u pravoj liniji u pravcu u kojem je ispaljen, bez promjene pravca. Brzina metka treba biti upravljiva i dosljedna. Za implementaciju ovog ponašanja mogu koristiti ugrađene metode za kretanje u Greenfootu. Pritom je važno naglasiti ulogu konstruktora u inicijalizaciji vrijednosti atributa prilikom kreiranja instanci objekata.

   Nastavnik zatim vodi raspravu o tome što se treba dogoditi kada metak dođe do ruba svijeta ili se sudari s neprijateljem. Učenici predlažu da, kada metak dostigne rub svijeta, treba biti uklonjen iz igre. Također raspravljaju da, pri sudaru s neprijateljem, metak treba nestati, a neprijatelj treba pretrpjeti štetu ili biti uništen.

   Učenici predlažu sljedeći pseudokôd za logiku kretanja i sudara metka: najprije pomaknuti metak naprijed konstantnom brzinom,

Co-funded by the Erasmus+ Programme of the European Union

zatim provjeriti je li dosegnuo rub svijeta. Ako jest, treba ukloniti metak iz svijeta. Ako pak nije, valja provjeriti je li metak udario neprijatelja. Ako dođe do sudara, to znači da je metak pogodio neprijatelja, pa treba ukloniti metak iz svijeta i nanijeti štetu neprijatelju, ili čak ukloniti i neprijatelja.

Nastavnik demonstrira kako implementirati ovu logiku koristeći metode iz Greenfoota **`move(int)`**, **`isAtEdge()`** i **`getOneIntersectingObject(Class cls)`**.

Potiče učenike da usavrše svoje ideje i razmisle o dodatnim detaljima, kao što su prilagođavanje brzine na osnovi težine igre ili dodavanje vizualnih efekata kada metak pogodi neprijatelja. Ova aktivnost pomaže učenicima da shvate principe kretanja i detekcije sudara u razvoju igara, pripremajući ih za daljnju implementaciju u svojim projektima.

3. **Implementirajte kretanje instance klase `Bullet`**
   Cilj: Nastavnik pomaže učenicima da implementiraju logiku kretanja za klasu **`Bullet`**.

   Koncepti za raspravu: kretanje likova, granice svijeta.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **5.11.** iz projekta **Tower Defense.**

   Učenici pišu kôd za kretanje metka naprijed i upravljanje njegovim uklanjanjem kada dostigne rub arene.

   Učenici počinju pregledom koncepata kretanja likova i granica svijeta, fokusirajući se na to kako se ovi koncepti primjenjuju na klasu **`Bullet`**. Pritom ih nastavnik podsjeća na prethodne zadatke, naglašavajući znanje koje trebaju primijeniti. Učenici se prisjećaju kako da dodaju kôd u metodu **`act()`** za upravljanje interakcijama

na rubu svijeta. Također se podsjećaju kako da upravljaju promjenama pravca kada lik uđe u određena područja. Osim toga, pamte kako da koriste brojače i mehanizme kašnjenja u metodi `act()` za kontrolirano kretanje.

Uzimajući u obzir ove koncepte, učenici prelaze na implementaciju logike kretanja za klasu `Bullet`. Počinju dodavanjem metode `move(int)` u metodu `act()` klase `Bullet` kako bi metak neprekidno napredovao.

Commit*:* d372827a831381b2254f838041fa4d9a42e53b82

4. **Raspravite kako će instanca klase `Tower` ispaljivati instancu klase `Bullet`**

   Cilj: Nastavnik raspravlja s učenicima o logici ispaljivanja metaka iz tornja.

   Koncepti za raspravu: kreiranje objekata, pozivanje metoda.

   Aktivnosti: Učenici razrađuju kako će toranj kreirati i ispaljivati metke. Nastavnik učenicima zadaje zadatak **5.12.** iz projekta **Tower Defense.**

   Započinju raspravom o općoj logici potrebnoj da toranj ispaljuje metke u igri. Fokusiraju se na ključne koncepte poput kreiranja objekata metaka i pozivanja metoda za njihovo ispaljivanje. Nastavnik naglašava da toranj ne bi trebao ispaljivati metke pri svakom pozivu metode `act()`, kao što je kretanje neprijatelja bilo upravljano mehanizmom kašnjenja.

   Učenici razmatraju korake potrebne da toranj ispaljuje metke s prekidima. Nastavnik ih usmjerava na razmatranje korištenja

Co-funded by the Erasmus+ Programme of the European Union

mehanizma kašnjenja za ispaljivanje. Objašnjava se uloga konstruktora u implementaciji ovog mehanizma.

Nastavnik objašnjava da će trebati uvesti novi atribut – **`shootDelay`** i brojač – **`nextShootCounter`**, u klasu **`Tower`**. Ovi atributi će kontrolirati učestalost ispaljivanja.

Nastavnik izlaže relevantne korake i metode za klasu **`Tower`**. Prvo treba definirati atribut **`shootDelay`** i inicijalizirati **`nextShootCounter`** na 0 u klasi **`Tower`**. Nakon toga metoda **`act()`** klase **`Tower`** trebala bi biti modificirana za upravljanje logikom ispaljivanja. Metoda bi trebala kreirati i ispaliti metak samo kada **`nextShootCounter`** dostigne 0. Poslije ispaljivanja **`nextShootCounter`** treba biti resetiran na vrijednost **`shootDelay`**. Ako **`nextShootCounter`** nije 0, treba se smanjiti za 1. Na kraju treba definirati zasebnu metodu **`fire()`** za upravljanje kreiranjem i ispaljivanjem metaka. Ova metoda će instancirati objekt **`Bullet`** i dodati ga u svijet.

Kroz ovaj proces učenici razumiju kako implementirati logiku ispaljivanja razdvajanjem relevantnih koraka u metode klase **`Tower`**. Nastavnik osigurava da učenici shvate važnost pozivanja metoda i kreiranja objekata, učvršćujući njihovo razumijevanje ovih koncepata u kontekstu vlastite igre.

5. **Raspravite kako bi instanca klase `Tower` trebala komunicirati s relevantnim objektima pomoću poruka prilikom ispaljivanja**

   Cilj: Nastavnik objašnjava kako toranj komunicira s mecima i drugim objektima putem poruka.

   Koncepti za raspravu: slanje poruka, pozivanje metoda.

   Aktivnosti: Učenici raspravljaju o mehanizmu slanja poruka za ispaljivanje metaka. Nastavnik učenicima zadaje zadatak **5.13.** iz projekta **Tower Defense.**

Ova nastavna cjelina započinje objašnjenjem nastavnika o konceptu slanja poruka i njegovom značaju u objektno orijentiranom programiranju. Nastavnik naglašava kako objekti u igri komuniciraju međusobno koristeći metode, koje služe kao poruke.

Kako bi to ilustrirao, nastavnik koristi UML dijagram sekvenci za opisivanje interakcija među objektima **`Tower`**, **`Bullet i Arena`**. Dijagram vizualno prikazuje tok poruka i poziva metoda, pomažući učenicima da razumiju redoslijed interakcija.

Učenici raspravljaju o detaljnom procesu kako klasa **`Tower`** treba komunicirati s mecima i drugim objektima prilikom ispaljivanja. Nastavnik objašnjava da kada toranj odluči ispaliti metak, šalje poruku (poziv metode) za kreiranje instance metka i dodavanje u arenu. Ova interakcija započinje unutar metode **`act()`** klase **`Tower`**. Korištenjem UML dijagrama sekvenci nastavnik pokazuje kako toranj šalje poruku Greenfoot *frameworku* da doda novi objekt metka **`(Bullet)`** u svijet. Zatim toranj šalje poruku instanci metka, postavljajući njegovu orijentaciju tako da odgovara trenutnoj rotaciji tornja. Time se osigurava da metak ide u željenom smjeru.

Kada se metak stvori i pozicionira, interakcija s drugim objektima u igri, poput neprijatelja ili rubova svijeta, započinje. Nastavnik objašnjava kako te interakcije obrađuje metoda **`act()`** metka, koja može uključivati provjeru sudara i uklanjanje metka kada je to potrebno.

Kroz cijelu lekciju nastavnik naglašava suradničku prirodu tih interakcija, prikazujući kako je algoritam raspoređen među suradničkim objektima. To pomaže učenicima da cijene modularni dizajn i jasne komunikacijske putanje unutar njihove igre.

Co-funded by the Erasmus+ Programme of the European Union

6. Implementirajte ispaljivanje instance klase `Tower`

Cilj: Nastavnik vodi učenike kroz implementaciju mehanizma pucača za klasu **`Tower.`**

Koncepti za raspravu: kreacija objekata, pozicioniranje aktera.

Aktivnosti: Učenici pišu kôd kako bi omogućili tornju da puca metke. Nastavnik učenicima zadaje zadatak **5.14.** iz projekta **Tower Defense.**

Na početku se pojašnjava krajnji cilj: implementirati mehanizam pucača za klasu **`Tower`**. Nastavnik zatim razlaže zadatak na upravljive korake i vodi učenike kroz svaki od njih.

Prvo, učenici pripremaju potrebne atribute i konstruktore za klasu **`Tower`**, a nastavnik objašnjava da tornju treba atribut za praćenje kada može pucati. Zatim učenici kreiraju dvije metode: **`boolean Tower.canShoot()`** i **`void Tower.fire().`** Na početku, ove metode mogu vraćati **`false`** i ne izvoditi nikakvu akciju, respektivno, kako bi se mogle koristiti u metodi **`act()`**.

Metoda **`act()`** se zatim ažurira kako bi koristila ove metode. Nastavnik objašnjava da metoda **`canShoot()`** treba vraćati **`true`** ako **`shootCounter`** dostigne 0. Metoda **`fire()`** se implementira za kreiranje instance metka i pravilno pozicioniranje.

Nastavnik osigurava da učenici razumiju svaki dio kôda, naglašavajući kreaciju objekta metka, njegovo pozicioniranje u svijetu i usklađivanje rotacije s tornjem.

Učenici zatim testiraju svoje rješenje pokretanjem igre, postavljanjem instance tornja i provjerom ispaljuje li metke u odgovarajućim intervalima, dok ih nastavnik ohrabruje da riješe

eventualne probleme, osiguravajući da se meci kreiraju i kreću prema očekivanjima.

Na kraju cjeline učenici su sposobni implementirati funkcionalan mehanizam pucača za klasu `Tower`, učvršćujući svoje razumijevanje kreacije objekata, pozicioniranja aktera i poziva metoda u Greenfootu.

Commit*:* 62aec085954beacf996865a55bed312a09c675f2

7. Tornjevi u areni

Cilj: Nastavnik pomaže učenicima da integriraju tornjeve i metke u igru, stvarajući funkcionalnu arenu.

Koncepti za raspravu: integracija igre, testiranje.

Aktivnosti: Učenici postavljaju tornjeve u arenu i testiraju njihovu interakciju s mecima i neprijateljima. Nastavnik učenicima zadaje zadatak **5.15.** iz projekta **Tower Defense.**

Lekcija počinje objašnjenjem cilja: integrirati tornjeve u arenu i osigurati da pravilno interagiraju s mecima i neprijateljima. To će uključivati postavljanje tornjeva u arenu i testiranje njihovog ponašanja unutar okruženja igre.

Učenici počinju postavljanjem instanci klase `Tower` na različite pozicije unutar arene. Nastavnik objašnjava kako dodati tornjeve putem sučelja Greenfoot, osiguravajući da svaki toranj bude pravilno pozicioniran.

Nakon toga uvodi koncept preopterećenja konstruktora. Ovo je posebno korisno za inicijalizaciju objekata klase `Tower` s različitim

Co-funded by the Erasmus+ Programme of the European Union

rotacijama. Nastavnik zatim vodi učenike kroz ažuriranje klase `Tower` kako bi uključila preopterećeni konstruktor koji prihvaća cijeli broj kao parametar za rotaciju. Ovo omogućava veću kontrolu nad postavljanjem i orijentacijom tornjeva unutar arene. Kad postave i konfiguriraju tornjeve, učenici testiraju njihovu funkcionalnost u igri, provjeravajući kako se ponašaju u interakciji s mecima i neprijateljima. Nastavnik potiče učenike da rješavaju eventualne probleme i optimiziraju rad tornjeva kako bi osigurali da igra funkcionira glatko i prema očekivanjima.

Na kraju lekcije učenici će imati funkcionalnu arenu s integriranim tornjevima i mecima, čime će učvrstiti svoje razumijevanje integracije igre i testiranja unutar Greenfoota.

Commit*:* bfb6a271f490c341c760e654b3f86a87111c54cb

7.4. Greenfootovi objekti na zadatku – meci, neprijatelji i dinamika igre

Tablica 17. *Greenfootovi objekti na zadatku – meci, neprijatelji i dinamika igre*

Naslov	Greenfootovi objekti na zadatku – meci, neprijatelji i dinamika igre
Ishodi učenja	Do kraja lekcije učenici će razviti vještinu dizajniranja i implementiranja interaktivne dinamike igre koristeći klase `Bullet i Enemy` unutar Greenfoota. Razumjet će kako olakšati interakciju objekata putem slanja poruka, omogućujući efektivnu komunikaciju između elemenata igre. Također će demonstrirati sposobnost implementiranja precizne detekcije kolizije i mehanizme odgovora, posebno detaljno opisujući kako instance klase `Bullet` komuniciraju

	s instancama klase **`Enemy`**. Dobit će praktičan uvid u osnovne Greenfootove metode kao što su **`Greenfoot.showText(String, int, int)`**, **`Greenfoot.getRandomNumber(int)`** i **`World.act()`**, koristeći ih u svrhu poboljšanja igre, uvođenja slučajnosti i upravljanja promjenama stanja igre. Nadalje, savladat će vještinu implementiranja mehanizama prikaza neprijatelja i uvjeta za završetak igre, osiguravajući dinamična iskustva igranja. Kroz reviziju objektnih asocijacija, učenici će učvrstiti svoje razumijevanje kako objekti surađuju da bi se stvorila zanimljiva dinamika igre, pripremajući ih za primjenu tih vještina za buduće projekte razvoja igara.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja, uključujući iteraciju i koncepte selekcije, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Rasprava o tome kako bi instanca klase **`Bullet`** trebala komunicirati s relevantnim objektima korištenjem poruka (15 min.) 2) Implementirajte da instanca klase **`Bullet`** pogađa instancu klase **`Enemy`** (30 min.) 3) Objašnjavanje kôda za metode **`Greenfoot.showText(String, int, int)`**, **`Greenfoot.getRandomNumber(int)`** i **`World.act()`** (15 min.) 4) Pojavljivanje neprijatelja i kraj igre (30 min.) 5) Revizija asocijacija (20 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN.

Co-funded by the Erasmus+ Programme of the European Union

	Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	U ovom 110-minutnom scenariju učenici će dublje ući u zamršenost dinamike igre, uključujući metke i neprijatelje unutar Greenfoota. Lekcija se fokusira na razvoj učeničkih vještina u kreiranju interaktivne i dinamične igrivosti kroz efektivnu interakciju objekata i mehaniku igre. Lekcija počinje 15-minutnom raspravom o načinu kako instance klase **`Bullet`** trebaju komunicirati s relevantnim objektima igre korištenjem poruka. Ova rasprava postavlja temelje za razumijevanje kako objekti komuniciraju i kolaboriraju kako bi se postigla specifična ponašanja igre. Nakon toga učenici će posvetiti 30 minuta implementiranju funkcionalnosti kojima instance klase **`Bullet`** uspješno pogađaju instance klase **`Enemy`**. Ovaj zadatak predstavlja učenicima izazov da primijene svoje razumijevanje detekcije kolizije objekata i upravljanje događajima kako bi kreirali dojmljive interakcije unutar igre. Idući segment uključuje 15-minutno objašnjenje esencijalnih metoda u Greenfootu: **`Greenfoot.showText(String,int,int)`**, **`Greenfoot.getRandomNumber(int)`**, i **`World.act()`**. Učenici će dobiti uvid u to kako te metode doprinose prikazu teksta, generiranju slučajnih brojeva za mehaniku igre i upravljanju ciklusom ažuriranja svijeta igre. Učenici zatim provode 30 minuta na zadacima povezanim s pojavljivanjem neprijatelja i uvjetima kraja igre. To uključuje dizajniranje i implementiranje mehanizama pojavljivanja neprijatelja u odgovarajućim intervalima i

	određivanju uvjeta za završetak igre temeljem igračevih akcija ili ciljeva igre. Slijedi 20-minutna revizija lekcije, s fokusom na konsolidiranju razumijevanja veza objekata i njihovih uloga u implementiranju dinamike igre. Učenici će pregledati i poboljšati svoje razumijevanje kako objekti međudjeluju i surađuju unutar Greenfoota kako bi postigli željeni efekt igre. Na kraju lekcije učenici će imati dublje razumijevanje kreiranja interaktivne i privlačne dinamike igre, uključujući metke, neprijatelje i stratešku mehaniku igre unutar okruženja Greenfoot. Steći će praktične vještine implementiranja interakcija objekata, upravljanja stanjima igre i poboljšanja iskustva igrača kroz principe strukturiranog dizajna igara.
Ocjenjivanje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

Co-funded by the Erasmus+ Programme of the European Union

7.4.1. Priručnik za nastavnike za pripremu lekcije

1. **Raspravite kako bi instance klase Bullet trebale komunicirati s relevantnim objektima korištenjem poruka**

   Cilj: Nastavnik vodi raspravu o tome kako bi meci trebali komunicirati s ostalim objektima, naročito neprijateljima, kroz slanje poruka.

   Koncepti za raspravu: prijenos poruka, detekcija kolizije i interakcija objekata.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **5.16.** iz projekta **Tower Defense.** Lekcija počinje pregledom interakcija objekata unutar Greenfoota s fokusom na to kako instance različitih klasa komuniciraju i utječu jedne na druge. Nastavnik uključuje učenike u raspravu kako bi se utvrdili ti koncepti i njihova praktična primjena u razvoju igara.

   Učenici zajednički razmišljaju i raspravljaju logiku za interakciju metaka i poruka koje moraju poslati. Nastavnik započinje pojašnjavanjem važnosti prijenosa poruka u objektno orijentiranom programiranju. Rasprava će se fokusirati na to kako instance klase `Bullet` međudjeluju s ostalim objektima kao što su neprijatelji i arena, naročito kada metak pogodi neprijatelja.

   Učenici se potiču da zajednički razmisle i podijele svoje ideje o logici za interakciju metka. Razmatraju pitanja kao što su: Što bi se trebalo dogoditi kada metak pogodi neprijatelja? Kako bi metak trebao iskomunicirati taj događaj ostalim objektima? Koje poruke se trebaju prenijeti kako bi se upravljalo tom interakcijom na odgovarajući način?

   Nastavnik uvodi koncept detekcije kolizije, objašnjavajući pritom kako igra treba detektirati kada se metak sudari s neprijateljem.

Također raspravljaju o akcijama kao što su smanjenje neprijateljevog zdravlja ili uklanjanje neprijatelja iz arene.

Kako bi se vizualizirale te interakcije, nastavnik koristi UML dijagram slijeda. Dijagram ilustrira poruke koje se izmjenjuju između klasa **`Bullet`**, **`Enemy i Arena`** tijekom interakcije.

2. Implementirajte da instanca klase Bullet pogađa instancu klase Enemy

Cilj: Nastavnik vodi učenike kroz implementaciju interakcije metak – neprijatelj.

Koncepti za raspravu: detekcija kolizije, pozivanje metoda i promjene stanja objekata.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.17.** iz projekta **Tower Defense.** Učenici pišu kôd za upravljanje kolizijom između metka i neprijatelja, uključujući efekte kolizije. Nastavnik započinje objašnjavanjem koncepata detekcije kolizije i pozivanja metode u okruženju Greenfoot, naglašavajući kako su oni važni za upravljanje interakcijama između objekata igre. Učenike će provesti korak po korak kako bi implementirali logiku kolizije između metaka i neprijatelja.

Nastavnik će prvo pripremiti potrebne atribute i metode u klasama **`Bullet i Enemy`**. Učenici će pisati kôd u klasi **`Bullet`** da se detektira kolizija s instancom **`Enemy`** i pozvati metodu **`Enemy.hit(Bullet)`**. Nakon toga će implementirati metodu **`Enemy.hit(Bullet)`** da bi upravljali efektima kolizije kao što su smanjenje neprijateljevog zdravlja ili njegovo uklanjanje iz igre. Učenici će testirati svoju implementaciju kako bi se osiguralo da interakcija između metaka i neprijatelja radi kako bi trebala.

Na kraju lekcije učenici će imati funkcionalni mehanizam kolizije detekcije između metaka i neprijatelja s odgovarajućim pozivom

Co-funded by the Erasmus+ Programme of the European Union

metode i promjenama stanja objekata. Ova vježba utvrđuje njihovo razumijevanje detekcije kolizije, pozivanja metoda i praktičnu primjenu tih koncepata u razvoju igara.

Commit*:* dcfe31bc006b7f3dcd8b8b759cc1be901c32913c

3. **Objašnjavanje kôda za metode: Greenfoot.showText(String, int, int), Greenfoot.getRandomNumber(int) i World.act()**

   Cilj: Nastavnik objašnjava upotrebu specifičnih metoda u Greenfootu koje će biti korisne u igri.

   Koncepti za raspravu: prikaz teksta, generiranje slučajnih brojeva i metoda `act()`.

   Aktivnosti: Učenici uče kako prikazati tekst na ekranu, generirati slučajne brojeve i implementirati logiku igre u metodi `act()`.

   Nastavnik počinje predstavljanjem metoda `Greenfoot.showText(String, int, int)`, `Greenfoot.getRandomNumber(int)` i `World.act()`. Raspravlja se o svrsi tih metoda, naglašavajući njihovu važnost u razvoju igara za prikaz informacija, stvaranje slučajnosti i definiranja ponašanja.

   Metoda `Greenfoot.showText(String, int, int)` se koristi za prikaz teksta na ekranu na specificiranim koordinatama. Nastavnik pojašnjava da je ta metoda korisna za prikaz informacija o igri – kao što su rezultati, zdravlje ili instrukcije – direktno na ekranu igre.

   Metoda `Greenfoot.getRandomNumber(int)` generira slučajan broj između 0 (inkluzivno) i specificirane vrijednosti (ekskluzivno). Nastavnik pojašnjava kako se ta metoda može koristiti za uvođenje slučajnosti u igru, kao što su pojavljivanje neprijatelja na slučajnim lokacijama ili generiranje slučajnih uzoraka kretanja.

   Metoda `World.act()` se opetovano poziva iz okvira Greenfoot kako bi se izvršila glavna logika igre. Nastavnik naglašava da je metoda

`act()` mjesto gdje su smještene akcije glavne igre i logika, dozvoljavajući kontinuirano ažuriranje i interakcije unutar igre.

4. **Pojavljivanje neprijatelja i kraj igre**

Cilj: Nastavnik pomaže učenicima implementirati pojavljivanje neprijatelja i uvjete kraja igre.

Koncepti za raspravu: pojavljivanje objekata, petlja igre i uvjeti kraja igre.

Aktivnosti: Nastavnik učenicima zadaje zadatak **5.18.** iz projekta **Tower Defense.** Učenici pišu kôd za periodično pojavljivanje neprijatelja i definiranje uvjeta koji rezultiraju krajem igre. Nastavnik počinje pojašnjavanjem važnosti pojavljivanja neprijatelja u reguliranim intervalima i definiranjem uvjeta kraja igre kada su svi neprijatelji poraženi. Raspravlja se o konceptima pojavljivanja objekata, petlji unutar igre i uvjeta kraja igre, pružajući učenicima jasno pojašnjenje što treba biti implementirano.

Metoda `Arena.act()` će se koristiti kako bi se periodično pozivao proces pojavljivanja objekta. Trebao bi se uvesti mehanizam odgode kako bi se kontrolirao interval pojavljivanja neprijatelja. Kreirajte metodu `spawn()` u klasi `Arena` kako bi se upravljalo procesom pojavljivanja neprijatelja. Ova metoda će kreirati instancu klase `Enemy`, dodijeliti osobine neprijatelju (npr. svojstva, atributi) i dodati instancu neprijatelja u arenu.

Učenici će definirati i implementirati uvjet kraja igre. Kada se broj neprijatelja spusti na nulu, igra je završena i igrač je pobijedio. Kako bi to napravili, učenici bi trebali u atributu klase `Arena` spremiti broj kreiranih neprijatelja. Taj atribut se implementira svaki put kada se neprijatelj pojavi i smanjuje se kada je neprijatelj ubijen. Metoda `Arena.kill(Enemy)` bi se trebala prilagoditi kako bi se provjerilo jesu li su svi neprijatelji ubijeni. Ako jesu, igru se zaustavlja

Co-funded by the Erasmus+ Programme of the European Union

korištenjem metode `Greenfoot.stop()` i prikazuje se poruku pobjede. Dakle, pozivanje metode `Greenfoot.stop()` trebalo bi biti zadnja naredba.

Na kraju, mehanizam pojavljivanja testira se promatrajući periodično kreiranje neprijatelja u areni, pri čemu valja osigurati da kašnjenja između pojavljivanja funkcioniraju ispravno. Na kraju se provjerava uvjet kraja igre simulirajući poraz nad svim neprijateljima i provjerava se zaustavlja li se igra, s prikazom poruke pobjede.

Do kraja lekcije učenici će implementirati funkcionalan sustav pojavljivanja neprijatelja i jasno definirane uvjete kraja igre, što će utvrditi njihovo razumijevanje petlji igre, upravljanjima objekata i uvjetovanim rezultatima igre.

Commit*:* d48341a095561500af6032d5c8f56e201060f9a4

5. Revizija asocijacija

Cilj: Nastavnik se osvrće na koncept asocijacija između klasa, naglašavajući kako objekti međudjeluju i komuniciraju u okruženju Greenfoot.

Koncepti za raspravu: veze, komunikacija objekata i slanje poruka.

Aktivnosti: Učenici raspravljaju i obnavljaju svoje znanje, povezujući različite objekte i njihove interakcije.

Nastavnik počinje pregledom temeljnih koncepata povezanih s asocijacijama između klasa. To uključuje kako objekti međudjeluju, komuniciraju i šalju poruke jedni drugima. Kako bi to olakšao, nastavnik se može osloniti na primjere i zadatke obrađene u prethodnim lekcijama, pomažući učenicima da konsolidiraju svoje

znanje i razumiju kako se ti koncepti primjenjuju unutar Greenfoota.

Koriste se UML dijagrami slijeda kako bi se vizualno prikazale interakcije između različitih objekata. Na primjer, nastavnik prikazuje slijed poruka kada **`Bullet`** pogodi klasu **`Enemy`** i kao **`Arena`** upravlja pojavljivanjem i uklanjanjem neprijatelja.

Do kraja lekcije učenici će utvrditi svoje razumijevanje asocijacija i interakcija između objekata unutar Greenfoota. Moći će jasno artikulirati kako različiti objekti u njihovoj igri komuniciraju i kolaboriraju, primjenjujući te koncepte na svoje projekte razvoja igara.

Co-funded by the Erasmus+ Programme of the European Union

8. Nasljeđivanje

U ovoj tematskoj jedinici kreirana su četiri scenarija.

8.1. Uvod u nasljeđivanje u okruženju Greenfoot

Tablica 18. *Uvod u nasljeđivanje u okruženju Greenfoot*

Naslov	Uvod u nasljeđivanje u okruženju Greenfoot
Ishodi učenja	Do kraja lekcije učenici će moći razumjeti koncepte nasljeđivanja. Razumijevanje proučenih koncepata bit će razmatrano u kontekstu razvoja igara, potičući kreativnost, timski rad i entuzijastičan pristup kodiranju u alatu Greenfoot.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja i objektno orijentiranog programiranja, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1. Temeljni koncepti nasljeđivanja (15 min.) 2. Hijerarhija klasa i nasljeđivanje (15 min.) 3. Identifikacija zajedničkih svojstava (15 min.) i identifikacija nadklase (15 min.) 4. Uvod u apstraktne klase (5 min.) 5. Definicija apstraktne klase u igri (10 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	U ovom 75-minutnom scenariju učenici se upoznaju s principima objektno orijentiranog programiranja povezanih s nasljeđivanjem kroz prizmu razvoja igara primjenjujući

	alat Greenfoot. Lekcija počinje 15-minutnim nastavnikovim uvodom temeljnih koncepata nasljeđivanja, uspostavljajući kontekst povezan s prethodnim lekcijama i budućim razvojem igre. U 15-minutnom segmentu koji slijedi učenici i nastavnik diskutiraju o hijerarhiji klasa unutar svoje igre. Kako bi se objasnili koncepti vezani uz nasljeđivanje, razmatraju se klase **`Orb i Direction`**. Tijekom idućeg, 15-minutnog zadatka pokušavaju se identificirati zajednička svojstva za ove klase. Uočit će se da one ne djeluju tijekom svog života: jednostavno reagiraju na poruke. Posljedično, identificira se zajednička metoda za djelovanje **`act()`**, koja će biti definirana u obje klase s praznim tijelom. Temeljem identificiranih zajedničkih svojstava u idućih 15 minuta bit će implementirana nova klasa **`PassiveActor`** s metodom **`act()`**. U 5-minutnom dijelu nakon toga uvode se apstraktne klase. One služe kao predlošci za ostale klase i ne mogu se instancirati. Međutim, one su bitne u dizajniranju hijerarhije klasa. Budući da je klasa **`PassiveActor`** predložak za djelovanje, definirana je kao apstraktna klasa u idućem 10-minutnom scenariju. Dodatno, **`PassiveActor`** je postavljen kao predak klasa **`Orb i Direction`** tako da **`Orb i Direction`** postaju njegovi potomci. Budući da je metoda **`act()`** već definirana u klasi **`PassiveActor`**, uklonjena je iz klasa **`Orb i Direction`**. Kao rezultat, do kraja lekcije učenici će biti upoznati s novim konceptima povezanih s nasljeđivanjem.
Ocjenjiva-nje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe.

Co-funded by the Erasmus+ Programme of the European Union

	Uzimajući u obzir važnost koncepata nasljeđivanja, projektna struktura otvara mogućnosti za daljnju raspravu i modifikaciju. U tom kontekstu, više klasa i njihova hijerarhija se mogu razmatrati i mogu se uvesti dodatne klase, metode i atributi. S druge strane, nastavnik može prilagoditi ovu temu kako bi prikazao dobrobiti nasljeđivanja i s tim povezanom univerzalnošću samo na ovdje predloženim hijerarhijama.
Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

8.1.1. Priručnik za nastavnike za pripremu lekcije

1. Temeljni koncepti nasljeđivanja

Cilj: Uspostavljanje konteksta koji je povezan s prethodnom lekcijom uvođenjem i objašnjavanjem koncepta nasljeđivanja kroz primjere iz stvarnog života i raspravom o njegovim prednostima.

Koncepti za raspravu: nasljeđivanje, primjeri nasljeđivanja iz stvarnog života.

Aktivnosti: U uvodnom dijelu uspostavlja se kontekst povezan s prethodnim lekcijama. Nastavnik uvodi koncept nasljeđivanja, a trebao bi ga približiti učenicima koristeći primjere iz stvarnog života (npr. ako se razmatra relacija roditelj – dijete, djeca nasljeđuju karakteristike svojih roditelja kao što su tip kose, boja očiju itd.). Pritom treba raspravljati i o prednostima nasljeđivanja. Ovi

koncepti se razmatraju u kontekstu okruženja Greenfoot i programskog jezika Java.

2. **Hijerarhija klasa i nasljeđivanje**

   Cilj: Uvođenje hijerarhije klasa u kontekstu nasljeđivanja objašnjavajući klase pretke i potomke, raspravljajući o primjerima iz stvarnog života i nasljeđivanju svojstava, s naglaskom na prednostima hijerarhije klasa.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, primjeri stvarnih hijerarhija klasa, prednosti nasljeđivanja i hijerarhije klasa.

   Aktivnosti: Nastavnik uvodi hijerarhiju klasa u kontekstu koncepta nasljeđivanja, spominjući klasu pretka (poznatu kao: nadklasu, klasu roditelja) i klase potomke (poznate kao: podklase, klase djece). Pritom se može raspravljati o prethodno ispitanim klasama iz stvarnog života. U ovom kontekstu utvrdit će se da podklase mogu naslijediti svojstva (to jest atribute i metode) od klase roditelja, ali uz napomenu da podklase mogu implementirati dodatna svojstva koja nisu dostupna u roditeljskoj klasi. Zatim bi valjalo raspraviti o prednostima hijerarhije klasa u kontekstu koncepta nasljeđivanja i pojasniti da u programskom jeziku Java svaka klasa može imati više podklasa, ali samo jednu nadklasu. Primjerice, može se razmotriti uloga klase `Object` u kontekstu hijerarhije klasa i nasljeđivanja.

3. **Identifikacija zajedničkih svojstava i identifikacija klase pretka**

   Cilj: Identificiranje zajedničkih svojstava u klasama igre, traženje klase pretka i implementiranje nove klase u hijerarhiji klasa.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, implementiranje nasljeđivanja i hijerarhije klasa u razvoju igara.

   Aktivnosti: Nastavnik učenicima zadaje zadatke **6.1. i 6.2.** iz projekta **Tower Defense.** U kontekstu razvoja igara razmatraju se klase `Orb i Directon`, s napomenom da ove klase reagiraju na poruke. Stoga bi se trebala identificirati zajednička metoda za

Co-funded by the Erasmus+ Programme of the European Union

djelovanje **`act().`** Temeljem identificiranih zajedničkih svojstava, trebala bi biti implementirana nova klasa **`PassiveActor`**, koja sadrži metodu **`act()`**. Ove klase (**`PassiveActor, Orb, i Direction`**) trebale bi se koristiti za prezentiranje hijerarhije klasa u kontekstu nasljeđivanja, a nastavnik može vizualno prezentirati hijerarhiju klasa korištenjem hijerarhijskog dijagrama. Na kraju naglašava što se mijenja u Greenfootu kada se klasa **`Actor`** zamijeni klasom **`PassiveActor.`**

Commit*:* afe617814c07a5d885ed06479bf71deda8725f19

4. Uvod u apstraktne klase

Cilj: Uvođenje koncepta apstraktnih klasa, raspravljanje o njihovoj ulozi kao predlošku u dizajniranja hijerarhije klasa i istraživanje primjera iz stvarnog života da se ilustrira njihova primjena i specijalizacija u podklase.

Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, apstraktne klase, primjeri apstraktnih klasa iz stvarnog života u kontekstu nasljeđivanja i hijerarhije klasa.

Aktivnosti: Nastavnik predstavlja koncept apstraktne klase. Trebalo bi povesti raspravu o tome da apstraktne klase služe kao predložak za ostale klase i da se ne mogu instancirati te da su bitne u dizajniranju hijerarhije klasa. Nastavnik i učenici mogu navoditi stvarne primjere apstraktnih klasa i podklasa (npr. klasa **`Računalo`** s osnovnim svojstvima se može definirati kao apstraktna klasa i može se specijalizirati kao **`Konzola`**, **`StolnoRačunalo`**, **`Laptop`** i **`MobilniTelefon`**, svaka sa specifičnim skupom svojstava). Sljedeći primjer mogu biti geometrijski likovi. Pravokutnik ili trokut mogu nasljeđivati apstraktnu klasu **`Lik`**. Pri izračunavanju opsega i površine općeg lika nemamo točnu formulu, ali imamo točnu formulu za pravokutnik i trokut. Kvadrat može naslijediti pravokutnik. Učenici bi trebali prodiskutirati više primjera geometrijskih likova i tijela.

5. Definicija apstraktnih klasa u igri

Cilj: Rasprava o ulozi klase **`PassiveActor`** `i` implementiranje klase kao apstraktne.

Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, apstraktne klase, implementacija apstraktne klase u razvoju igara.

Aktivnosti: Nastavnik učenicima zadaje zadatak **6.3.** iz projekta **Tower Defense.**

Razmatra se koncept apstraktne klase u Greenfootu i programskom jeziku Java. U kontekstu razvoja igara klasa **`PassiveActor`** je predložak za djelovanje. Stoga je definirana je kao apstraktna klasa i postavljena kao predak klasa **`Orb`** i **`Direction`**, čime oni postaju potomci. Budući da je metoda **`act()`** već definirana u klasi **`PassiveActor`**, treba ju izbrisati iz klasa **`Orb i Direction.`**

Commit*:* f7a5702cae29bf21c9c88620d01ef64e4127c21c

Co-funded by the Erasmus+ Programme of the European Union

8.2. Koncepti nasljeđivanja u okruženju Greenfoot (1. dio)

Tablica 19. *Koncepti nasljeđivanja u okruženju Greenfoot*

Naslov	Koncepti nasljeđivanja u okruženju Greenfoot
Ishodi učenja	Do kraja ove lekcije učenici će razumjeti dodatne koncepte nasljeđivanja. O istraženim konceptima raspravljat će se u kontekstu razvoja igara, potičući kreativnost, timski rad i entuzijastičan pristup kodiranju u Greenfootu.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja i objektno orijentiranog programiranja, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Identifikacija zajedničkih svojstava povezanih s kretanjem entiteta (15 min.) 2) Definicija apstraktne klase povezane s kretanjem entiteta (15 min.) 3) Identifikacija svojstava specifičnih za klasu povezanih s kretanjem entiteta (15 min.) 4) Uvođenje ključne **`super`** riječi u kontekst nasljeđivanja (20 min.) 5) Refaktoriranje kôda povezano s kretanjem entiteta (30 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	Tijekom ove 95-minutne lekcije učenici se uvode u napredne koncepte nasljeđivanja koristeći Greenfoot. Cjelina počinje 15-minutnim segmentom koji istražuje zajednička svojstva povezana s kretanjem entiteta,

	fokusirajući se na klase **`Bullet i Enemy`**. One se ponašaju slično tijekom cijelog životnog ciklusa, to jest kreću se na isti način i nakon što reagiraju na okruženje. Temeljem identificiranih zajedničkih svojstava u idućem, 15-minutnom zadatku implementira se nova apstraktna klasa **`MovingActor`** koja sadrži metodu **`act()`**. Ova klasa je zajednički predak koji će implementirati metodu **`act()`** da se kreće na isti način i omogućujući svojim podklasama da se fokusiraju na svoju specifičnu svrhu. Dodatno, klasa **`MovingActor`** je postavljena kao predak klasa **`Bullet i Enemy`**, čime one postaju njezini potomci. U narednih 15 minuta istražuju se svojstva specifična za klasu povezanu s kretanjem entiteta. U tom kontekstu se ispituje metoda respektivnih klasa **`act()`**, kao i atributi **`moveDelay`** i **`nextMoveCounte`**r. Može se uočiti da je kôd metode odgovoran za kretanje isti. Nakon toga slijedi 20-minutni dio tijekom kojeg se uvodi ključna riječ **`super`** u kontekstu nasljeđivanja. U zadnjem, 30-minutnom dijelu izvodi se refaktoriranje kôda povezanog s kretanjem entiteta. Kao rezultat, prethodno identificirani atributi klasa **`moveDelay`** i **`nextMoveCounter`** iz **`Bullet i Enemy`** se prebacuju u klasu pretka **`MovingActor`**. Osim toga, parametarski konstruktor je definiran u klasi **`MovingActor`** kojom se inicijaliziraju ti atributi. Taj konstruktor s odgovarajućim parametrima je bio pozvan iz podklasa **`Bullet i Enemy`** korištenjem ključne riječ **`super.`** Nadalje, kôd odgovoran za kretanje u metodi **`act()`** u podklasama **`Bullet i Enemy`** je maknut u metodu **`act()`** klase **`MovingActor`**, dok ostatak implementacije ostaje nepromijenjen u podklasama. Konačno, roditeljska verzija metode **`act()`** poziva se u prvoj liniji metode **`act()`** u podklasama **`Bullet i Enemy`**.

Co-funded by the Erasmus+ Programme of the European Union

	Kao rezultat svega, do kraja lekcije učenici će se upoznati s novim konceptima povezanim s nasljeđivanjem.
Ocjenjivanje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe. Uzimajući u obzir važnost koncepata nasljeđivanja, projektna struktura otvara mogućnosti za daljnju raspravu i modifikaciju. U tom kontekstu može se razmatrati više klasa i njihove hijerarhije i mogu se uvesti dodatne klase, metode i atributi. S druge strane, nastavnik može prilagoditi ovu temu tako da prikaže prednosti nasljeđivanja i s tim povezanu univerzalnost samo na ovdje predloženim hijerarhijama.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

8.2.1. Priručnik za nastavnike za pripremu lekcije

1. **Identifikacija zajedničkih svojstava povezanih s kretanjem entiteta**

   Cilj: Ispitivanje klasa **`Bullet i Enemy`**, s naglaskom na njihova slična ponašanja tijekom njihovih života, naročito kako se kreću i reagiraju na okruženje.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, apstraktne klase.

Aktivnosti: Nastavnik učenicima zadaje zadatak **6.4.** iz projekta **Tower Defense.** Fokus je na klasama **`Bullet i Enemy`** koje se ponašaju slično tijekom svojeg života. Treba zamijetiti da se te klase miču na isti način i nakon toga reagiraju na okruženje.

2. **Definicija apstraktne klase povezane s kretanjem entiteta**

   Cilj: Ispitivanje **`Bullet i Enemy`** klasa, naglašavajući njihova slična svojstva tijekom njihovog života, naročito kako se kreću i reagiraju na okruženje.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, apstraktne klase, implementacija apstraktnih klasa u razvoju igara.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **6.5.** iz projekta **Tower Defense**. Temeljem identificiranih zajedničkih svojstava trebala bi biti implementirana nova apstraktna klasa **`MovingActor`** koja sadrži metodu **`act()`**. Dodatno, **`MovingActor`** je postavljen kao predak klasa **`Bullet`** i **`Enemy`**, čineći **`Bullet`** i **`Enemy`** svojim potomcima. Trebalo bi prodiskutirati da podklase nasljeđuju zajednička svojstva od roditeljske klase. Klasa **`MovingActor`** je predložak za dizajn klasa i treba se definirati kao apstraktna.

Commit*:* 43e53b533563ce0a860b294ad9009f77409c48d4

3. **Identifikacija svojstava specifičnih za klasu povezanih s kretanjem entiteta**

   Cilj: Ispitivanje klasa **`Bullet i Enemy`** klasa, s naglaskom na njihova slična svojstva tijekom njihova života, naročito kako se kreću i reagiraju na okruženje.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, apstraktne klase.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **6.6.** iz projekta **Tower Defense.** Ispituju se svojstva specifična za klasu povezana s kretanjem entiteta. Istražuje se metoda respektivnih klasa **`act()`**, kao i atributi **`moveDelay i nextMoveCounter.`** Može se uočiti da je kôd metode **`act()`** odgovoran za kretanje isti.

Co-funded by the Erasmus+ Programme of the European Union

4. **Uvođenje ključne riječi `super` u kontekst nasljeđivanja**

   Cilj: Uvođenje ključne riječi **`super`** u kontekst nasljeđivanja, demonstrirajući njezinu upotrebu u pozivanju svojstava iz roditeljske klase i raspravljajući o njezinoj prednosti.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, ključna riječ **`super`** u kontekstu nasljeđivanja, pozicija izraza **`super.`**

   Aktivnosti: Nastavnik uvodi ključnu riječ **`super`**, koja se u konceptu nasljeđivanja može koristiti u više situacija: da bi se pozvao konstruktor roditeljske klase; da bi se pozvala metoda roditeljske klase; da bi se pozvali atributi iz roditeljske klase, pri čemu **`super`** mora biti prva riječ.

   Zatim bi se trebalo raspravljati o prednosti korištenja ključne riječi **`super`** u kontekstu nasljeđivanja.

5. **Refaktoriranje kôda povezanog s kretanjem entiteta**

   Cilj: Refaktoriranje kôda povezanog s kretanjem entiteta iz podklasa **`Bullet`** i **`Enemy`** u njihovu nadklasu **`MovingActor.`**

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, ključna riječ **`super`** u kontekstu nasljeđivanja.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **6.7.** iz projekta **Tower Defense.** Izvodi se refaktoriranje kôda povezanog s kretanjem entiteta. Prethodno identificirani **`moveDelay i nextMoveCounter`** iz podklasa **`Bullet i Enemy`** se pomiču u klasu pretka **`MovingActor`**. Parametarski konstruktor koji inicijalizira te atribute je definiran u klasi **`MovingActor`**. Taj konstruktor s odgovarajućim parametrima se poziva iz podklasa **`Bullet`** i **`Enemy`** koristeći ključnu riječ **`super.`** Kôd odgovoran za kretanje u metodi **`act()`** podklasa **`Bullet i Enemy`** je maknut u **`act()`** klase **`MovingActor`**, dok ostatak implementacije ostaje nepromijenjen u podklasama. Konačno, roditeljska verzija metode **`act()`** se poziva u prvoj liniji u podklasama **`Bullet i Enemy`**. Trebalo bi prodiskutirati

da podklase mogu implementirati dodatna svojstva koja nisu dostupna u roditeljskoj klasi (to jest. različitu implementaciju metode **`act()`**).

Commit*:* ca1f010a63445c1847b74259a1c6cd4817121db3

Co-funded by the Erasmus+ Programme of the European Union

8.3. Koncepti nasljeđivanja u okruženju Greenfoot (2. dio)

Tablica 20. *Koncepti nasljeđivanja u okruženju Greenfoot*

Naslov	Koncepti nasljeđivanja u okruženju Greenfoot
Ishodi učenja	Do kraja ove cjeline učenici će razumjeti dodatne koncepte nasljeđivanja. O istraženim konceptima će se raspravljati u kontekstu razvoja igara, potičući kreativnost, timski rad i entuzijastičan pristup kodiranju unutar Greenfoota.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja i objektno orijentiranog programiranja, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Kreiranje prilagođenih neprijatelja (30 min.) 2) Uvod u Liskov Substitution Princip (20 min.) 3) Pojavljivanje prilagođenih neprijatelja (20 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	Tijekom ove 70-minutne lekcije učenici se upoznaju s naprednim konceptima nasljeđivanja koristeći Greenfoot. Lekcija počinje 30-minutnim segmentom koji se fokusira na klasu **`Enemy`** gdje učenici kreiraju dodatne podklase koje predstavljaju različite neprijatelje (npr. Frog i Spider). U tom kontekstu slike i konstruktori bez parametara (s odgovarajućim pozivom roditeljskog konstruktora) definiraju se za svaki tip neprijatelja. U narednih 20 minuta uvodi se Liskov Substitution Princip (LSP). Ovaj princip, dio principa SOLID objektno

	orijentiranog programiranja, govori da bi funkcije koje koriste pokazivače ili reference na roditeljske klase trebale koristiti objekte podklasa. Na temelju toga sljedeći zadatak od 20 minuta je posvećen pojavljivanju prilagođenih neprijatelja. Ispituje se metoda **`Arena.spawn()`**, a prilagođeni neprijatelji se kreiraju kroz različite odluke i spremaju u varijablu tipa **`Enemy`**. Može se uočiti da niti jedan kôd u aplikaciji ne treba biti promijenjen uz primjenu LSP-a. Do kraja lekcije, učenici će biti uvedeni u napredne koncepte nasljeđivanja i praktične primjene.
Ocjenjivanje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe. Uzimajući u obzir važnosti koncepata nasljeđivanja, projektna struktura otvara mogućnosti za daljnju raspravu i modifikaciju. U tom kontekstu može se razmatrati više klasa i njihova hijerarhija, a mogu se uvesti i dodatne klase, metode i atributi. Nastavnik može prilagoditi ovu temu tako da prikaže prednosti nasljeđivanja i s tim povezanu univerzalnost samo na ovdje predloženim hijerarhijama.
Rezultat diseminacije	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

Co-funded by the Erasmus+ Programme of the European Union

8.3.1. Priručnik za nastavnike za pripremu lekcije

1. **Kreiranje prilagođenih neprijatelja**

   Cilj: Definiranje podklasa klase `Enemy`, svake sa slikama i konstruktorima bez parametara koji na odgovarajući način pozivaju roditeljski konstruktor.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, ključna riječ `super` u kontekstu nasljeđivanja.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **6.8.** iz projekta **Tower Defense.** Fokus je na klasi `Enemy` i na definiciji dodatnih podklasa koje predstavljaju različite neprijatelje (npr. Frog i Spider). Slike i konstruktori bez parametara (s odgovarajućim pozivom roditeljskog konstruktora) trebali bi biti definirani za svaki tip neprijatelja.

Commit*:* b0ac1fbe793548a32f7700c292aed631918c8388

2. **Uvod u Liskov Substitution Princip**

   Cilj: Uvođenje Liskov Substitution Principa, rasprava o realnom svijetu i istraživanje prednosti slijeđenja tog principa u kontekstu nasljeđivanja.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, reference, Liskov Substitution Princip.

   Aktivnosti: Uvodi se Liskov Substitution Princip, kao princip koji je dio principa SOLID objektno orijentiranog programiranja. LSP govori da bi funkcije koje koriste pokazivače ili reference na roditeljske klase trebale moći koristiti objekte podklasa. Valjalo bi navesti primjere iz stvarnog svijeta (npr. klasa `Računalo` definirana je kao roditeljska klasa, dok su klase `Konzola`, `StolnoRačunalo`, `Laptop` i `MobilniUređaj` definirane kao podklase, a Liskov Substitution Princip govori da će funkcije koje koriste klasu `Računalo` također funkcionirati sa svim podklasama bez ijedne promjene u kôdu). U kontekstu nasljeđivanja treba povesti i raspravu o prednostima korištenja Liskov Substitution Principa.

3. **Pojavljivanje prilagođenih neprijatelja**

   Cilj: Demonstracija primjene Liskov Substitution Principa kreiranjem prilagođenih neprijatelja.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, reference, Liskov Substitution Princip.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **6.9.** iz projekta **Tower Defense.** Zadatak je posvećen pojavljivanju prilagođenih neprijatelja. Metoda `Arena.spawn()` je ispitana i prilagođeni neprijatelji su kreirani kroz različite odluke i spremljeni u varijablu tipa `Enemy`. Treba uočiti da niti jedan drugi kôd u aplikaciji ne treba biti promijenjen, demonstrirajući primjenu Liskov Substitution Principa.

Commit*:* 8cd4397f585ec957bbc18ca98e01823f434a13a6

Co-funded by the Erasmus+ Programme of the European Union

8.4. Koncepti nasljeđivanja u okruženju Greenfoot (3. dio)

Tablica 21. *Koncepti nasljeđivanja u okruženju Greenfoot*

Naslov	Koncepti nasljeđivanja u okruženju Greenfoot
Ishodi učenja	Do kraja lekcije učenici će razumjeti dodatne koncepte nasljeđivanja. O ispitanim konceptima raspravljat će se u kontekstu razvoja igara, potičući kreativnost, timski rad i entuzijastičan pristup kodiranju u Greenfootu.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja i objektno orijentiranog programiranja, koji poznaju i okruženje Greenfoot.
Trajanje scenarija	1) Raspravite hijerarhiju **`Arena`** (20 min.) 2) Napravite univerzalnu **`Arenu`** (30 min.) i kreirajte **`DemoArenu`** (15 min.) 3) Kreirajte prilagođene podklase **`Arena`** (30 min.) 4) Revizija teorije nasljeđivanja (20 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	Tijekom ove 115-minutne lekcije učenici se upoznaju s naprednim konceptima nasljeđivanja koristeći Greenfoot. Lekcija započinje 20-minutnom raspravom o hijerarhiji **`Arena.`** Podklase `Arena` su odgovorne za prilagođene izglede (npr. pozicija instanci **`Orb i Direction`**, veličina arene). Ti zadaci se izvode u konstruktorima podklasa koji postavljaju i spremaju pojavljujuće pozicije, rotacije i dimenzije arene.

	U 30-minutnom zadatku koji slijedi uvodi se univerzalna klasa **`Arena`**. Dodatni atributi (**`spawnPositionX`**, **`spawnPositionY`**, **`ispawnRotation`**) definirani su, inicijalizirani u konstruktoru i korišteni u metodama **`spawn()`** i **`respawn(Enemy)`**. Atributi vezani uz dimenzije arene (**`width`** i **`height`**) također su definirani i inicijalizirani u konstruktoru. Budući da klasa **`Arena`** služi kao predložak za definiranje konkretnih arena, definira se kao apstraktna klasa. Temeljem identificirane klase **`Arena`** u 15-minutnom zadatku nakon toga uvodi se podklasa **`DemoArena`**. Konstruktor za **`DemoArenu`** je definiran tako da poziva konstruktor roditeljske klase i kôd odgovoran za izgled smjerova, kugli i tornjeva je pomaknut iz konstruktora za **`Arenu`** u klasu konstruktora **`DemoArene`**. Konačno, nova instanca klase **`DemoArena`** je kreirana. U idućih 30 minuta kreiraju se ostale inovativne podklase klase **`Arena`**. Kôd se može dijeliti s ostalim učenicima u grupi. Zatim se narednih 20 minuta posvećuje teoriji povezanoj s nasljeđivanjem. Kao rezultat, do kraja lekcije učenici su upoznati s naprednim konceptima nasljeđivanja i praktičnom primjenom.
Ocjenjiva-nje	Igrifikacija predstavlja neformalno ocjenjivanje, ali će povećati interes, intrinzičnu motivaciju i ishode učenja cijele grupe. Uzimajući u obzir važnost koncepata nasljeđivanja, projektna struktura otvara mogućnosti za daljnju raspravu i izmjenu. U tom kontekstu može se razmatrati više klasa i njihova hijerarhija, a mogu se uvesti i dodatne klase, metode i atributi. Nastavnik može prilagoditi ovu temu da

Co-funded by the Erasmus+ Programme of the European Union

	prikaže prednosti nasljeđivanja i s tim povezanom univerzalnošću samo ovdje predložene hijerarhije.
Disemina-cija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

8.4.1. Priručnik za nastavnike za pripremu lekcije

1. **Raspravite hijerarhiju `Arena`**

   Ciljevi: Istraživanje hijerarhije klasa **`Arena`**, naglašavajući kako su podklase odgovorne za definiranje prilagođenih izgleda i implementiranje tih izgleda s respektivnim konstruktorima.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, konstruktori.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **6.10.** iz projekta **Tower Defense.** Raspravlja se o hijerarhiji klasa **`Arena`**. Treba uočiti da su podklase **`Arene`** odgovorne za prilagođen izgled (npr. pozicije instanci **`Orb i Direction`**, veličina arene). Ti zadaci se izvode u konstruktorima podklasa, koje postavljaju i spremaju pozicije pojavljivanja, rotacije i dimenzije arena.

2. **Napravite univerzalnu `Arenu` i kreirajte `DemoArenu`**

   Cilj: Uvođenje univerzalne apstraktne klase **`Arena`**, definiranje i inicijaliziranje konkretne podklase **`Arena`** i istraga hijerarhije klasa **`Arena`**.

   Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, apstraktne klase, konstruktori.

Aktivnosti: Nastavnik učenicima zadaje zadatke **6.11. i 6.12.** iz projekta **Tower Defense.** Na temelju prethodnih rasprava uvodi se univerzalna klasa **`Arena`**. Definirani su dodatni atributi (**`spawnPositionX, spawnPositionY i spawnRotation`**), inicijalizirani u konstruktoru i korišteni u metodama **`spawn() i respawn(Enemy)`**. Atributi povezani s dimenzijama arene (**`width`** i **`height`**) također su definirani i inicijalizirani u konstruktoru. Budući da klasa **`Arena`** služi kao predložak za definiranje konkretnih **`Arena`**, definirana je kao apstraktna klasa.

Commit*:* e9844d7d9b5f19969618b469ebc907d0fe3c1357

Na temelju identificirane klase **`Arena`** definirana je podklasa **`DemoArena`**. Konstruktor **`DemoArena`** također je definiran i on poziva konstruktor roditeljske klase, a kôd odgovoran za izgled smjerova, kugli i tornjeva je pomaknut od konstruktora **`Arena`** u konstruktor **`DemoArena`**. Konačno, kreirana je nova instanca klase **`DemoArena`**. Kako bi se aktivirala **`DemoArena`**, kliknite desnom tipkom miša i odaberite novu **`DemoArenu`**.

Commit*:* 6a6569774b5735f453a56c7cb2cdbf19d228eae9

3. **Kreirajte prilagođene podklase `Arena`**

Cilj: Definiranje i inicijaliziranje prilagođenih podklasa **`Arena`**, istražujući hijerarhiju klasa **`Arena`**.

Koncepti za raspravu: nasljeđivanje, hijerarhija klasa, apstraktne klase, konstruktori.

Aktivnosti: Nastavnik učenicima zadaje zadatak **6.13.** iz projekta **Tower Defense.** Kreirane su inovativne podklase klase **`Arena`**. Kôd se može dijeliti s ostalim učenicima unutar grupe.

4. **Revizija teorije nasljeđivanja**

Cilj: Pregled koncepta i prednosti nasljeđivanja, rasprava o hijerarhiji klasa i apstraktnih klasa, istraživanje ključne riječi **`super`**, Liskov Substitution principa, zajedno s njihovim prednostima i

Co-funded by the Erasmus+ Programme of the European Union

ispitujući primjere nasljeđivanja i njihove implementacije u realnom svijetu i svijetu igara.

Koncepti za raspravu: Nasljeđivanje, hijerarhija klasa, apstraktne klase, ključna riječ **`super`**, Liskov Substitution Princip, primjeri i implementacija nasljeđivanja u stvarnom svijetu i svijetu igara.

Aktivnosti: Razmatra se koncept nasljeđivanja i njegove prednosti, hijerarhija klasa i njene prednosti u kontekstu koncepta nasljeđivanja. Raspravlja se koncept apstraktne klase te prednosti ključne riječi **`super`** u kontekstu nasljeđivanja, a proučava se i Liskov Substitution Princip i prednosti njegova korištenja u kontekstu nasljeđivanja. Ispituju se primjeri nasljeđivanja iz stvarnog života, kao i primjeri nasljeđivanja iz svijeta igara.

9. Enkapsulacija

U ovoj tematskoj cjelini kreirana su dva scenarija poučavanja.

9.1. Istraživanje enkapsulacije kroz razvoj igara alatom Greenfoot (1. dio)

Tablica 22. *Istraživanje enkapsulacije kroz razvoj igara alatom Greenfoot (1. dio)*

Naslov	Istraživanje enkapsulacije kroz razvoj igara alatom Greenfoot (1. dio)
Ishodi učenja	Svrha ove nastavne cjeline je predstavljanje učenicima enkapsulacije kroz daljnji razvoj igre **Tower Defense**.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja i objektno orijentiranog programiranja, koji su upoznati i s okruženjem Greenfoot.
Trajanje scenarija	1. Uvod (5 min.) 2. Rad u timu i kôdiranje (20 min.) 3. Korištenje privatnih metoda i enkapsulacije za upravljanje stanjem objekta (30 min.) 4. Rasprava (35 min.) 5. Objašnjavanje kôda (25 min.) 6. Primjena privatnih metoda i enkapsulacije u kontroli tornja (10 min.)
Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	U ovoj lekciji učenici će naučiti o enkapsulaciji u objektno orijentiranom programiranju u alatu Greenfoot. Lekcija

Co-funded by the Erasmus+ Programme of the European Union

	počinje sažetim 5-minutnim uvodom u kojem nastavnik naglašava ishode. Učenici počinju s razvijanjem **`ManualTower`** kao klase i podklase klase **`Tower`**. Aktivnost se fokusira na definiranje dva konstruktora koji su konzistentni s konstruktorom roditeljske klase kako bi se osigurala pravilna inicijalizacija. Trebaju implementirati metodu **`act()`** koja prvo zove metodu **`act()`** roditeljske klase. Slijedeći kreiranje klasa, nastavnik uvodi atribut **`boolean isManuallyControlled`**, koji je inicijaliziran na **`false.`** Učenici kreiraju metodu **`changeControl(boolean)`** koja mijenja stanje **`isManuallyControlled`** i sliku tornja, demonstrirajući enkapsulaciju kontroliranjem pristupa stanju objekta kroz metode. Svaki učenik bi trebao tada ručno pokrenuti metodu **`changeControl()`** na instancama klase **`ManualTower`** i razmotriti kako se mijenja interno stanje i vanjska reprezentacija. Srž ove lekcije je razvoj privatne metode **`processUserControl()`** koja bi trebala detektirati klikove mišem na instancama tornja. Kada je kliknut, metoda mijenja kontrolno stanje tornja i ažurira njegovu orijentaciju na temelju pozicije miša, koristeći enkapsulaciju kako bi se sakrila kompleksna kontrolna logika. Učenici bi trebali implementirati metodu i integrirati je unutar metode **`act()`**, testirajući interakciju s okruženjem igre kako bi se osigurala funkcionalnost i učeći kako privatne metode štite kôd od vanjskih promjena.
Ocjenjiva-nje	Ova će aktivnost omogućiti nastavnicima da daju formativne povratne informacije o ocjenjivanju na temelju rasprava i praćenja obrnute učionice i timskog rada učenika.

	Međusobno ocjenjivanje će biti provedeno online kao dio domaće zadaće. To će podsjetiti učenike na važne aspekte vježbe, morat će kritički ocijeniti rad ostalih učenika, dat će im uvid u dobra i ne tako dobra rješenja i povećat će ukupno postignuće ishoda učenja. Rad na timskom projektu također će pozitivno djelovati na ishode učenja i sveukupno znanje.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na forumu koji im je dostupan putem alata za upravljanje učenjem.

9.1.1. Priručnik za nastavnike za pripremu lekcije

1. Uvod

Nastavnik bi trebao početi prethodno razvijenu igru i razmotriti kako se različiti učesnici ponašaju te sugerirati razvoj drugačijeg tipa tornja koji se može ručno kontrolirati da bi se lagano uklonili neprijatelji. Korisnik bi trebao moći kontrolirati jedan toranj istovremeno. Kada je toranj kliknut, trebao bi se ručno kontrolirati. Kako bi se indiciralo kojim se tornjem ručno kontrolira, trenutno kontrolirani toranj bi trebao imati drugačiji izgled.

2. Rad u timu i kôdiranje

Cilj: Priprema klase `ManualTower` za lekciju.

Koncepti za raspravu: nasljeđivanje, klase, konstruktori.

Aktivnosti: Nastavnik učenicima zadaje zadatak **7.1.** iz projekta **Tower Defense.** Budući da učenici već znaju kako napraviti podklasu, neka formiraju timove i kreiraju klasu `ManualTower` kao nasljednika klase `Tower`. Učenici bi trebali implementirati oba

Co-funded by the Erasmus+ Programme of the European Union

konstruktora i metodu, osiguravajući da se konstruktori iz roditeljskih klasa pozivaju iz tih metoda. U ovom dijelu učenici će pregledati materijal, primijeniti ga i poboljšati svoje praktično znanje nasljeđivanja.

Commit*:* 63a02fa0c5080165cba8b467da08c4b65f31d0a8

3. Korištenje privatnih metoda i enkapsulacije za upravljanje stanjem objekta

Cilj: Nastavnik objašnjava potrebu za privatnim metodama učenicima definirajući funkciju **`changeControl()`**.

Koncepti za raspravu: metode, klase, atributi, modifikatori pristupa

Aktivnosti: Nastavnik učenicima zadaje zadatke **7.2. i 7.3.** iz projekta **Tower Defen**Nastavnik bi trebao pripremiti ikone za ručno kontrolirani toranj. Da bi se programski promijenila ikona, trebao bi objasniti učenicima kako da koriste metodu **`Actor.setImage(String)`** i dati im neko vrijeme da testiraju tu funkciju.

Zatim bi trebao s učenicima razmotriti kako odrediti je li toranj ručno kontroliran, naglašavajući da nije važno samo promijeniti stanje objekta, nego valja promijeniti i njegovu sliku. Pritom treba istaknuti da će, ako korisnik želi stanje objekta **`Tower`** i samo mijenja stanje atributa direktno, slika ostati ista. Sve navedeno trebalo bi pomoći učenicima da razumiju potrebu za promjenom vrijednosti atributa kroz metodu i da postavljaju atribute kao privatne, a ne kao javne. Ta se praksa naziva enkapsulacija i u njoj je interno stanje sakriveno, a javne metode se koriste kako bi se promijenilo stanje na kontroliran način.

Učenici će na kraju implementirati logiku funkcije i ručno pozvati svoje metode te promotriti promjene u internom stanju.

Commit*:* 2257746b7dac5eaab7acc55d6493319230338f3a

4. **Rasprava**

   Cilj: Razumijevanje logike enkapsulacije unutar odvojene metode.

   Koncepti za raspravu: metode, grananja.

   Aktivnosti: Nastavnik bi trebao istaknuti da se stanje tornja može promijeniti ručno pozivanjem metode, s napomenom da miš može biti izvan svijeta, a tada će informacija miša biti nula. Podsjetit će učenike da se metoda `act()` konstantno vrti tijekom igre i da treba provjeriti je li objekt bio kliknut te samo tada pozvati metodu `changeControl()`. Naglasit će da bi logika za obradu kontrole trebala biti enkapsulirana unutar odvojene metode `processUserControl()`.

5. **Objašnjavanje kôda**

   Cilj: Uvođenje metode potrebne za rješavanje problema.

   Koncepti sa raspravu: metode, okruženje Greenfoot.

   Aktivnosti: Učitelj bi s učenicima trebao razmotriti kako promijeniti stanje učesnika klikanjem na taj objekt. Da se to implementira, valja objasniti metodu `GreenFoot.mouseClicked(Object)`. Također, uvodi se `MouseInfoobject`, koji se može koristiti za vraćanje informacija o poziciji miša.

6. **Primjena privatnih metoda i enkapsulacije u kontroli tornja**

   Cilj: Poboljšati razumijevanje privatnih metoda i enkapsulacije kroz praktični zadatak.

   Koncepti za raspravu: metode, modifikatori pristupa, klase.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **7.4.** iz projekta **Tower Defense.** Nakon definiranja privatne metode `processUserControl()` učenici bi trebali implementirati njezinu logiku. Kada je miš pritisnut , kontrolirani toranj bi se trebao promijeniti. Ako je toranj ručno kontroliran, trebao bi slijediti i biti usmjeren prema mišu. Pritom se valja prisjetiti i kako da je moguće da miš bude izvan svijeta. Nakon formiranja timova učenici će implementirati logiku metode `processUserControl()`.

Co-funded by the Erasmus+ Programme of the European Union

Jedan ili dva tima će prezentirati svoj rad, a grupa će raspraviti rezultate zajedno s nastavnikom. Do kraja lekcije svi učenici bi trebali razumjeti kako je ta metoda implementirana.

Commit*:* 6ec1f489576019a6493490f9e97797920b923869

9.2. Istraživanje enkapsulacije kroz razvoj igara alatom Greenfoot (2. dio)

Tablica 23. *Istraživanje enkapsulacije kroz razvoj igara alatom Greenfoot (2. dio)*

Naslov	Istraživanje enkapsulacije kroz razvoj igara alatom Greenfoot (2. dio)
Ishodi učenja	Svrha ovog nastavne cjeline je predstaviti enkapsulaciju učenicima kroz daljnji razvoj igre **Tower Defense**.
Ciljana publika	Učenici koji pohađaju predmet OOP4FUN, s osnovnim znanjem programiranja i objektno orijentiranog programiranja, koji su upoznati i s okruženjem Greenfoot.
Trajanje scenarija	1) Obrnuta učionica: Identifikacija i rješavanje problema korisničke kontrole (30 min.) 2) Atributi klasa (5 min.) 3) Dodati dokaz ručno kontroliranog tornja (5 min.) 4) Metoda klase (10 min.) 5) Promijeniti ručno kontroliran toranj s centraliziranog mjesta (20 min.) 6) Pozvati promjenu ručno kontroliranog tornja (15 min.) 7) Revizija teorije (10 min.)

Materijali i resursi	Udžbenik iz projekta OOP4FUN. Resursi iz projekta OOP4FUN. Izvorni kôd projekta s GitHuba/GitLaba. Internetski resursi.
Opis	Lekcija počinje raspravom učenika i pronalaženjem problema povezanih s korisničkom kontrolom kao što su nemogućnost deselektiranja tornja jedanput kada je odabran, a zatim će predložiti rješenja da prate trenutno kontroliran toranj i modificiraju klasu `ManualTower` da uključe mehanizam deselektiranja tornja. Posljednje, metoda klase `changeControlledInstance()` bi trebala biti implementirana jer omogućava promjenu kontrole tornjeva iz centralizirane metode, poboljšavajući razumijevanje enkapsulacije i pokazujući kako metode klase mogu upravljati dijeljenim stanjem kroz instance. Ovaj opsežan edukacijski pristup podučava koncept enkapsulacije i demonstrira svoju važnost i korisnost u aplikacijama realnog svijeta, razvijajući vještine rješavanja problema i suradnju među učenicima.
Ocjenjivanje	Aktivnost će omogućiti nastavnicima da daju formativnu povratnu informaciju na temelju rasprava i praćenja učeničkih obrnutih učionica i timskog rada. Učenici će prakticirati rad na timskom projektu, kao i ishode učenja i znanje.
Diseminacija rezultata	Za predstavljanje vlastitih rezultata nastavniku i drugim učenicima koristit će se uobičajeno postavljanje na GitHub/GitLab i sustav za upravljanje učenjem (npr. Moodle), a učenici mogu nastaviti raspravu o temi na

Co-funded by the Erasmus+ Programme of the European Union

	forumu koji im je dostupan putem alata za upravljanje učenjem.

9.2.1. Priručnik za nastavnike za pripremu lekcije

1. **Obrnuta učionica: Identifikacija i rješavanje problema korisničke kontrole**

   Cilj: Učenici bi trebali prepoznati potrebu inicijaliziranja atributa na jednome mjestu.

   Koncepti za raspravu: atributi klasa, prethodno implementirane korisničke kontrole

   Aktivnosti Na početku učenici identificiraju problem s korisničkom kontrolom: trenutno nije moguće deselektirati toranj. Nastavnik će ih potaknuti da razmisle o tome kako bi taj problem mogao biti riješen i pojašnjava da bi u igri samo jedan toranj trebao biti odabran istovremeno. Rasprava bi trebala voditi učenike do ideje da imaju jedno mjesto u programu koje je inicijalizirano samo jednom, a ostali objekti i učesnici mogu mu pristupiti iz ostalih dijelova programa .

2. **Atributi klasa**

   Cilj: Uvesti temelje atributa klasa.

   Koncepti za raspravu: atributi, atributi klasa, klase.

   Aktivnosti: Nastavnik objašnjava što su atributi klasa: varijable koje pripadaju klasi, a ne instance klasa, te povezuje taj koncept sa scenarijem igre raspravljenim prije, gdje je postojanje središnjeg atributa za upravljanje trenutno odabranim tornjem riješilo problem.

3. **Dodajte dokaz ručno kontroliranog tornja**

   Cilj: Praktična primjena atributa klasa.

   Koncepti za raspravu: atributi, atributi klasa, klase, modifikatori pristupa

   Aktivnosti: Nastavnik učenicima zadaje zadatak **7.6.** iz projekta **Tower Defense.** Kako bi se pratio koji toranj je trenutno odabran u igri, dodaje se privatni statički atribut `controlledInstance` klasi `ManualTower` te se inicijalizira na nulu. Statički atribut je povezan s cijelom klasom, a ne se objektom klase. Dakle, definiranje statičke varijable će omogućiti da se odredi je li toranj bio odabran i, ako jest, koji je to, referencirajući ime klase bez potrebe da se pristupi objektu. Nastavnik bi trebao naglasiti da postoji jedan atribut, `controlledInstance`, za cijelu igru. Na početku `controlledInstance` bi trebao biti inicijaliziran na nulu, budući da ne postoji selektiran toranj. Zatim slijedi istraživanje internog stanja klase, pri čemu nastavnik objašnjava razlike između statičkih i nestatičkih atributa. S učenicima razmatra prednosti korištenja statičkih atributa u igri, a trebao bi spomenuti i statičke metode te raspraviti s učenicima kada ih je dobro koristiti.

Commit*:* c4739460bed583d2126de066acc6b1149d022990

4. **Klasa metode**

   Cilj: Uvesti osnove metoda klasa.

   Koncepti za raspravu: metode klasa, klasa, objekti.

   Aktivnosti: Nastavnik prezentira koncept metoda klasa koje mogu operirati podacima na razini klasa i s učenicima razmatra potrebu za metodama kao što je `changeControlledInstance` za upravljanje promjenama trenutno kontroliranog tornja. Naglašava da te metode mogu biti pozvane bez potrebe za instancom klase. Na primjer, školsko zvono zvoni za sve u isto vrijeme, nije bitno tko je

Co-funded by the Erasmus+ Programme of the European Union

tko, ali, s druge strane, provjera učeničkih zadaća zahtijeva informaciju o određenom učeniku.

5. **Promijenite ručno kontrolirani toranj iz centraliziranog mjesta**

   Cilj: Praktična primjena metoda klasa.

   Koncepti za raspravu: metode, metode klasa, atributi klasa.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **7.7.** iz projekta **Tower Defense.** Nastavnik bi trebao dodati metodu `changeControlledInstance()` kako bi se promijenio ručno kontrolirani toranj. Parametar metode će biti toranj koji korisnik hoće promijeniti. Prvo treba provjeriti je li kontrolirana instanca trenutno odabrana. Ako jest, ništa se ne bi trebalo promijeniti, ali ako je proslijeđena instanca različita, tada bi se trebala promijeniti trenutno kontrolirana instanca (referenca na trenutno kontroliranu instancu bi se trebala promijeniti). Zatim bi valjalo testirati funkciju ručno i uvidjeti kako se ikone tornja ne mijenjaju te istaknuti da se samo promjenom reference kontrolirane instance ne mijenja kontrola i da treba biti napravljena ručno. Potom treba dodati kôd koji otpušta trenutno kontroliranu instancu i nakon ažuriranja reference dodati kôd koji postavlja ručnu kontrolu ručno kontrolirane instance. Pritom valja naglasiti i potrebu za provjerom reference nula, koje bi se mogle pojaviti ako nema trenutno kontrolirane instance (kada je parametar nula).

Commit*:* 9dc6d8dd4dcbbd71edb8009c1a72403dea1a0ee0

6. **Pozovite promjenu ručno kontroliranog tornja**

   Cilj: Praktična primjena metoda klasa.

   Koncepti za raspravu: metode, metode klase, atributi klase.

   Aktivnosti: Nastavnik učenicima zadaje zadatak **7.8.** iz projekta **Tower Defense.** Ručno testirajte ispravnost funkcije, radi korektno**.** Nakon toga, diskutirajte sa učenicima kada bi ta funkcija

trebala biti pozvana. Metoda bi trebala biti pozvana unutar Arenine **`act()`** funkcije i unutar **`userControl()`** funkcije. Naposljetku, učinite metodu **`ManualTower.changeControl(boolean`**) privatnom i promatrajte promjene instance **`ManualTower.`**

Commit*:* c052bbb6aa4c7e690d4d8cf55d3831028fa2b9e3

7. Revizija teorije

Nastavnik će sumirati ovu cjelinu, s naglaskom na važnosti atributa klasa i metoda u efikasnom upravljanju logikom igre. Ohrabrit će učenike da i dalje istražuju njihovu primjenu u svojim programerskim konceptima.

oop4fun.eu

2021-1-SK01-KA220-SCH-00027903

Co-funded by the
Erasmus+ Programme
of the European Union

unizg.hr

foi.unizg.hr

Financirano od strane Europske unije. Stavovi i mišljenja izražena u tekstu pripadaju isključivo autoru(ima) i ne odražavaju nužno stavove Europske unije ili tijela koje je dodijelilo sredstva. Europska unija niti tijelo koje je dodijelilo sredstva ne mogu snositi odgovornost za njih.

oop4fun@fri.uniza.sk

@oop4fun

TISKANO IZDANJE
ISBN 978-953-6071-80-7

E-IZDANJE
ISBN 978-953-6071-81-4