PROPOSAL OF NEW SYLLABUS FOR ACCREDITATION BODIES IN THE SECONDARY LEVEL

PR5 - REPORT

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

Project	Object Oriented Programming for Fun
Project acronym	OOP4FUN
Agreement number	2021-1-SK01-KA220-SCH-00027903
Project coordinator	Žilinska univerzita v Žiline (Slovakia)
Project partners	Sveučilište u Zagrebu (Croatia) Srednja škola Ivanec (Croatia) Univerzita Pardubice (Czech Republic) Gymnazium Pardubice (Czech Republic) Obchodna akademia Povazska Bystrica (Slovakia) Hochschule fuer Technik und Wirtschaft Dresden (Germany) Gymnasium Dresden-Plauen (Germany) Univerzitet u Beogradu (Serbia) Gimnazija Ivanjica (Serbia)
Year of publication	2024

Disclaimer:
Funded by the European Union. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or Slovak Academic Association for International Cooperation. Neither the European Union nor the granting authority can be held responsible for them.

Table of contents

1. Introduction

The aim of this project result was to prepare documents for country-wise accreditation of the new syllabus with included project results as well textbook accreditation. As there are different rules in different countries the project result 5 should show the possible transferability all project results between the consortium countries. The project result set contains not only documents but also syllabus adapted to each country with light OOP topics as well as description of practices and procedures in each country.

2. Country-wise accreditation

Each high school prepared in English and native language document for syllabus accreditation and literature approval. English version for each country is in the following sections of this chapter.

2.1. Accreditation and literature approval in Croatia

The activity of secondary education is carried out by secondary schools and other public institutions, and includes various types and forms of education, training and improvement that are carried out in accordance with the provisions of the Law on Education in Primary and Secondary Schools (Official Gazette, No. 87/2008, 86/2009, 92/2010, 105/2010-corrected, 90/2011, 16/2012, 86/2012, 94/2013, 152/2014, 7/2017, 68/2018, 98/2019, 64/2020, 151/2022 and 156/2023).

Secondary education programs are:

- programs for obtaining a lower level of secondary education;
- programs for acquiring secondary education;
- training and development programs.

Secondary schools, depending on the type of educational program, are:

- gymnasiums;
- vocational schools;
- art schools.

Gymnasiums are four-year general education schools that students finish by taking the state matriculation exam. In gymnasiums, students acquire competencies (knowledge and skills) from general knowledge, which represents a quality basis for continuing education at higher education institutions.

The education program in gymnasiums is enriched with electable and optional subjects and extracurricular activities that schools offer in accordance with their material and personnel capabilities. There are five types of gymnasium educational programs that differ in the increased number of hours of those subjects that are characteristic for each program, namely:

- general gymnasium (grammar school);
- language oriented gymnasium;
- classical gymnasium;
- science and mathematics oriented gymnasium;
- science oriented gymnasium;

In gymnasiums (all types), the only subject related to programming and informatics in general is the subject named Informatics. Depending on the type of gymnasiums, this subject is taught from 1 to 4 years, either as a compulsory or electable subject.

Curricula and programs of compulsory, electable and cross-subject and/or interdisciplinary content and/or modules are adopted by the minister by decision, and the optional part is adopted by the school through its annual curriculum.

The curriculum of the Informatics subject is provided by the Ministry of Science, Education and Youth, and the decision on adopting the curriculum is available at the following link (Official Gazette, No. 22/2018):

https://narodne-novine.nn.hr/clanci/sluzbeni/2018_03_22_436.html

The curriculum is also available on the website of the Ministry of Science, Education and Youth:

https://mzom.gov.hr/UserDocsImages/dokumenti/Publikacije/Predmetni/Kurikulum%20nastavnog%20predmeta%20Informatika%20za%20osnovne%20skole%20i%20gimnazije.pdf

The general gymnasium is by far the most represented of all other gymnasium types, and more than half of the students who enroll in gymnasium education enroll in the general gymnasium course. A detailed study of the availability of secondary school programs in Croatia from 2013 to 2021 was conducted by the Institute for Social Research in Zagreb and can be viewed at the following link:

https://mzom.gov.hr/UserDocsImages/dokumenti/Obrazovanje/NacionalniPlan2021-2027/Studija-srednje-obrazovanje.pdf

Percentages of enrolled students by types of gymnasium programs in the school years 2021/2022. can be seen in this table:

Type of gymnasium	**Percentage of enrolled students**
general gymnasium	57.7%
language oriented gymnasium	14.5%
classical gymnasium	5.7%
science and mathematics oriented gymnasium	17.3%
science oriented gymnasium	3.7%
Other gymnasium types (experimental programs)	1.1%

In the general gymnasium, which is the most represented of all the gymnasiums, the Informatics subject is taught in 2 forms: as a compulsory subject in the 1st grade and as an electable subject in the remaining 3 grades.

As can be seen in the curriculum of Informatics subject for general gymnasiums, the learning outcomes that are adopted in the areas of programming, computational thinking and problem solving are as follows:

Year/grade (type of subject)	Learning outcome	Elaboration of learning outcomes
1st (compulsory)	The student analyzes the problem, defines input and output values and observes the steps to solve the problem.	The student: • analyzes the problem, determines the type and scope of input data, considers ways to solve the problem • notices separate entities and breaks down the problem into smaller parts • shows a step-by-step troubleshooting process • recognizes basic algorithmic structures in its algorithm: sequence, branching and repetition • discusses the correctness of the algorithm and changes it if necessary • recognizes the limitations of the algorithm
	The student applies simple data types and arguments their selection, applies different types of expressions, operations, relations and standard functions for modeling a simple problem in the selected programming language.	The student: • selects the type of data suitable for solving the given problem • describes the operation of certain mathematical and logical operations • applies standard operations and functions on simple data types when solving a given problem, as well as input and output functions • determines the result of mathematical and logical expression • analyzes expressions • models a simple problem with appropriate expressions • arguments the advantages of using the selected data type with respect to other data types
	The student develops an algorithm and creates a program in the selected programming language, solving the problem using a branch and repetition structure.	The student: • develops an algorithm by applying structures of repetition (with a predetermined number of repetitions and conditionally) and branching • follows the execution of the algorithm, implements it in the selected programming language, tests and evaluates in cooperation with others, uses different methods of error correction • values different solutions to the same problem
2nd (electable)	The student analyzes basic algorithms with simple data types and basic program structures and applies them to solving new problems.	The student: • explains the basic idea of several key algorithms (adding/multiplying natural numbers within a certain interval, entering and adding/multiplying n numbers, checking if the number is prime, the largest entered value, etc.) • considers other ways of solving the same problem • analyzes the efficiency of the algorithm depending on the amount and type of input values • applies known algorithms when solving new problems
	In a given problem, the student observes smaller components, solves them, and then integrated them into a unique solution to the problem.	The student: • breaks down the given problem into smaller functional units that it describes • determines the input and output parameters of the functional units • distinguishes between global, local and formal variables • solves functional units in a specific programming language using programming functions correctly and integrates them into a complete problem solution • collaborates in team problem solving
	The student solves the problem using one-dimensional data structure.	The student: • describes the one-dimensional data structure of the given programming language • explains the role of indexes

		• notices the possibility of using a one-dimensional data structure when solving a given problem • describes and uses standard functions and methods for working with a selected one-dimensional data structure • arguments the selection of a one-dimensional data structure for solving a given problem
	In cooperation with others, the student designs the algorithm, implements it in the chosen programming language, tests the program, documents and presents the possibilities and limitations of the program to others.	The student: • in cooperation with others, considers the problem, designs an algorithm and develops a conceptual solution • tests and documents the conceptual solution • based on the conceptual solution, he/she develops, tests and documents the software solution • presents to others the possibilities and limitations of the software solution • evaluates the success of the software solution
3rd (electable)	The student applies standard algorithms defined over integer numbers.	The student: • applies an algorithm for searching for the largest and smallest number, an algorithm for replacing the values of two variables, an algorithm for dividing a number into digits, an algorithm for checking the complexity of a number, and the Euclid algorithm
	The student analyzes data sorting as an important concept for solving different problems.	The student: • describes and applies some of the simple algorithms for sorting and searching data • applies sorting as part of a problem-solving strategy
	Using a graphic module, the student visualizes and graphically represents some problem from his environment.	The student: • applies the basic functions of the selected graphic module when drawing compositions of shapes of different complexity and for displaying mathematical functions • visualizes and graphically shows a real problem
	The student solves the problem using complex data types defined by the given programming language.	The student: • describes complex data types • describes the basic methods and functions of complex data types • observes the possibility of using complex data types in a given problem • solves a given problem by applying methods and functions of a complex data type
	The student defines a real-life problem and creates a programming solution by going through all programming phases.	The student: • defines a real-life problem • analyzes the problem and breaks it down into smaller parts • uses the capabilities of the programming language to solve a specific problem • documents the software solution and presents it to others
4th (electable)	The student solves the problem using a recursive function.	The student: • describes the basic elements of the recursive procedure • writes a mathematically described recursive function in a programming language • detects recursiveness in a given problem, determines the recursive relation and termination condition, and realizes the recursive function in the programming language • evaluates the efficiency of the recursive solution • depending on the problem, chooses a recursive or inductive solution

	The student compares different data sorting and searching algorithms.	The student: ● describes and applies standard data sorting and searching algorithms ● applies sorting as part of a problem-solving strategy ● arguments the use of faster sorting and searching algorithms by giving examples
	The student designs an object model with associated complex data structures and implements it in the default programming language.	The student: ● describes the basic concepts associated with object-oriented programming (class, object, property, method ...) ● within the given problem, it observes the basic properties and methods and forms the corresponding class in the specific programming language ● explains the concept of class inheritance ● models a more complex problem and implements the solution in a programming language
	The student defines a real-life problem and creates a programming solution by going through all programming phases. It presents the software solution and values it.	The student: ● defines a real-life problem ● analyzes the problem and breaks it down into smaller parts ● uses the capabilities of the programming language to solve a specific problem ● documents the software solution and presents it to others ● together with others, evaluates the success of the software solution

It is obvious (as prescribed by the subject curriculum for general gymnasiums) that object-oriented programming is represented to a very small extent and only in the 4th grade as an electable subject, as single learning outcome.

The situation in High school Ivanec is that Informatics is compulsory in the 1st grade, and electable subject in the 2nd and 3rd grades (in 4th grade there is no Informatics subject available[1]).

This means that our students at High School Ivanec do not encounter object-oriented programming content to any extent. For this reason, the only possible solution is to **create a new curriculum of an optional subject**. In this way, our students will be able to familiarize themselves with basic OOP concepts, and students from other schools who have Informatics in the 4th grade and encounter the basics of object-oriented programming through electable subject, will be able to acquire and expand their knowledge of OOP concepts in a more fun and simple way.

So, High School Ivanec plans to create a curriculum for the **new optional subject** named *Object-oriented programming through game development* for 32 hours per year (1 school hour per week) based on the syllabus developed under PR3 and the teacher's textbook developed under PR4. The subject is intended for the students of the 4th grade of high school (general gymnasium students). By knowing basic IT concepts such as programming, algorithms or data structures, students will more easily understand the contents covered by the curriculum of this subject, which is why the curriculum is implemented in the 4th grade, after students have adopted the basic concepts of programming and ICT in general.

Since there is **no obligation to obtain approval for the implementation of a specific optional subject from the Ministry**, the high school independently adopts the curriculum of the optional subject and the associated literature.

[1] As mentioned earlier in the text, schools can decide about representations of electable subjects by themselves, depending on material and personnel capabilities.

The curriculum will include:

- Introduction
- Methodology of creating a curriculum
- Purpose and description of the curriculum
- Educational goals of implementation the optional subject
- Key curriculum domains
- Learning outcomes, key contents and levels of adoption
- Organization of learning and teaching
- Materials and contents for study
- Evaluation of adoption of outcomes
- List of necessary qualifications of teachers for the implementation of the curriculum
- Literature

The set of learning outcomes contains the following elements:

- Designation of learning outcome set
- Name of the set of learning outcomes
- List of learning outcomes
- Description of learning outcomes
- Key contents for the adoption of educational outcomes
- Recommendations for the adoption of educational outcomes

The main source for learning and teaching will be the teacher's textbook and the online platform created as PR4, and as additional literature, the following can be used (approved by the Ministry of Science and Education):

- Budin, L., Brođanac, P., Markučič, Z., Perić, S., Wendling, E., INFORMATIKA 3, textbook for the 3rd grade of high school, Element, 2020.
- Volarić, T., Toić Dlačić, K., Ivošević, I., Draganjac, M., Think IT, textbook for the 4th grade of high school, Alfa, 2021.

There are no additional requirements in the literature approval process.

In order to implement a new optional subject, the created curriculum must be presented to the School's Teaching Council, and approval for its implementation is given by the School Board.

The prerequisite for conducting the course is defined by the State Pedagogical Standard[2] (the number of students in the educational group must be at least 10).

[2] https://narodne-novine.nn.hr/clanci/sluzbeni/2008_06_63_2130.html
https://narodne-novine.nn.hr/clanci/sluzbeni/2010_07_90_2539.html

2.2. Accreditation ad literature approval in Czech Republic

In the year 2004, the Ministry of Education of the Czech Republic approved a new principle of education in the Czech Republic. For each type of school, a framework educational program (RVP) is approved. It is created by the National Pedagogical Institute, which falls directly under the Ministry of Education. The framework educational programs specify in particular: specific goals, forms, length and mandatory curriculum. Furthermore, it determines the educational areas and outcomes in each area. 4 RVPs in the Czech Republic for:

- Preschool education
- Elementary education
- Secondary education
- Vocational education

Based on the mandatory RVP of a specific area, each school creates its own school educational programs (SVP). They can thus expand the RVP according to their goals, e.g. expand the teaching of certain subjects, or focus more on different areas with respect to the school focus. Schools can also reflect the pedagogical intentions of the school organizer, which are the regional authorities. The RVP does not specify the year of study for each the curriculum. This makes the SVP more flexible in terms of intersubject relationships. The SVP is approved by the school director and the school is responsible for its compatibility with the relevant RVP. SVP is controlled by the Czech School Inspectorate.

2.2.1. RVP for secondary schools

Schools of our type are covered by the RVP for the second stage of elementary education and secondary education. The educational area of the OOP4FUN project is informatics within the RVP concept. Elementary education refers to the first four years of a more-year gymnasium. RVP for elementary schools and secondary schools include:

1. Data, information and modeling
2. Algorithmization and programming
3. Information systems
4. Digital technology.

The following outcomes are expected from elementary school students from algorithmization and programming.

- after reading the individual steps of the algorithm or program is student able to explain the entire procedure; determine the problem that is solved by the given algorithm
- student is able to divide the problem into individually solvable subproblems and proposes and describes the steps to solve them
- student is able to choose a suitable algorithm for the solving the problem from several options and justifies his choice; student is able to modify the given algorithm for other problems, student is able to propose different algorithms for solving the problem
- student is able to create a readable program in a block-oriented programming language with regard to its possible consequences and student is responsible for

them; student is able to test the program and corrects any errors in it; student is able to uses repetition, program branching, variables

- student is able to verifies the correctness of the procedure, student is able to find and correct any errors in it

The SVP for elementary schools should contain the following topics:

- algorithmization: decomposition of the task, problem; creation, writing and algorithm and its adaptation
- programming: programming environment tools, block-oriented programming language, cycles, branching, variables
- checking: verification of the algorithm, program (for example, by changing the inputs, checking the outputs, running it repeatedly); error finding (for example, by stepping); modification of algorithm and program
- creation of digital content: creation of programs (e.g. stories, games, simulations, robots); user needs, program user interface; program authorship and license; programmer ethics

The following outcomes are expected from secondary school students from algorithmization and programming.

- student is able to explain the given algorithm, program; student as able to determine whether the given procedure is an algorithm
- student as able to analyze the problem, divide the problem into smaller parts, decide which should be solved algorithmically, justify his decision; student is able to compiles and write algorithms to solve the problem
- student as able to evaluate the requirements of algorithms in relation to the nature and size of the input; student is able to compare algorithms according to different aspects, choose the most suitable one for the solved problem; student is able to improve the algorithm according to the selected viewpoint; student is able to make a general solution for a wider class of problems
- student is able to create a readable program for solving a specific problem with regard to its possible consequences and his responsibility for them; student is able to use repetition, program branching with compound conditions, variables, lists, subroutines with parameters and return values; in the pursuit of higher efficiency is student able to propose, manage and evaluate the concurrence of processes
- student as able to verifies correctness, find and correct possible errors in the algorithm, test, debug and optimize his program

The SVP for secondary schools should contain the following topics:

- algorithmization – assignment of the task, input, output, solution conditions; split the problem into parts, identification of data relations, repeating patterns and decision points; the term algorithm, properties of an algorithm, natural and formal languages, different notations of algorithms
- programming concepts – programming language; variables, data types and their properties, data input and output; subroutines with parameters and with return values; program branching with compound conditions, cycles, lists
- testing, optimization – syntactic, runtime and logical (functional) errors, program stepping and debugging; the effect of input data on consumed computing resources

- program development – tool selection according to task specification; design of a clear user interface of the program; program help and documentation; program authorship and license; programmer ethics

2.2.2. SVP for secondary schools

The educational field of informatics related to the OOP4FUN project is "Algorithmization and programming" for secondary schools. In elementary education, only block programming is expected. The choice of programming language and development environment is therefore entirely up to the school. The school selects the subjects for the teaching areas and compiles the SVP containing the teaching topics and the student's outcomes in accordance with the RVP. Furthermore, it is necessary to define intersubject topics with other subjects, so-called inter-subject relationships, within each topic of the curriculum. The SVP is usually presented in a table. As an example, let's give:

School outcomes	Curriculum	Intersubject topics
Student • is able to compile a program • understands the concept of a variable • distinguishes between different types of variables • is able to create a program for basic calculations	Student • Knows the structure of program and the notation of the variables	Math – simple computations Physics – simple computations

School outcomes must be in accordance with RVP. Furthermore, it is necessary to choose a suitable hourly dotation in accordance with RVP and to characterize the subject.

2.2.3. Study literature approval process

The process of approving literature is carried out by the Ministry of Education, Youth and Sports (MSMT). This grants or withdraws approval clauses from textbooks and teaching texts for primary and secondary schools. The granting of an approval clause is governed by: "Directive of the Deputy Minister for Education of the Ministry of Education, Youth and Sports on the procedure and conditions set for granting and withdrawing approval clauses for textbooks and teaching texts and for including textbooks and teaching texts in the list of textbooks", which is in the Czech language downloadable from: https://msmt.gov.cz/file/32170/download/.

Textbooks and teaching texts for health subjects in secondary schools are granted and withdrawn by the Ministry in agreement with the Ministry of Health. The list of textbooks and teaching texts to which the clause has been granted is published by the Ministry in the "Journal of the Ministry of Education, Youth and Sports".

The granting of an approval clause can be resolved at any time at the Ministry of Education, Youth and Sports. The request can be made by post or personally to the address of the ministry. Information describing the procedure for obtaining an approval clause for textbooks is published in the Czech language on the website:

https://portal.gov.cz/sluzby-vs/udelovani-schvalovacich-dolozek-ucebnicim-S979:

The condition for granting the approval clause is two positive recommending review opinions out of three.

2.3. Accreditation and literature approval in Germany – Saxony

In Germany the process of adapting new curriculums and subjects depends on the state and type of school. This section describes the process and conditions for grammar schools/high schools (German "Gymnasium") in saxony, as the project partner is a "Gymnasium" in saxony.

2.3.1. General process of defining curricula and timetables

According to § 35 of the "Sächsisches Schulgesetz" (the law regulating schools in saxony) all teaching and subjects are based one the national educational standards and the state-specific timetables and subject-specific curricula. Both curricula and timetables are determined by the highest school supervising authority, the "Staatsministerium für Kultus" (state ministry of culture). The curricula are developed by commissions, which are part of the "Landesamt für Schule und Bildung" (State Office for Schools and Education). Members of the curriculum commission include subject teachers. It is therefore very complicated and politically difficult to create new subjects or modify existing ones. However, the subject computer science already allows elements of OOP4Fun to be integrated, as discussed in the next section.

2.3.2. Integration of OOP4Fun in the existing subject computer science

Regarding the subject "Informatik" (computer science) there are currently two curricula. The current curriculum for grades 10, 11 and 12 does not include any mandatory content on object oriented programming. However in grades 11/12 there is a elective learning area (chosen by the teacher) called "8C - Practical Computer Science - Advanced Programming" which includes the basics of object-oriented programming (inheritance, polymorphism, encapsulation) and their application to the solution of a complex problem. The learning area comprises 14 lessons (45 minutes each). Parts of the OOP4Fun curriculum could be implemented here, as it is compatible with the objectives of this study area of the curriculum. However, the curriculum is only valid for 3 more years.

In the new curriculum, object orientation also plays a role in the compulsory part of the curriculum. The learning goals of learning area 2 - "algorithms and programming" (grade 11/12) include concepts of object-oriented programming (class and object, attributes, methods) and applying them to simple problems. The learning area comprises 20 lessons (45 minutes each). Since the learning area also covers many other aspects, only a small part of this is available for object orientation, depending on the teacher. However, the learning area "8B: software development" in the elective part of the curriculum allows to extend the topic with another 8 lessons, if the following topics are covered:

- basics of software architecture
- knowing approaches to software development (classical approaches, agile)
- transferring knowledge of a software development method to the implementation of a problem solution

By using this learning area and the learning area 7 “computer science project” (10 lessons), it is possible to implement parts of the OOP4Fun curriculum, if it is modified to include the additional objectives.

2.3.3. Establishing of new (elective) subjects

As already discussed, the timetable and curriculum are determined by the school supervisory authority and therefore it is very difficult to create new subjects or modify existing ones. However, under certain conditions individual schools can implement new subjects.

There are two possibilities: In accordance with the timetable ("VwV Stundentafeln, Anhang 4a") every school offers a subject called "Individuelle Förderung" ("individual support") comprising of 5 lessons per week, distributed over years 5 to 10. If the school uses a part of this lessons for computer science related classes, a curriculum such as OOP4Fun can also be implemented, as there are no central curriculum requirements for the subject. The prerequisite here is that enough teachers are willing to implement the curriculum and the conference of all teachers and school management approve it.

In grades 11/12, schools have the option of offering interdisciplinary courses (SOGYA § 43(1)). These must be approved by the school supervisory authority, in this case the "Landesamt für Schule und Bildung" (State Office for Schools and Education). The school must submit an application for this. This therefore requires additional commitment on the part of the teaching staff as well as sufficient teaching staff to teach the subject. However, the OOP4Fun curriculum is not interdisciplinary, so it does not fulfill the requirements. It can therefore not be implemented here without changes.

2.3.4. Accreditation process for teaching materials

Currently there is no accreditation process for teaching materials. Teachers can use the teaching materials they see fit to reach the learning goals defined in the curriculum.

However, since teaching materials must be provided by the school, in the case of textbooks the actual usability depends on whether the school purchases the textbook. The decision which textbooks are purchased is made by the school, in this case the "Fachkonferenz" (conference of the subject teachers of the school) and school management.

Additionally, the highest school supervising authority can make the use of teaching and learning materials subject to approval (see Sächsisches Schulgesetz § 60). To the author's knowledge, this is not currently the case.

2.4. Accreditation and literature approval in Serbia

The curriculum is created in accordance with:

1. Regulations on the curriculum for high school ("Official Gazette of SRS - Education Gazette", No. 5/90 and "Educational Gazette", No. 3/91, 3/92, 17/93, 2/94, 2/ 95, 8/95, 23/97, 2/02, 5/03, 10/03, 11/04, 18/04, 24/04, 3/05, 11/05, 2/06, 6/06, 12/06, 17/06, 1/08, 8/08, 1/09, 3/09, 10/09, 5/10, 7/11, 4/13, 14/13, 17/13, 18/ 13, 5/14, 4/15, 18/15, 11/16 and 13/16), in the part that refers to the natural-mathematical and social-linguistic course of Gymnasium 2. By the Rulebook on the curriculum of the subject of religious education ("Educational Bulletin", no. 6/03, 23/04, 9/05 and 11/16); 3. Rulebook on curriculum for gifted students in Mathematical Gymnasium ("Official Gazette of the Republic of Serbia - Education Gazette", no. 12/16 and 13/16).

Gymnasium is an independent educational institution, one of two secondary schools in Ivanjica.

The high school in Ivanjica has three majors, and in the 2024/25 school year. she enrolled in 5 classes:

- General course, 3 classes (84 students)
- Students with special abilities for computing and informatics, 1 class (20 students)
- Students with special abilities for sports, 1 class (20 students)

As of September 1, 2024, the maximum number of students in high school classes in the Republic of Serbia is 28.

2.4.1. About the direction

The high school IT department is a specialized department for students with strong digital competences.

Specialization in one scientific field enables a high level of preparedness for continuing higher education - studying in that field, but does not deny students access to study programs in completely different fields.

The number of students, as in other specialized high school classes, is 20. Most of the teaching of so-called computer subjects takes place in groups of 10 students each. Dividing classes into groups enables better quality of work, and therefore better student achievements.

2.4.2. Advantages

Students will master all basic, but also some more advanced concepts of programming, they will get to know different approaches to solving problems through programming. Understand computer networks, network layers and protocols and know what it takes to keep computer networks secure.

In addition to a larger list of specific computer competencies, it is particularly important to emphasize that students will be able to effectively use programming and work with databases to solve various problems in further education, professional work and everyday life. This represents the functionalization of knowledge.

By attending this educational program, students will improve strategies and techniques of independent learning using computer capabilities and will build readiness to follow new solutions in the field of information technology.

2.4.3. IT subjects

Specialized subjects are studied in different classes. In four years, students will have a total of 934 hours of IT subjects. This number of classes, the content studied, the division of classes into smaller groups, professional teachers and equipped schools represent conditions for high school graduates that will be extremely effective during further studies in IT majors of various faculties.

The first generation, 20 students (the maximum number of students in the class is 20) was enrolled on September 1, 2018 (school year 2018/19). The professional subjects that the students will have are:

IT subjects		I class	II class	III class	IV class	Number of students in the group
		Number of hours of exercises				
1.	Application of computers	74	74	74		8 – 12
2.	Programming	111	111	111		8 – 12
3.	Computer systems	74				8 – 12
4.	Operating systems and computer networks		74			8 – 12
5.	Object-oriented programming			111		8 – 12
6.	Databases			37	66	8 – 12
7.	Program paradigms				99	8 – 12
8.	Web programming				66	8 – 12

2.4.4. The process of obtaining permission to use a new textbook

The preparation, approval, selection, issuance, withdrawal and monitoring of textbooks and textbook sets, manuals and compulsory and additional teaching aids for primary and secondary schools are regulated by the Law on Textbooks ("Official Gazette of RS", No. 27 of April 6, 2018, 92 from October 27, 2023)

https://pravno-informacioni-sistem.rs/eli/rep/sgrs/skupstina/zakon/2018/27/2/reg

The publisher submits the request for approval of the textbook manuscript to the Ministry. The report contains three expert evaluations of the quality of the textbook manuscript given by the publisher's review committee. The Ministry submits four copies of the textbook manuscript and the report in printed and electronic form to the Institute within ten days from the date of receipt of the manuscript from the publisher. Expert assessment of the quality of textbook manuscripts is given by a committee formed by the Institute from among its employees. The Institute submits an expert evaluation to the Ministry within 90 days from the date of receipt of the manuscript by the Ministry.

Before passing the expert assessment, if the manuscript contains defects, the institute will return the textbook manuscript to the publisher for revision with an explanation of all observed deficiencies, in order to eliminate them. based on the reasoned professional evaluation, makes a proposal to accept or reject the request for approval of the manuscript. The institute submits a copy of the corrected textbook manuscript to the Ministry. Based on the expert evaluation of the committee, the minister makes a decision on accepting or rejecting the textbook manuscript within eight days.

2.4.5. Opinion on the results of the project

In our school, there are 144 classes of OOP per year, and a month and a half is planned for the preparation of a project assignment, which the students design and present.

We think to insert the results of the project into the existing program instead of the current project assignment.

That is, within the subject OOP as a separate area, insert the results of the project. We have enough space to add them to the current schedule. We do not need any permission for that, because according to the law we can change the plan and program by ourselves by about 20 percent.

2.5. Accreditation ad literature approval in Slovakia

In Slovakia, the State Educational Program (SEP) is used as the basis for teaching. The SEP defines the compulsory content of education - the core curriculum - in the field of Vocational Education and Training, which is guaranteed by the state. They are issued and published by the Ministry of Education, Research, Development and Youth of the Slovak Republic (MoERDaY SR) after consultation with employers, school founders and their professional and specialist associations with a nationwide scope of activity and with line ministries within the scope of their sectoral competence in accordance with generally obligatory legislation. SEP are created for individual groups of fields and their name corresponds to the name of the given group of fields. SEP are a binding curriculum document for the development of school educational programmes (SchEP), textbooks, teaching texts and workbooks, for the evaluation and control of learning outcomes (source:https://siov.sk/en/vzdelavanie/odborne-vzdelavanie-a-priprava/).

Each school develops its own SchEP on the basis of a specific subject field's SEP. They can thus expand the SEP according to their goals, e.g. to broaden the teaching of certain subjects or to focus more on different areas of the field. It reflects the focus of the school and the strategy determined by the school management together with the region and the town. It takes into account the needs and possibilities of students, teaching staff. The school uses optional hours for the development of the SchEP and develops its own school curriculum (SC). All the hours available to the school are used in the SC. The school may also have more than one SC if it has classes with different specialization. New subjects chosen by the school are also listed in the SC. The SC will also indicate how the cross-cutting themes will be implemented. For each subject, if the school has added hours for the subjects in SEP, the school must develop a curriculum. In the case of the incorporation of new subjects, it will prepare the curriculum. A school can create a name for its SchEP that reflects its mission. The curriculum for the school Business Academy Považská Bystrica is available at https://oapb.edupage.org/a/skvp.

SchEP is issued by the school principal after discussion in the pedagogical council of the school, the school council and written consent of the school foundation. Compliance of the SchEP with the SEP, the aims and principles of education is checked by the State School Inspection (source: https://www.statpedu.sk/sk/svp/statny-vzdelavaci-program/statny-vzdelavaci-program-gymnazia/zasady-tvorby-skvp/, https://www.minedu.sk/8388-sk/vzorove-skolske-vzdelavacie-a-vychovne-programy/).

2.5.1. SEP for gymnasiums - subject Informatics

In Slovakia, the educational area of the OOP4FUN project at grammar schools will focus on the area of Mathematics and work with information and specifically on the subject of Computer science. It also makes a difference whether it is a gymnasium with a four- and five-year education program or an eight-year education program.

For gymnasium with a four- and five-year educational program (source: https://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_g_4_5_r.pdf), the following parts are of interest within the educational standard of the subject:

Communication and collaboration - working with collaboration and information sharing tools

Performance standard	Content standard
The student knows/has the ability to • use tools to share and publish information, • create and edit shared products using collaboration tools, • use collaborative problem-solving tools**.**	*Processes*: sharing folders and files on the web, parallel work with text, images, spreadsheets and presentations

Algorithmic problem solving - problem analysis

Performance standard	Content standard
The student knows/has the ability to • identify the input information from the task assignment, • describe expected outputs, outcomes, actions, • identify the problem to be solved algorithmically, • formulate and express informally (in natural language) the idea of the solution, • consider the characteristics of the implementer (e.g., turtle, graphic pen, robot, etc.), • plan the solution to a problem as a sequence of branching statements and repetition	*Properties and relationships*: problem - input - output *Processes*: breaking the problem into smaller parts, synthesizing a solution from solutions of smaller parts, identifying recurring patterns, identifying decision points (branching and repetition), identifying general relationships between information

Algorithmic problem solving - a language for writing solutions

Performance standard	Content standard
The student knows/has the ability to - use the language to write an algorithmic solution to a problem (use language constructs, apply language rules), - use mathematical expressions to express relationships and conditions, - recognize and correct errors in notation, - create notations and interpret notations according to new established rules (syntax) for writing algorithms.	*Terms:* program, programming language *Properties and relationships*: algorithm writing and program execution, input - program execution - output/action *Processes*: program compilation, identification, search, and correction of errors

Algorithmic problem solving - using a sequence of commands

Performance standard	Content standard
The student knows/has the ability to - solve the problem by composing the commands into a sequence, - apply the rules and language constructs to compose a sequence of commands.	*Terms*: command, command parameter, command sequence *Properties and relationships*: how the commands and the outcome of the program implementation are related Processes: assembling and modifying commands, evaluating the sequence of commands, modifying the sequence of commands (adding, removing commands, changing the order of commands)

Algorithmic problem solving - using tools for interaction

Performance standard	Content standard
The student knows/has the ability to - recognise situations where input needs to be obtained, - identify the characteristics of the input information (constraints, scope, format), - recognize situations, where output needs to be displayed, perform an action, - write an algorithm that responds to the input, - form a hypothesis about how an unknown algorithm handles a given input, given input-output/action pairs.	*Properties and relationships*: language means for obtaining input, processing input and displaying output *Processes*: waiting for unknown input - execution of action - output, after-effect

Algorithmic problem solving - using variables

Performance standard	Content standard
The student knows/has the ability to - identify from the task assignment which data must be remembered or are changing (and thus require the use of variables), - solve problems in which the remembered values must be remembered and later used in expressions,	*Terms*: variable, variable name, variable value, operation (+, -, *, /) *Properties and relationships*: language rules for variable usage, variable name - variable value

- generalise the solution so that it works not only with constants.	*Processes*: setting a value (assignment), getting a value (using a variable), changing the value of a variable, evaluating an expression with variables, numbers and operations

Algorithmic problem solving - using cycles

Performance standard	Content standard
The student knows/has the ability to - recognise repeating patterns, - recognize what part of the algorithm to execute before, during and after the cycle, - solve problems where the result is to be obtained by accumulating partial results within a loop, - solve problems that require an unknown number of iterations, - solve problems in which cycles and branching are combined, - determine the bounds and conditions for the execution of cycles.	*Terms*: repetition, number of repetitions, condition of cycle execution, cycle body *Properties and relationships*: how the number of repetitions relates to the result, which applies after the cycle is over *Processes*: evaluation of cycle boundaries/conditions, cycle execution

Algorithmic problem solving - using branching

Performance standard	Content standard
The student knows/has the ability to - recognise situations and conditions where branching should be used, - recognise what part of the algorithm to execute before, during and after branching, - solve problems that require branching with compound conditions (with logical conjunctions), - solve problems that combine cycles and branching.	*Terms*: branching, condition *Properties and relationships*: true/false - fulfilled/unfulfilled condition *Processes*: constructing and modifying branching, creating a condition and evaluating a condition with negations and logical conjunctions (and, or)

Algorithmic problem solving - interpretation of the solution notation

Performance standard	Content standard
The student knows/has the ability to - to step through the solution, simulate the action of the executor with a sequence of statements, with expressions and variables, with branching and with loops, - express the idea of a given instruction (discover and describe in their own words the idea of the written solution - how the program works, what the notation implements for different inputs), - modify the solution to the problem given different constraints, - add to, complete, modify the worked solution,	*Properties and relationships*: language - program execution *Processes*: troubleshooting what happens in the computer in case of an error in the program

- look for the relationship between the input, the algorithm and the result, - reason about different solutions, suggest improvements.	

Algorithmic problem solving - finding and fixing errors (bugs)

Performance standard	Content standard
The student knows/has the ability to - detect when a program is not working properly, - find the error in your own program that is not working properly and fix it, - to find out for which inputs, in which cases, in which situations the program works wrong, - give counterexamples when something does not fit or does not work, - to evaluate and verify the correctness of a solution (your own and someone else's), - distinguish an error in implementation from an error in notation.	*Properties and relationships*: error in the sequence of statements (wrong statement, missing statement, swapped statement or extra statement), error in expressions with variables, error in algorithms with loops and branching, error in implementation (logical error), error in notation (syntax error) *Processes*: error recognition, error search

In a gymnasium with an eight-year curriculum, even more emphasis is placed on the teaching of the thematic parts of the curriculum, but the basic structure of the relevant parts of the curriculum is similar. More detailed information can be found at https://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_g_8_r.pdf.

2.5.2. SEP for other secondary schools - subject Informatics

For other types of schools, their ability to use the results from the OOP4FUN project depends on their SEP and field of study. Here it is necessary to note what has already been mentioned above and that is the school defines its SchEP on the basis of SEP. This opens up possibilities for implementing the necessary parts to apply the results from the OOP4FUN project. As for the subject Computer Science, the basic information can be found here https://siov.sk/statne-vzdelavacie-programy/. Out of the three offered options, the educational standard for fields of study in which the student receives a full secondary vocational education is of interest.

Procedures, problem solving, algorithmic thinking

Performance standard	Content standard
Students after graduation from a given field of study knows/has the ability to - analyse the problem, propose an algorithm for solving the problem, write the algorithm in an understandable formal form, verify the correctness of the algorithm, - solve problems using algorithms, write them in a programming language, find and correct errors, - understand finished programs, determine the properties of inputs, outputs and relationships between them, test and modify them,	- problem, algorithm, algorithms from everyday life, ways of writing algorithms, - stages of problem solving, problem analysis, algorithm, program, debugging, - programming language, syntax, program execution, logic errors, runtime errors, - concepts, commands (assignment, input, output), control structures (conditional statements, loops), variables, types, set operations.

• solve problems using commands with various constraints on the use of commands, variables, types and operations, • use the basic types of programming language used, • recognize and correct syntax errors, correct errors occurring during program execution.	

However, it is also possible to define a subject (e.g. Applied Computer Science - Seminar) that is not general education but vocational, which may be more advantageous for some schools. This subject would correspond to similar educational standards as Computer Science.

2.5.3. Approval process of study literature

The literature approval process is carried out by MoERDaY SR. It grants or withdraws approval endorsements from textbooks and study literature for primary and secondary schools. The granting of the approval endorsement is governed by the applicable directives, which can be found e.g. here https://siov.sk/vzdelavanie/ucebnice-a-ucebne-pomocky/vyber-a-schvalovanie-ucebnic/ and the current directive can be found here https://www.minedu.sk/36670-sk/smernica-c-462023-o-edukacnych-publikaciach-ucinnost-od-20122023/. These directives provide a breakdown of educational publications as well as the approval process for a given educational publication. The granting of an approval endorsement can be addressed at any time at MoERDaY SR. The request can be submitted by mail or in person to the address of the Ministry. Based on the record of the assessment of the educational publication, the Ministry will issue the endorsement or will not issue the endorsement. The relevant section shall notify the submitter in writing and at the same time send the direct assessment and the expert opinion if it is a special educational publication. The assessment of the educational publication shall be carried out by a reviewer who shall prepare the review, by a member of the staff of the directly managed competent organization who shall prepare the direct assessment and by a member of the staff of the directly managed competent organization who shall prepare the expert opinion if it is a special educational publication. The criteria for the evaluation of the literature can be found at https://siov.sk/vzdelavanie/ucebnice-a-ucebne-pomocky/kriteria-na-hodnotenie-ucebnic/.

3. New syllabuses

Each high school prepared new subject syllabuses with results of OOP4FUN project. The syllabuses contain classes with Greenfoot environment and “light OOP” topics. In the sections below there are subject syllabuses and their characteristics for each country.

Subject syllabuses are in the form of tables with topics names, teaching hour dotation, used software and developing environment. The tables are supplemented OOP topics and by graphs of hour dotation of OOP topics defined in PR1 with highlighting “Light OOP” topics. List of OOP topic from PR1 are bellow. Light OOP topics are with bold font.

1. **classes, objects, instance**
2. **methods, passing methods arguments**
3. **constructors**

4. **attributes**
5. method and constructor overloading
6. **static variables and methods**
7. packages
8. **encapsulation**
9. class diagram
10. association
11. **inheritance**
12. composition
13. send object message
14. immutable object
15. aggregation
16. **abstract classes**
17. polymorphism
18. **interface**
19. **exception**
20. **object live cycle**
21. virtual methods
22. UML
23. Generic classes
24. Nested classes

As in PR2 there was introduced list of learning and teaching activities, each subject description contains list of learning and teaching activities (LTA) related to each topic and graph with number of hours of using each learning and teaching activity. List of learning and teaching activities is below:

1. Peer learning
2. Team teaching
3. Inquiry learning
4. Flipped classroom
5. Problem-based learning
6. Interdisciplinary learning
7. Blended learning
8. Gamification
9. Visual learning

To compare also teaching materials support for students there were defined its following types

1. Pure explanation
2. Text support
3. Prepared code support
4. Video support

While teaching OOP supporting materials are needed. In some situation, students are able to understand topics immediately after they are explained by teacher. Some parts need text supporting materials or prepared source codes. Most difficult part can be supported by a video with detailed explanation. Each subject description contains list of support materials related to each topic and graph with number of hours, where specific support is needed.

Due to different time dotation in each country a percentage graph is added.

There are two types of subjects - optional and mandatory. Mandatory subjects should prepare and motivate students for choosing optional subject with programming. If there is a situation that there is no mandatory subject with programming in the institution there is a description of student motivation process to select subject with programming. In following sessions there will be English version of documents.

3.1. New syllabus in Croatia

In school in Ivanec there was created one optional subject called “OOP through game development” for students of 4th year of study. The aim of the subject is to show OOP principal bases to the students because the OOP principals were not taught before.

3.1.1. Subject “OOP through game development”

Table 1- OOP through game development subject

Topic		Name	Hours	SW
1.		Subject introduction, software requirements, methods and ways of working, students' obligations	1	
2.		Repetition of basic concepts of programming (instructions, functions, branches, loops)	2	
3.	Catch a fly game	Motivation game: Catch a fly	2	Greenfoot
4.		Git	2	GitHub, SourceTree
5.	Tower defense game	Greenfoot and the Java language - introduction	3	Greenfoot
6.		Algorithm, application control, method creation	3	Greenfoot
7.		Branching and enemy control	3	Greenfoot
8.		Variables and expressions	3	Greenfoot
9.		Association	4	Greenfoot
10.		Inheritance	4	Greenfoot
11.		Encapsulation	3	Greenfoot
12.	Students' games	Students' independent work presentations (creating your own game - year-long project)	2	Greenfoot

TABLE 2 - OOP THROUGH GAME DEVELOPMENT SUBJECT - LIST OF OOP TOPICS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
3	Y	Y	Y	Y	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
4	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
5	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
7	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N
8	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N
9	Y	Y	Y	Y	Y	Y	N	N	N	Y	N	N	N	N	N	Y	N	N	N	Y	N	N	N	N
10	Y	Y	Y	Y	N	N	N	N	N	N	Y	N	N	N	N	Y	N	N	N	Y	N	N	N	N
11	Y	Y	Y	Y	N	Y	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
12	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	N	N	N	N	Y	N	N	N	Y	N	N	N	N

TABLE 3 - OOP THROUGH GAME DEVELOPMENT SUBJECT- LIST OF LTA + SUPPORT

	TLA									Support			
	1	2	3	4	5	6	7	8	9	1	2	3	4
1	N	N	N	N	N	N	N	N	N	Y	N	N	N
2	Y	N	N	N	N	N	N	N	N	Y	N	Y	N
3	N	N	Y	N	Y	N	N	Y	N	Y	N	Y	N
4	Y	N	Y	N	N	N	N	Y	N	Y	N	N	N
5	N	N	Y	N	N	N	N	Y	N	Y	N	N	N
6	Y	N	Y	N	Y	N	N	Y	N	Y	Y	N	N
7	Y	N	Y	N	Y	N	N	Y	N	Y	Y	N	N
9	Y	N	Y	N	Y	N	N	Y	N	Y	Y	N	N
9	Y	N	Y	N	Y	N	N	Y	N	Y	Y	N	N
10	Y	Y	Y	N	Y	N	N	Y	N	Y	Y	N	N
11	Y	Y	Y	N	Y	N	N	Y	N	Y	Y	N	N
12	Y	Y	Y	Y	Y	N	N	Y	N	N	N	N	N

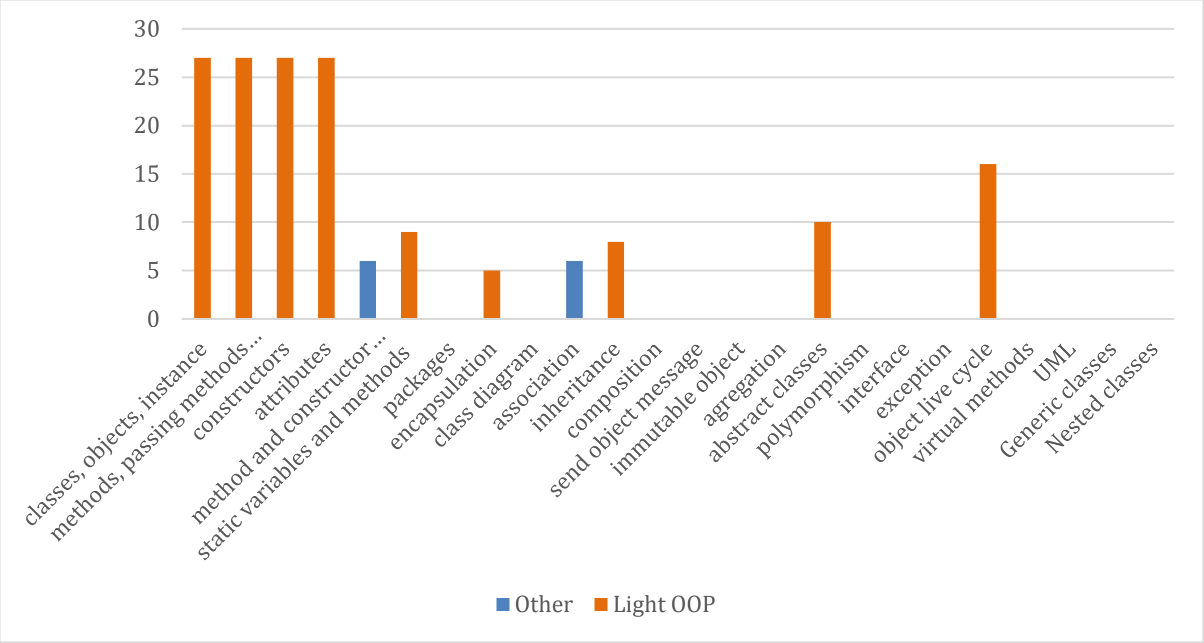

FIGURE 1 - OOP THROUGH GAME DEVELOPMENT SUBJECT - OOP TOPICS HOURS

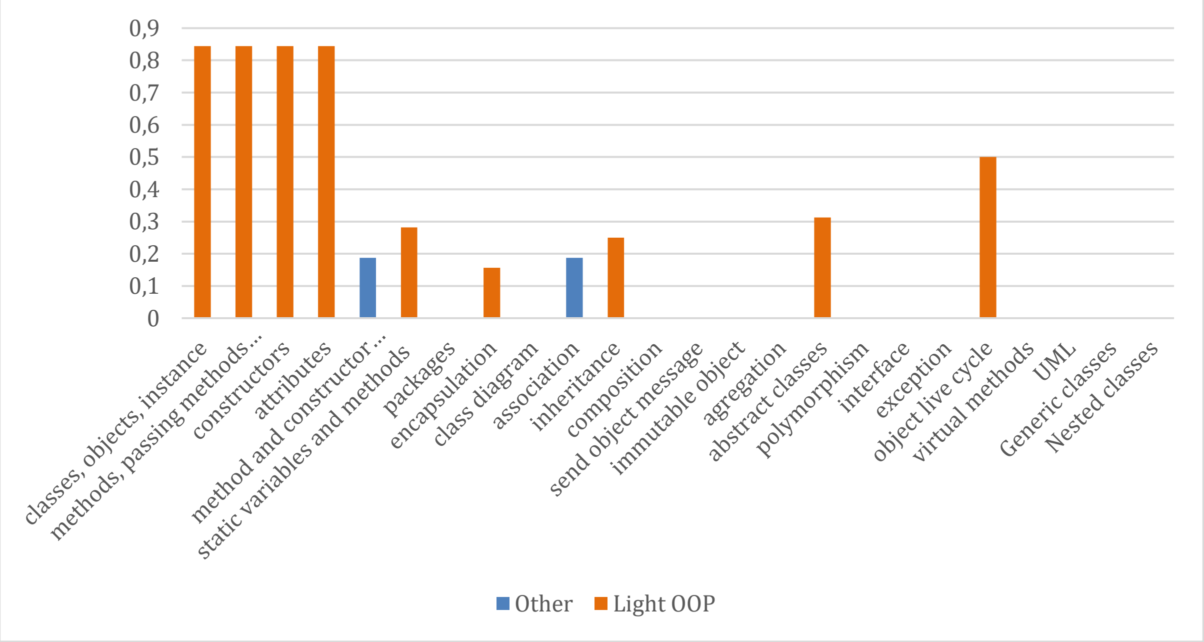

FIGURE 2 - OOP THROUGH GAME DEVELOPMENT SUBJECT- PERCENTAGE OF OOF TOPICS HOURS

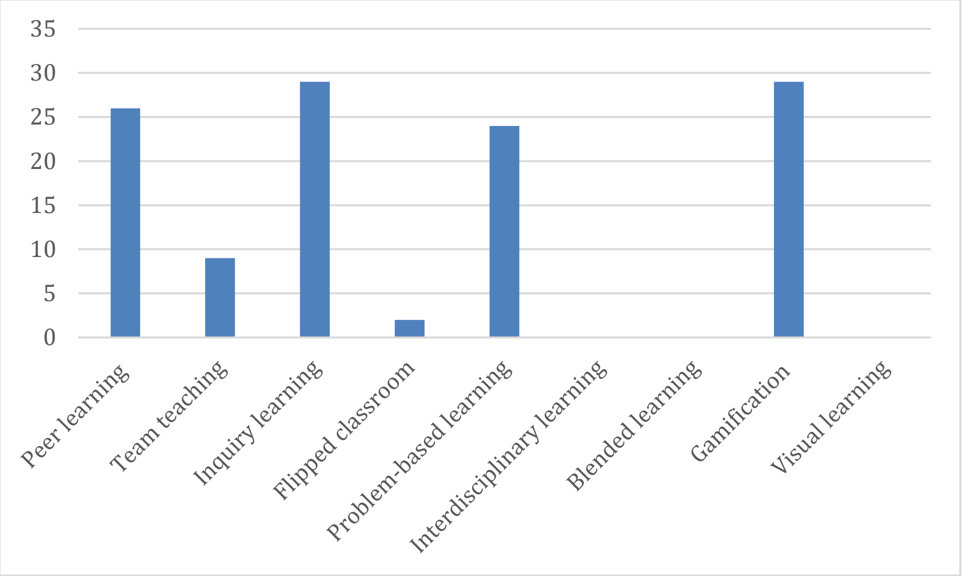

FIGURE 3 - OOP THROUGH GAME DEVELOPMENT SUBJECT - LTA

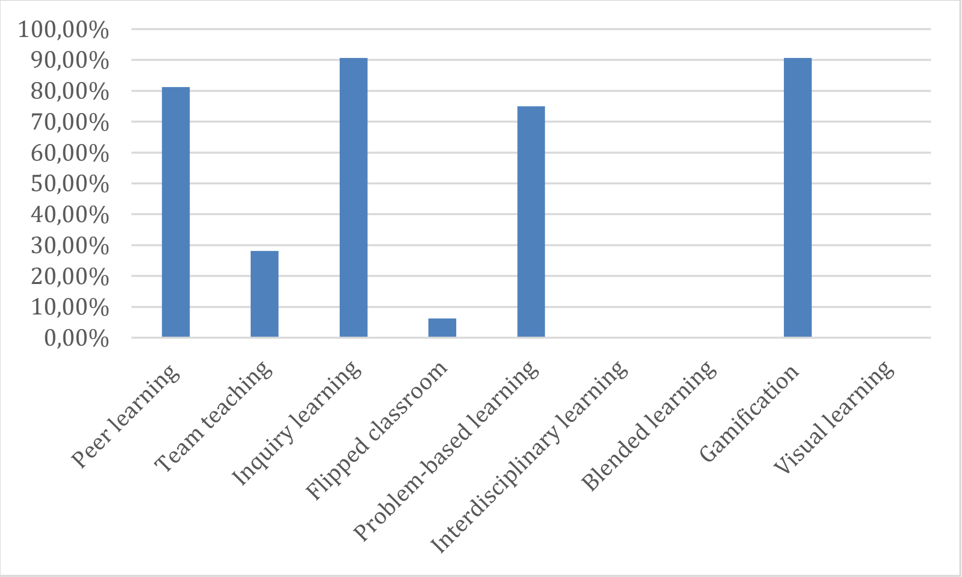

FIGURE 4 - OOP THROUGH GAME DEVELOPMENT SUBJECT - LTA PERCENTAGE

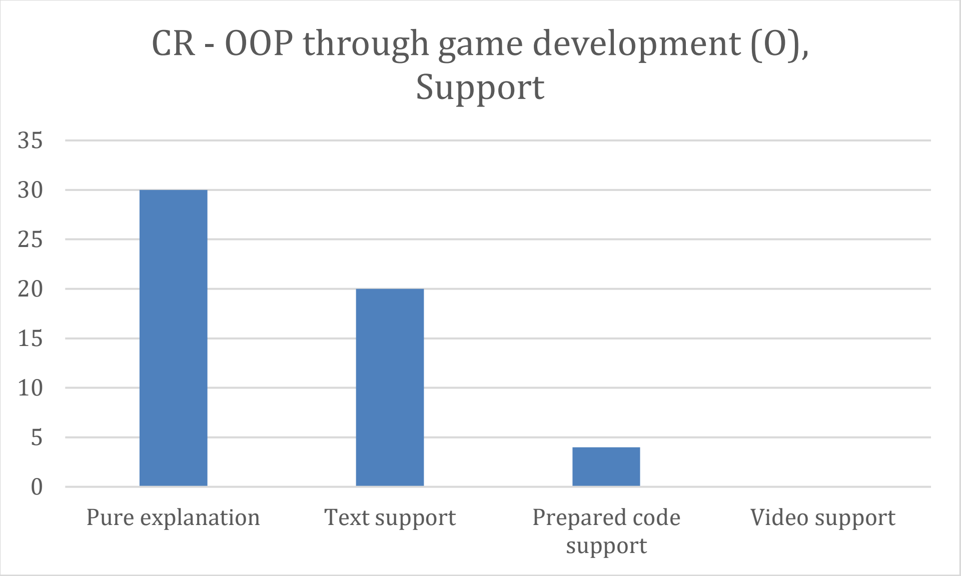

FIGURE 5 - OOP THROUGH GAME DEVELOPMENT SUBJECT- SUPPORT

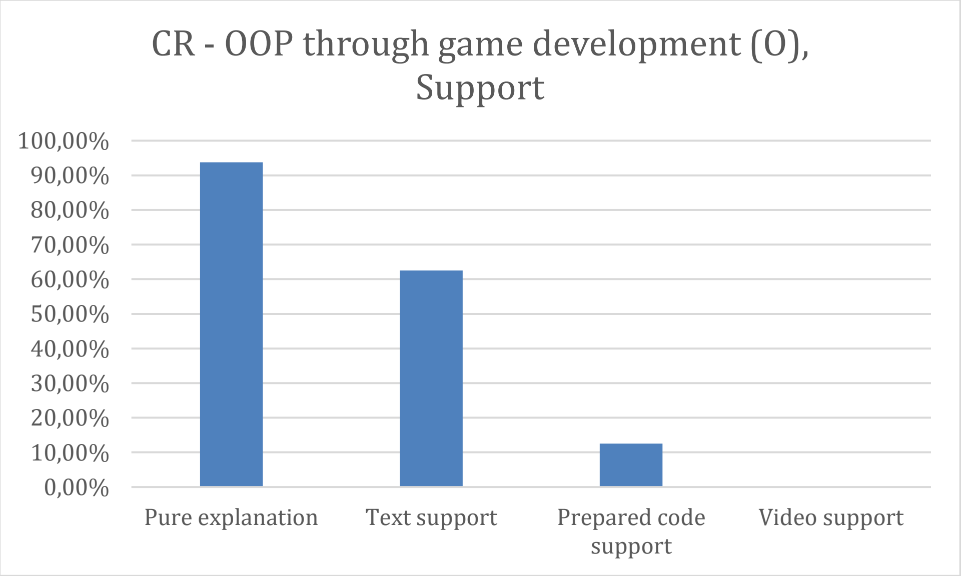

FIGURE 6 - OOP THROUGH GAME DEVELOPMENT SUBJECT– SUPPORT PERCENTAGE

3.1.2. Motivation of students

At the end of their 3rd grade, we will motivate the students to choose the mentioned optional subject through a demonstration of the games they will develop and why it will be useful for them to choose that subject. The motivation text should look like this:

Dear students,

you have a unique opportunity to enroll in an optional subject that can open doors to the world of programming and game development! Name of this subject is **Object-oriented programming through game development** and as it's name says, the focus is on gaining the basics of object-oriented programming, but in a interactive and fun way, through game development. Here are several reasons why you should choose this subject and what benefits does it offer:

1. **Valuable skills for the future**:
   - Programming is a highly sought-after skill in today's job market. By learning the basics of object-oriented programming, you will gain a solid foundation for further development in the IT field. These skills will significantly ease your admission to related university programs, such as computer science, electrical engineering, and related disciplines. Additionally, the knowledge you acquire can also help you in various other fields that utilize technology, such as science, engineering, and business.
2. **Game development as a fun learning method:**
   - Programming can be quite challenging, especially object-oriented programming, which you might not have encountered before. Learning through game development will greatly simplify and help you grasp fundamental concepts in a fun way. By creating your own games, you will learn key programming concepts such as classes, objects, inheritance, and association, all while having fun. This approach makes learning less stressful and much more interesting.
3. **Teamwork**:
   - The classes will be conducted in teams, meaning you will have the opportunity to work together with your peers, exchange ideas, and develop teamwork skills. Working in teams simulates real working environments in the IT industry, where collaboration is key to success. You will learn how to communicate effectively, solve problems together, and divide tasks in a way that best utilizes each team member's strengths.
4. **Minimal time commitment**:
   - The subject will be conducted just one hour a week, so it won't take up too much of your time but will provide you with significant knowledge and skills. This means you will be able to easily balance this subject with your other school obligations. One hour a week is enough to gain basic skills and knowledge about object oriented programming, yet short enough not to be a heavy burden on you.
5. **Equality and inclusivity**:
   - The subject is designed to be equally beneficial for both female and male students. We believe that everyone has the potential to become a successful programmer, regardless of gender. Programming and IT are fields where diversity is extremely important, and we want all students to have an equal opportunity to develop and progress.

By choosing this subject, you will step into the world of programming and technology, fields that are shaping the future. Take advantage of this opportunity to have fun, learn something new, and prepare for future challenges. Sign up and become part of this exciting adventure!

In addition, a demonstration workshop can be organized for the students where they will be presented with which games will be developed, perhaps for the students to implement some simpler functionality themselves in order to further motivate them to choose the subject.

3.2. New syllabus in Czech Republic

In Gymnazium, Pardubice, Dašická 1083 there were innovated two subjects – one mandatory and one optional.

The mandatory subject is named "Informatics". It is subject for students of the 2nd year of study. The programming skills are taught only in the second half of the subject. So the syllabus contains only topics related to programming. The aim of this subject is to motivate students that want to study on technical universities to choose optional subject related to programming skills. The main goal of this subject is to show the difference between block-based programming and code-based programming language and simplify transition between them.

After the 2nd year of study students can choose from an offer of two 2-years optional subject, which are taught in the third and fourth year of study, and two 1-year optional subjects which are taught in the fourth year of study.

Subject related to programming skills is "Seminar of programming". It was completely innovated by using of the project results. New principal - object first was used here and demonstrated by Greenfoot environment. Since the subject "Seminar of programming" is a 2-year subject, the syllabus table will be divided by year.

3.2.1. Subject "Informatics"

Table 4 - Informatics subject

Topic		Name	Hours	SW
1.		Classroom rules (mandatory in CZ for computer room)	1	
2.		Algorithm - definition, complexity, examples of algorithms	1	
3.	Algorithmization and programming	Programming language	2	
4.		Block oriented language vs. Java code - play and modify prepared game	4	Greenfoot
5.		Development tool Greenfoot	4	Greenfoot
6.		Methods, attributes, constructor, stepping the program	4	Greenfoot
7.		Program branching, variables, Object Life cycle	6	Greenfoot
8.		Loops	4	Greenfoot
6.		Arrays and lists	4	Greenfoot
10.		Game creation	8	Greenfoot

TABLE 5 - INFORMATICS SUBJECT - LIST OF OOP TOPICS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
3	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
4	**Y**	**Y**	N	**Y**	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
5	**Y**	**Y**	N	**Y**	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
7	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	**Y**	N	N	N	N	N	N	N	N	N	**Y**	N	N	N	N
8	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	**Y**	N	N	N	N	N	N	N	N	N	**Y**	N	N	N	N
9	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	**Y**	N	N	N	N	N	N	N	N	N	**Y**	N	N	N	N
10	**Y**	**Y**	**Y**	**Y**	**Y**	N	N	N	N	**Y**	N	N	N	N	N	N	N	N	N	**Y**	N	N	N	N

TABLE 6 - INFORMATICS SUBJECT - LIST OF LTA + SUPPORT

	TLA									**Support**			
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**1**	**2**	**3**	**4**
1	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	N	N	N	N
3	N	N	N	N	N	N	N	N	N	**Y**	N	N	N
4	N	N	N	N	N	N	N	**Y**	**Y**	N	N	**Y**	N
5	**Y**	N	**Y**	N	N	N	N	N	**Y**	**Y**	N	N	N
6	**Y**	N	**Y**	N	N	N	N	N	**Y**	**Y**	N	N	N
7	**Y**	N	**Y**	N	N	N	N	N	**Y**	**Y**	N	N	N
8	**Y**	N	**Y**	N	**Y**	N	N	N	**Y**	**Y**	N	N	N
9	**Y**	N	**Y**	N	**Y**	N	N	N	**Y**	**Y**	N	N	N
10	**Y**	N	N	**Y**	**Y**	N	N	**Y**	**Y**	**Y**	N	N	N

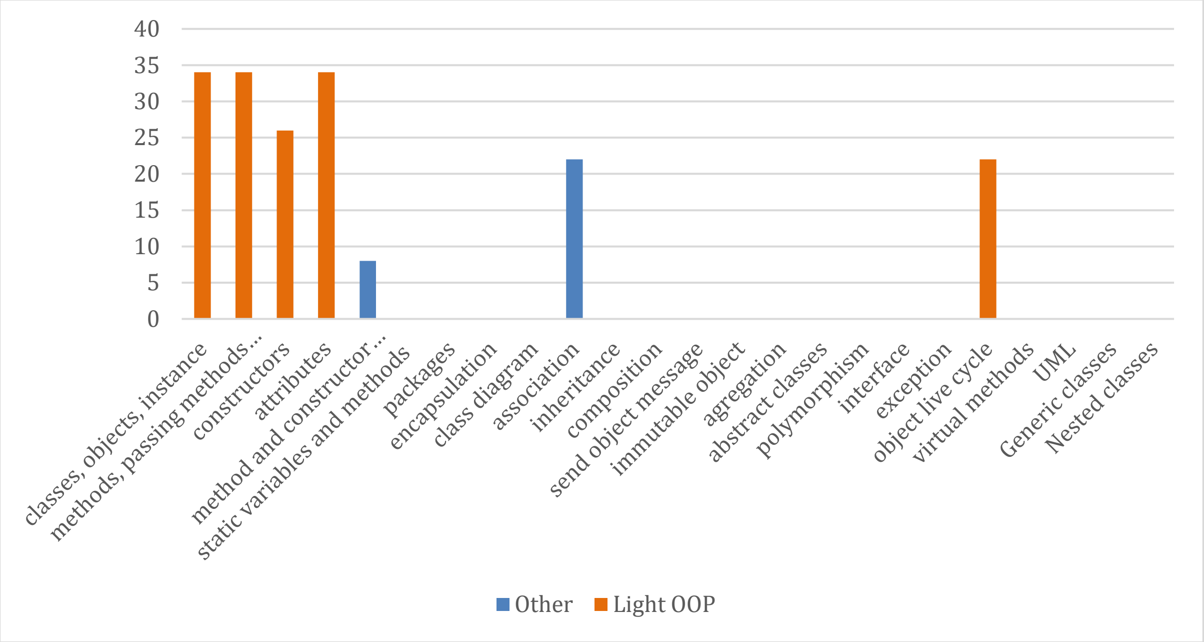

FIGURE 7 - INFORMATICS SUBJECT - OOP TOPICS HOURS

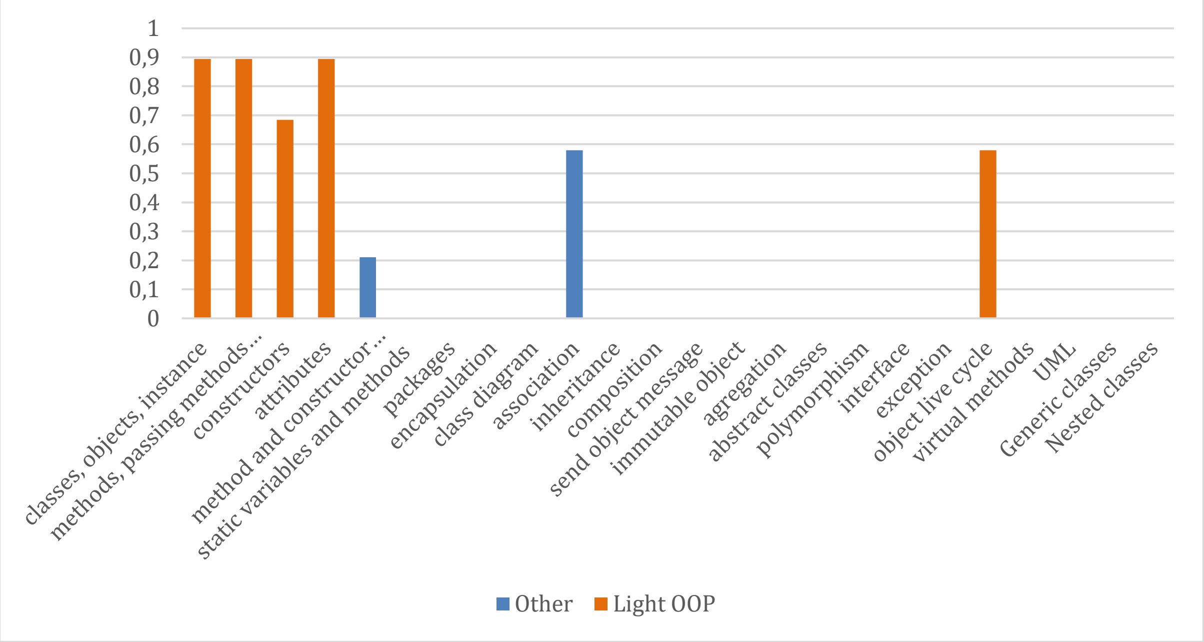

FIGURE 8 - INFORMATICS SUBJECT - PERCENTAGE OF OOP TOPICS HOURS

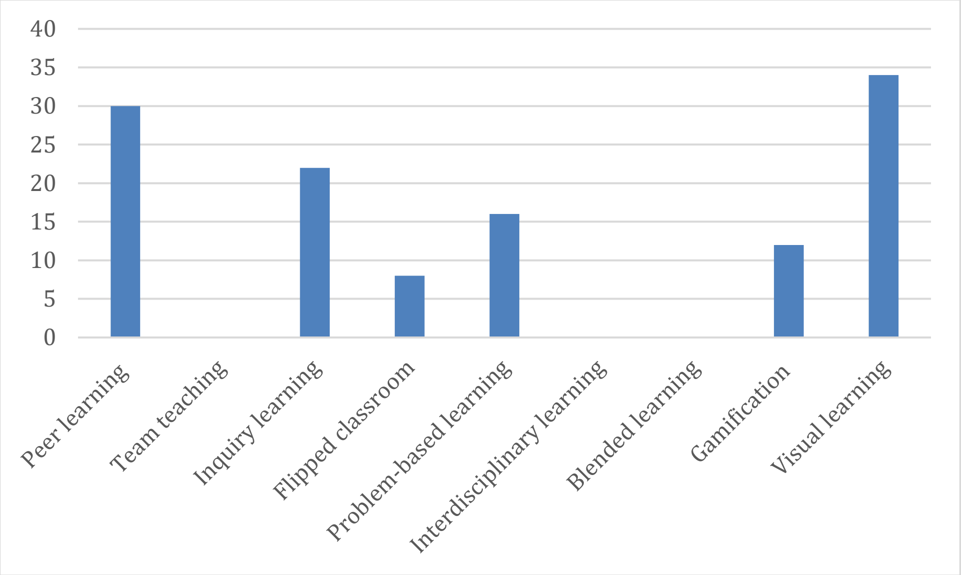

FIGURE 9 - INFORMATICS SUBJECT, LTA

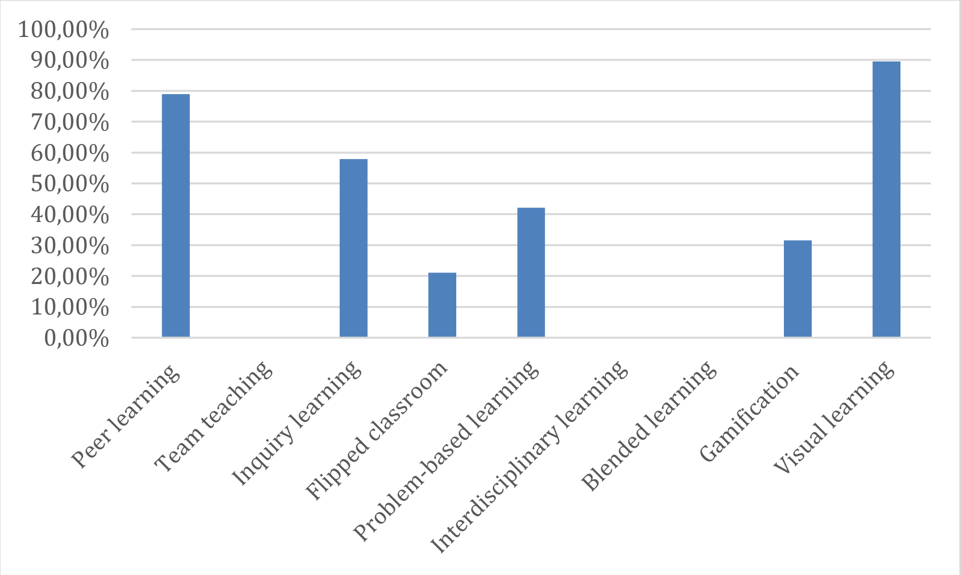

FIGURE 10 - INFORMATICS SUBJECT - LTA PERCENTAGE

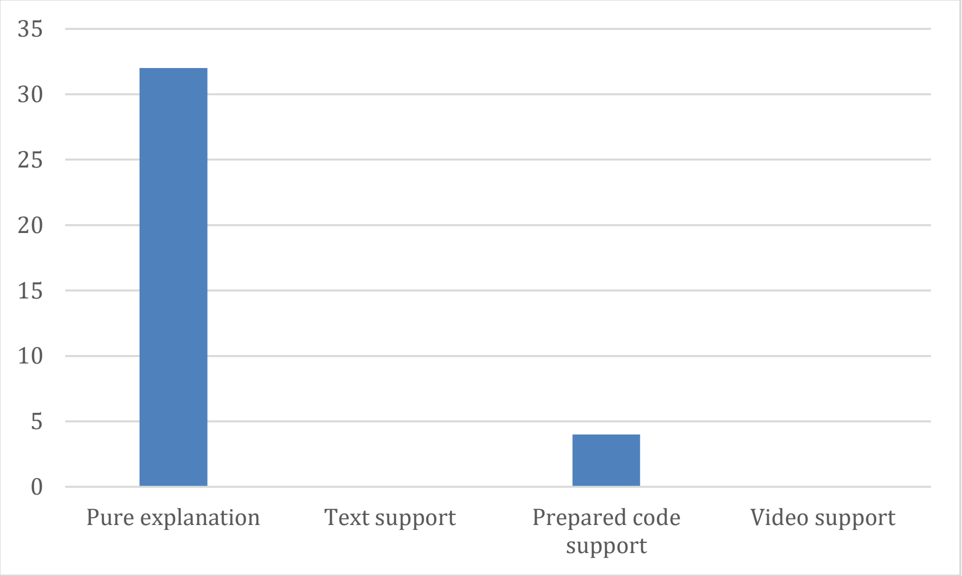

FIGURE 11 - INFORMATICS SUBJECT - SUPPORT

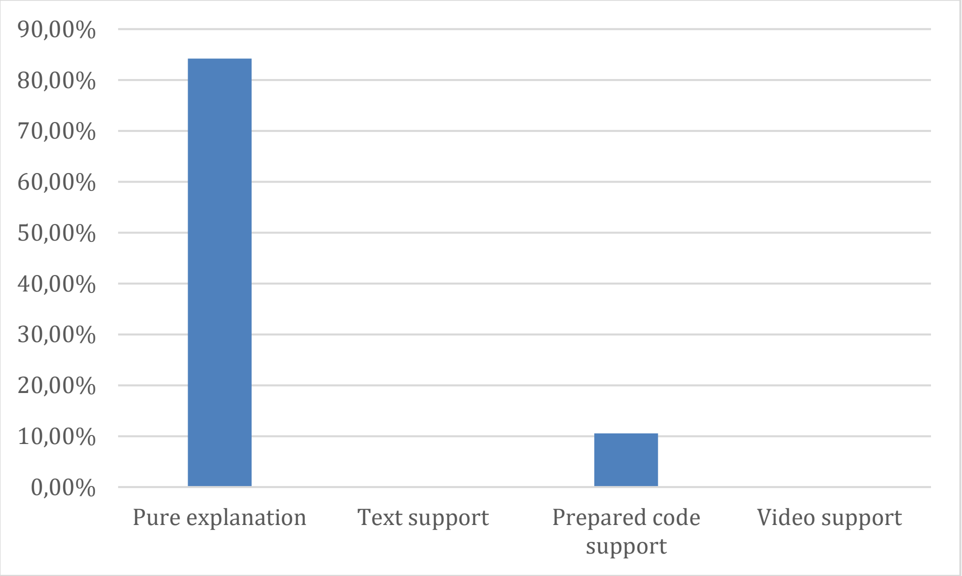

FIGURE 12 - INFORMATICS SUBJECT - SUPPORT PERCENTAGE

3.2.2. Subject “Seminar of programming”

Table 7 - Seminar pro programming subject - first year

Topic		Name	Hours	SW
1.		Classroom rules (mandatory in CZ for computer room)	1	
2.		Algorithm – definition, complexity, examples of algorithms	3	
3.	Introduction to GF	Greenfoot and the Java language – world, object, class, program structure, control of objects, method act	2	Greenfoot
4.		Working with objects – attributes, methods and constructors, IF statement, variables	2	Greenfoot
5.		Communication between objects, events, object life cycle	2	Greenfoot
6.		Object inheritance	2	Greenfoot
7.		For loop, stop the game (Greenfoot STOP)	2	Greenfoot
8.		Foreach loop, array, work with set of objects (getObjectsAt)	2	Greenfoot
9.		Independent work – creating your own simple game (e.g. pong)	6	Greenfoot
10.	Game creation	Game preparations – creation of objects and methods	4	Greenfoot
11.		Inheritance, encapsulation	2	Greenfoot
12.		Timer, casting in Java, intersection of two objects (getOneIntersectionObject)	2	Greenfoot
13.		Abstract methods and classes, while loop	2	Greenfoot
14.		Static methods and variables, worlds switching, arrays	2	Greenfoot
15.		intersection of more objects (getIntersectionObjects), method overloading, for each loop, switch case	2	Greenfoot
16.		Menu creation	2	Greenfoot
17.		Application completion	8	Greenfoot
18.		Jetbrains and java – simple methods	2	IntelliJ IDEA
19.		Array, basic operations, pass by value, by reference	2	IntelliJ IDEA
20.		Sorting	1	IntelliJ IDEA
21.		Dynamic array	3	IntelliJ IDEA
22.		Recursion	6	IntelliJ IDEA
23.		Java strings	6	IntelliJ IDEA

TABLE 8 - SEMINAR OF PROGRAMMING SUBJECT - FIRST YEAR - LIST OF OOP TOPICS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
3	**Y**	**Y**	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
4	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
5	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	**Y**	N	N	N	N
6	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	**Y**	N	N	N	N	N	N	N	N	**Y**	N	N	N	N
7	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	**Y**	N	**Y**	N	N	N	N	N	N	**Y**	N	N	N	N
8	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	**Y**	N	**Y**	N	N	N	N	N	N	**Y**	N	N	N	N
9	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	**Y**	N	**Y**	N	N	N	N	N	N	**Y**	N	N	N	N
10	**Y**	**Y**	**Y**	**Y**	N	N	N	N	N	N	**Y**	N	**Y**	N	N	N	N	N	N	**Y**	N	N	N	N
11	**Y**	**Y**	**Y**	**Y**	N	N	N	**Y**	N	N	**Y**	N	**Y**	N	N	N	N	N	N	**Y**	N	N	N	N
12	**Y**	**Y**	**Y**	**Y**	N	N	N	**Y**	N	N	**Y**	N	**Y**	N	N	N	N	N	N	**Y**	N	N	N	N
13	**Y**	**Y**	**Y**	**Y**	N	N	N	**Y**	N	N	**Y**	N	**Y**	N	N	**Y**	N	N	N	**Y**	N	N	N	N
14	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	N	**Y**	N	N	**Y**	N	**Y**	N	N	**Y**	N	N	N	**Y**	N	N	N	N
15	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	N	**Y**	N	N	**Y**	N	**Y**	N	N	**Y**	N	N	N	**Y**	N	N	N	N
16	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	N	**Y**	N	**Y**	**Y**	N	**Y**	N	N	**Y**	N	N	N	**Y**	N	N	N	N
17	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	N	**Y**	N	**Y**	**Y**	N	**Y**	N	N	**Y**	N	N	N	**Y**	N	N	N	N
18	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
19	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
20	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
21	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	N	**Y**	N	N	**Y**	N	N	N	N	N	N	N	N	**Y**	N	N	**Y**	N
22	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
23	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

TABLE 9 - SEMINAR OF PROGRAMMING SUBJECT - FIRST YEAR - LIST OF LTA + SUPPORT

	TLA									Support			
	1	2	3	4	5	6	7	8	9	1	2	3	4
1	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	Y	N	N	N
3	N	N	Y	N	N	N	N	Y	N	Y	N	N	N
4	N	N	Y	N	N	N	N	Y	N	Y	N	N	N
5	N	N	Y	N	N	N	N	Y	N	Y	N	N	N
6	N	N	Y	N	N	N	N	Y	N	N	N	N	Y
7	N	N	Y	N	N	N	N	Y	N	Y	N	N	N
8	N	N	Y	N	N	N	N	Y	N	Y	N	N	N
9	Y	Y	Y	Y	Y	N	N	Y	N	N	N	N	N
10	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
11	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
12	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
13	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
14	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
15	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
16	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
17	Y	Y	Y	N	Y	N	N	Y	N	N	Y	N	N
18	N	N	N	N	N	N	N	N	Y	N	N	N	N
19	N	N	N	N	N	N	N	N	Y	N	N	N	N
20	N	N	Y	N	Y	N	N	N	Y	N	N	N	N
21	N	N	Y	N	Y	N	N	N	Y	N	N	N	N
22	N	N	Y	N	Y	N	N	N	Y	N	N	N	N
23	N	N	N	N	N	N	N	N	Y	N	N	N	N

TABLE 10 - SEMINAR OF PROGRAMMING SUBJECT - SECOND YEAR

Topic		Name	Hours	SW
1.		Classroom rules (mandatory in CZ for computer room)	1	
2.		Repetition of the curriculum	5	IntelliJ IDEA
3.		Regular expressions	4	IntelliJ IDEA
4.		Multidimensional array	4	IntelliJ IDEA
5.		Exception	4	IntelliJ IDEA
6.		Text files	8	IntelliJ IDEA
7.		Linked list	4	IntelliJ IDEA
8.		Interface	4	IntelliJ IDEA
9.		Stack and queue and its using (for ex. Replacement of recursion)	6	IntelliJ IDEA
10.		Advanced sorting algorithms	6	IntelliJ IDEA
11.		Theory of computer networks, HW, SW, OSI model, TCP/IP model, WAN networks, transmission media	2	
12.		Final repetition and preparing for leaving exam	8	

TABLE 11 - SEMINAR OF PROGRAMMING SUBJECT - SECOND YEAR - LIST OF OOP TOPICS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
3	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
4	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
5	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	Y	Y	N	N	N	N
6	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	Y	N	N	N	N	Y	Y	N	N	N	N
7	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	N	Y	N	N	N	N	N	N	Y	N	N	Y	N
8	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	N	Y	N	N	N	N	Y	N	Y	N	N	Y	N
9	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	N	Y	N	N	N	N	Y	N	Y	N	N	Y	N
10	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
11	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
12	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

Table 12 - Seminar of programming subject - second year - list of LTA + Support

	TLA									Support			
	1	2	3	4	5	6	7	8	9	1	2	3	4
1	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	**Y**	N	N	N
3	N	N	N	N	**Y**	N	N	N	N	**Y**	**Y**	N	N
4	N	N	N	N	**Y**	N	N	N	N	**Y**	N	N	N
5	N	N	N	N	**Y**	N	N	N	N	**Y**	N	N	N
6	N	N	N	N	**Y**	N	N	N	N	**Y**	N	N	N
7	**Y**	N	**Y**	N	**Y**	N	N	N	N	**Y**	N	N	N
8	**Y**	N	**Y**	N	**Y**	N	N	N	N	N	N	N	**Y**
9	**Y**	N	**Y**	N	**Y**	N	N	N	N	**Y**	N	N	**Y**
10	**Y**	N	**Y**	N	**Y**	N	N	N	N	**Y**	N	N	N
11	N	N	N	N	N	N	N	N	N	N	N	N	N
12	N	N	N	N	N	N	N	N	N	N	N	N	N

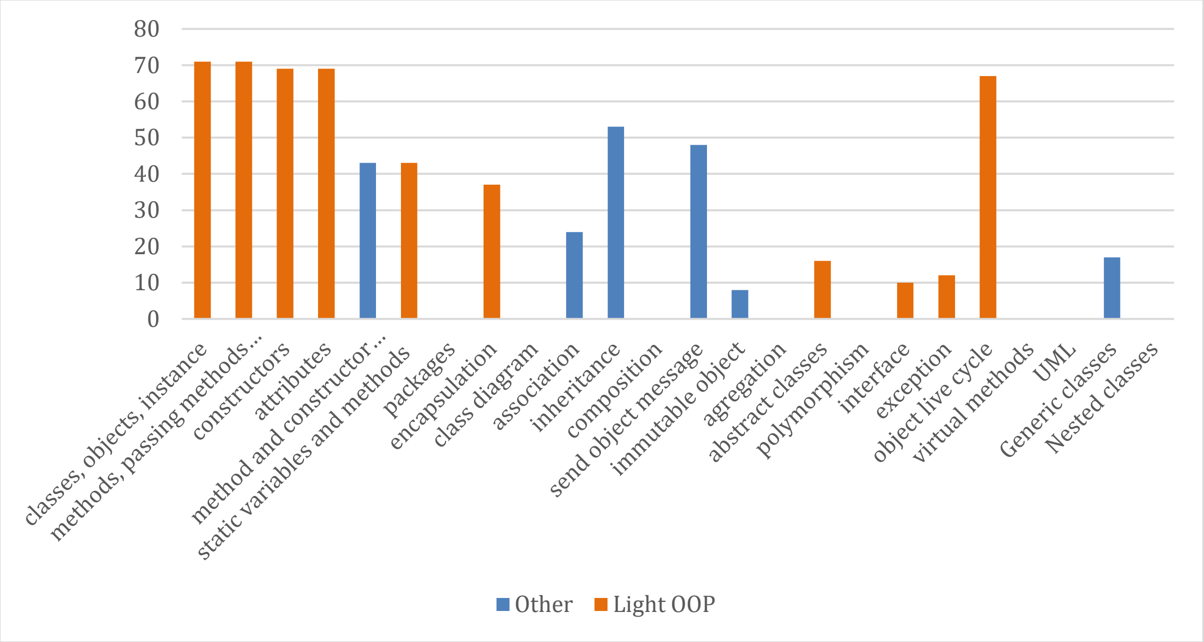

FIGURE 13 - SEMINAR OF PROGRAMMING SUBJECT- OOP TOPICS HOURS

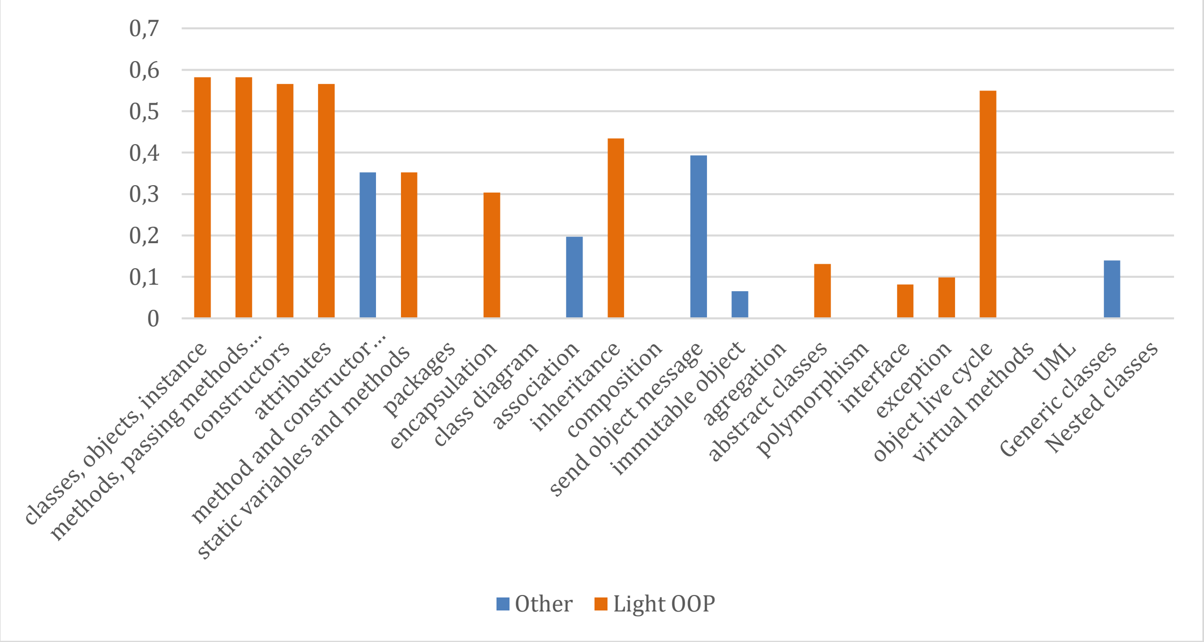

FIGURE 14 - SEMINAR OF PROGRAMMING SUBJECT - PERCENTAGE OF OOP TOPICS HOURS

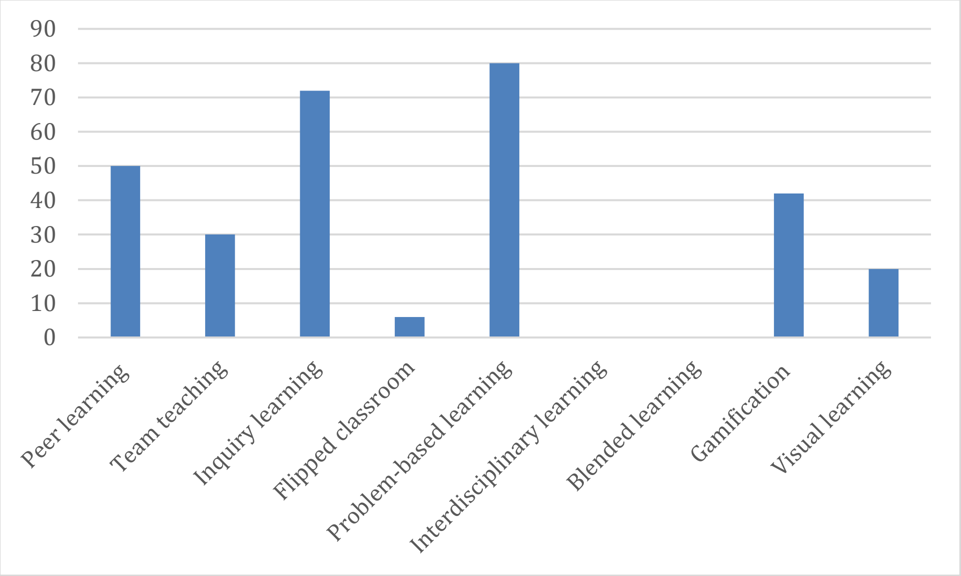

FIGURE 15 - SEMINAR OF PROGRAMMING SUBJECT - LTA

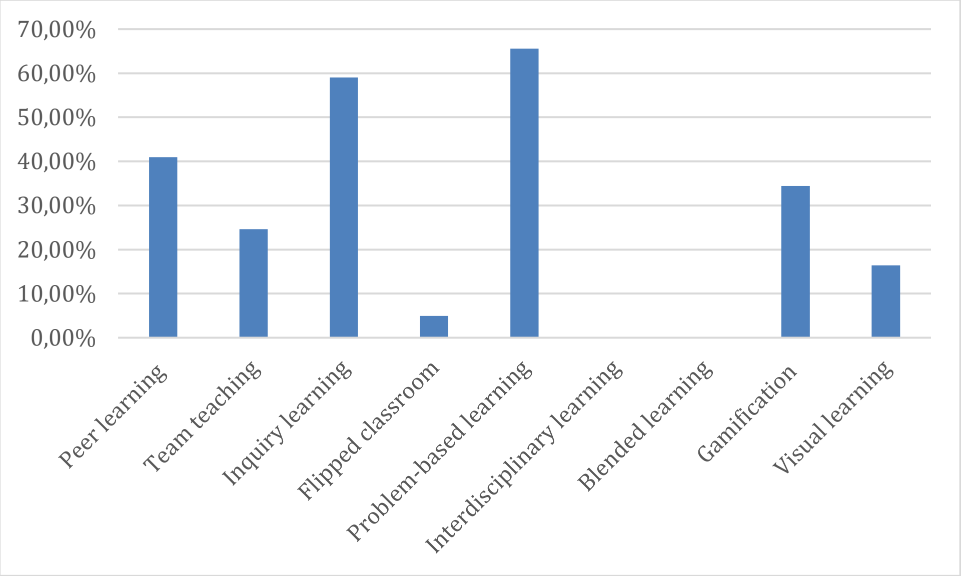

FIGURE 16 - SEMINAR OF PROGRAMMING SUBJECT - LTA PERCENTAGE

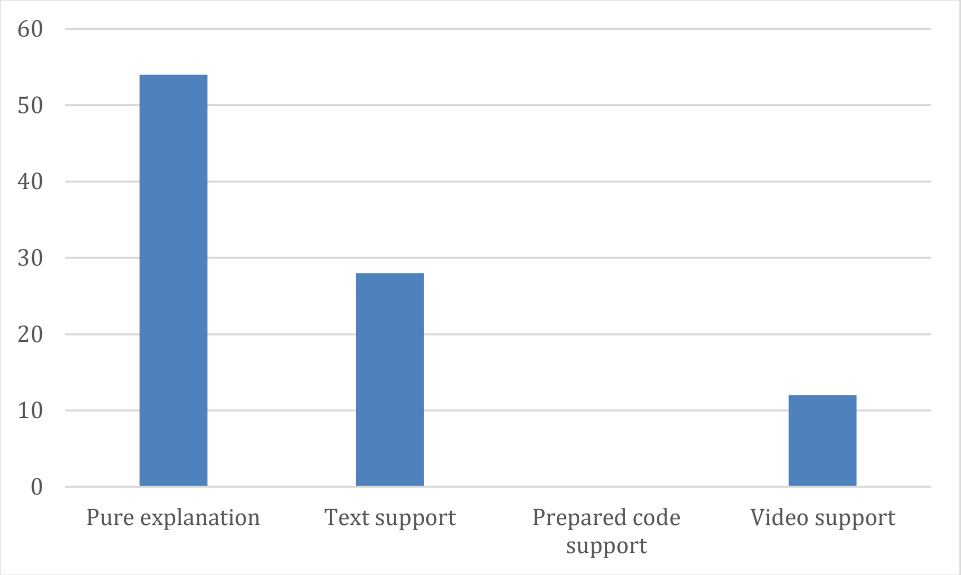

FIGURE 17 - SEMINAR OF PROGRAMMING - SUPPORT

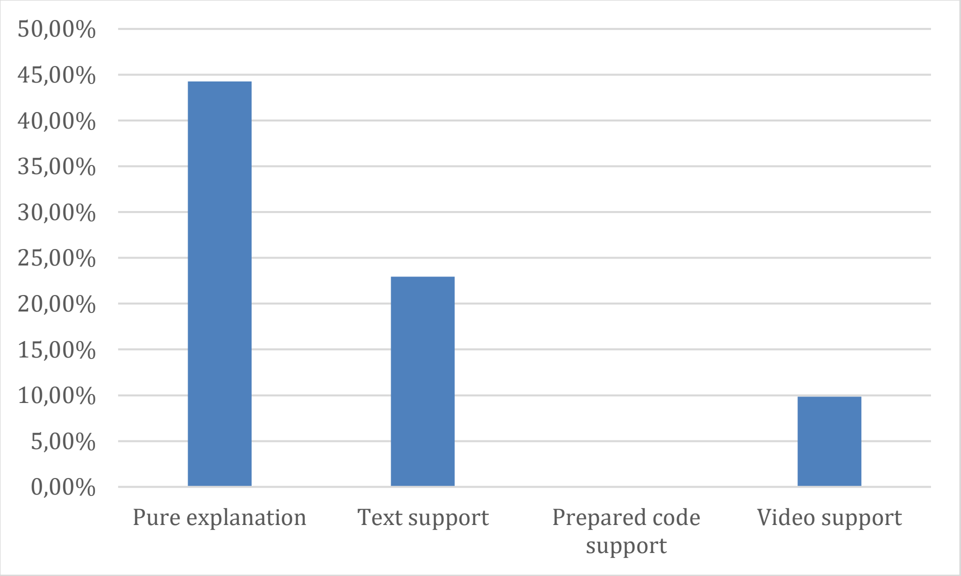

FIGURE 18 - SEMINAR OF PROGRAMMING - SUPPORT PERCENTAGE

3.3. New syllabus in Germany – Saxony

In Gymnasium Dresden Plauen the project results were used in the new curriculum of the subject “Computer science” (Informatik) for students at 11th year of study. It equivalent to 3th year of study in other countries. This subject includes concepts of OOP principals. The learning area comprises 20 lessons (45 minutes each). Learning area "8B: software development" in the elective part of the curriculum allows to extend the topic with another 8 lessons. By using this learning area and the learning area 7 “computer science project” (10 lessons), it is possible to implement parts of the OOP4Fun. The subject related to project results can have 38 hours.

3.3.1. Subject “Computer science”

TABLE 13 - COMPUTER SCIENCE SUBJECT

Topic	Name	Hours	SW
1.	Introduction/Repetition basic concepts of programming (instructions, functions, branches, loops)	4	
2.	data structures	4	
3.	recursion and iteration	4	
4.	efficiency, complexity and computability	2	
5.	OOP basics: classes and objects	4	Greenfoot
6.	OOP basics: inheritance	2	Greenfoot
7.	basic project management and git	2	Greenfoot & git
8.	group work on game projects (part 1)	4	Greenfoot & git
9.	interim review	2	Greenfoot & git
10.	OOP: additional concepts	2	Greenfoot & git
11.	group work on game projects (part 2)	6	Greenfoot & git
12.	presentations	2	Greenfoot & git

TABLE 14 - COMPUTER SCIENCE SUBJECT - LIST OF OOP TOPICS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
3	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
4	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
5	Y	Y	Y	Y	Y	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6	N	N	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
7	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
8	Y	Y	Y	Y	Y	N	N	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	Y	N	N
9	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
10	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	Y	N	N	Y	N	N	N
11	Y	Y	Y	Y	Y	N	N	Y	Y	Y	Y	N	N	N	N	N	Y	Y	N	N	Y	Y	N	N
12	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

TABLE 15 - COMPUTER SCIENCE SUBJECT - LIST OF LTA + SUPPORT

	TLA									Support			
	1	2	3	4	5	6	7	8	9	1	2	3	4
1	N	N	Y	N	N	Y	N	N	N	Y	Y	Y	N
2	N	N	Y	N	N	Y	N	N	N	Y	Y	Y	N
3	N	N	Y	N	N	Y	N	N	N	Y	Y	Y	N
4	N	N	N	N	N	N	N	N	N	Y	Y	N	N
5	N	N	Y	N	N	Y	N	N	N	Y	Y	Y	N
6	N	N	Y	N	N	Y	N	N	N	Y	Y	Y	N
7	N	N	N	N	N	N	N	N	N	Y	Y	N	N
8	Y	N	N	N	Y	Y	N	N	N	N	N	N	N
9	N	N	N	N	N	N	N	N	N	N	N	N	N
10	N	N	N	N	N	N	N	N	N	Y	Y	Y	N
11	Y	N	N	N	Y	Y	N	N	N	N	N	N	N
12	N	N	N	N	N	N	N	N	N	N	N	N	N

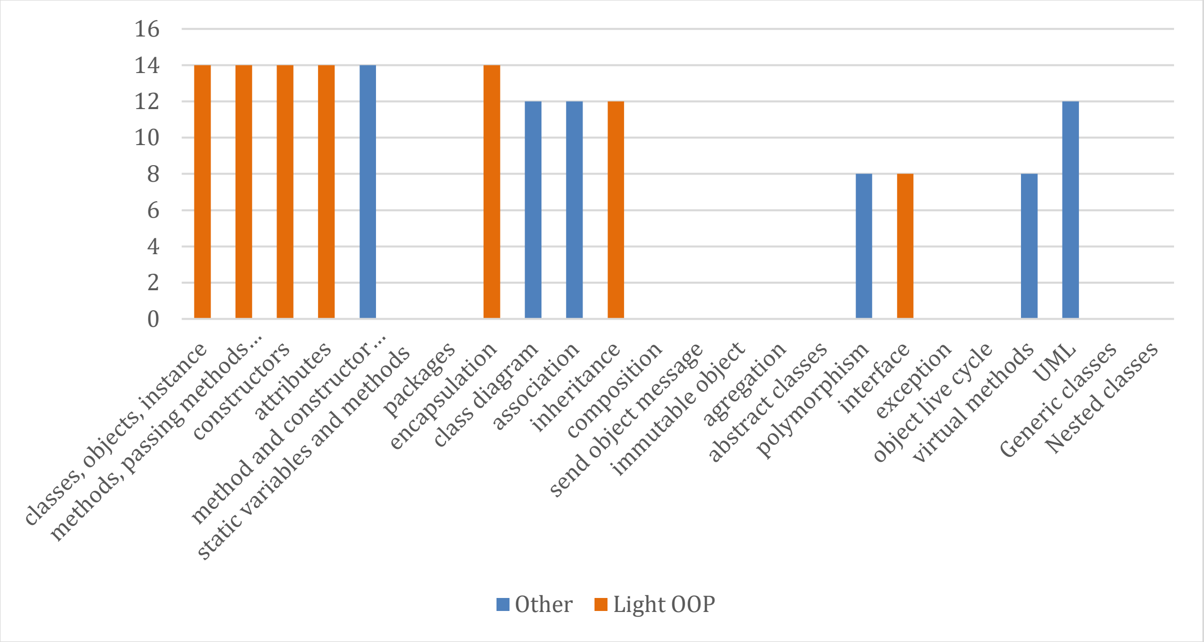

FIGURE 19 - COMPUTER SCIENCE SUBJECT - OOP TOPICS HOURS

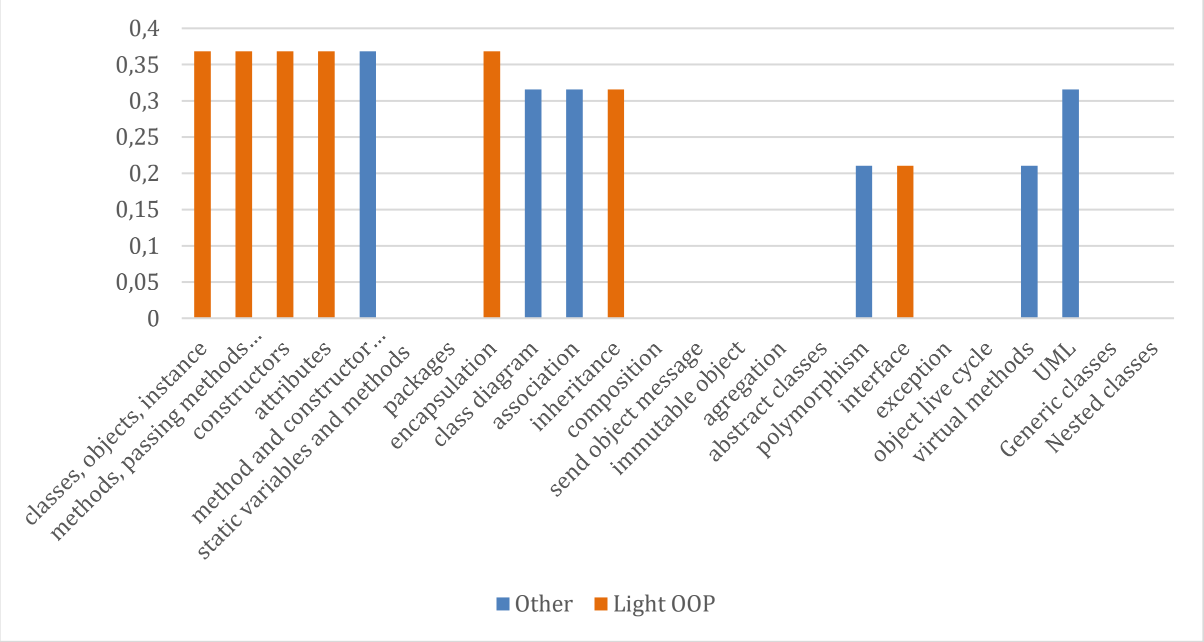

FIGURE 20 - COMPUTER SCIENCE SUBJECT - PERCENTAGE OF OOP TOPICS HOURS

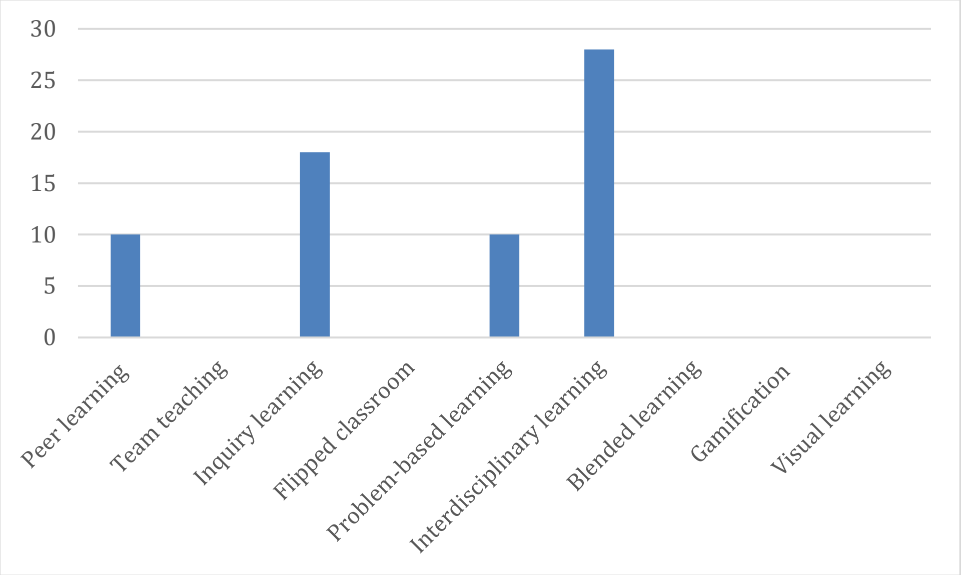

FIGURE 21 - COMPUTER SCIENCE SUBJECT - LTA

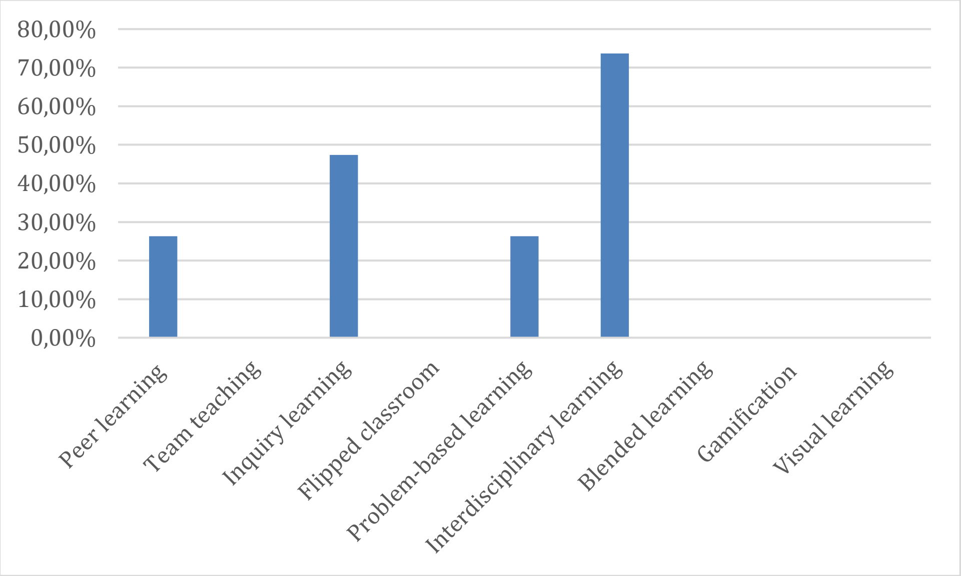

FIGURE 22 - COMPUTER SCIENCE SUBJECT - LTA PERCENTAGE

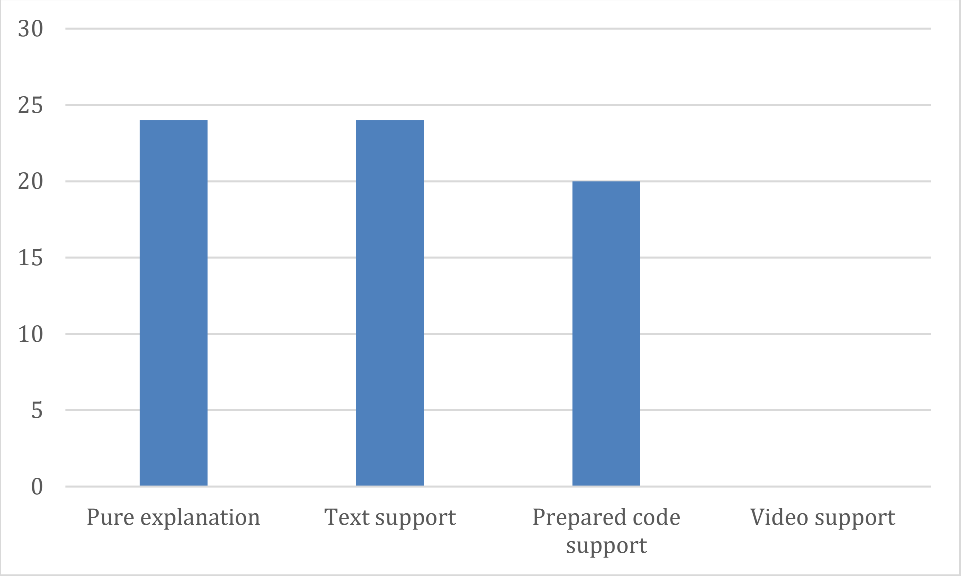

FIGURE 23 - COMPUTER SCIENCE SUBJECT - SUPPORT

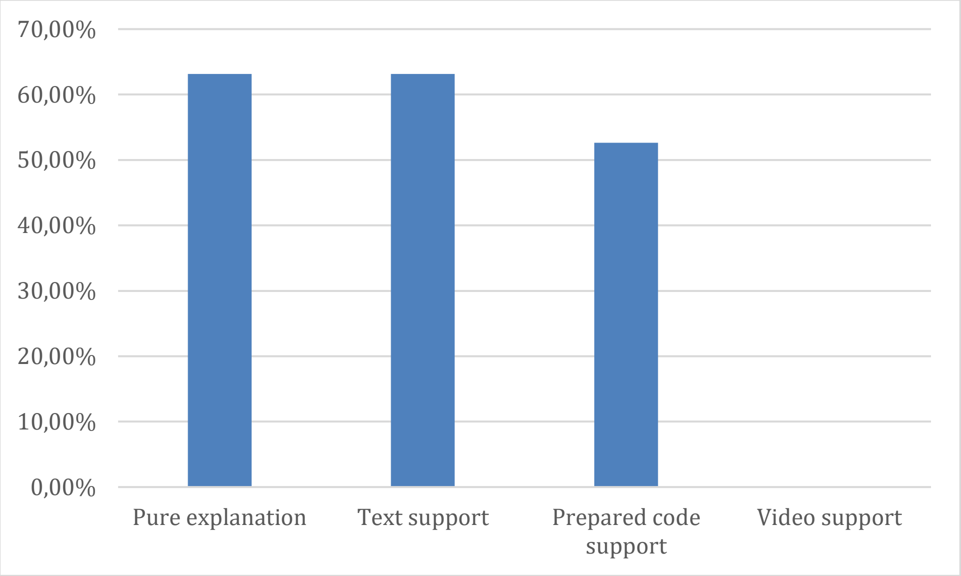

FIGURE 24 - COMPUTER SCIENCE SUBJECT - SUPPORT PERCENTAGE

3.3.2. Explanation course selection and student motivation

In grade 10 students choose their subjects for grade 11 and 12. Computer science, the course which includes object oriented programming and can be adapted to use the OOP4Fun-Curriculum, can be chosen by students to replace one of the following courses:

- geography
- political science („Gemeinschaftskunde/Rechtserziehung/Wirtschaft“)
- biology
- chemistry
- physics

Typically teachers approach high-performing and interested students in computer science in grade 10 to motivate them to choose computer science in grade 11/12.

Additionally subject teachers often outline the topics of computer science in grade 11/12 to motivate interested students to choose the subject.

3.4. New syllabus in Serbia

In Gimnazija Ivanjica there was innovated one mandatory subject called Object-oriented programming. In the school there are 144 classes of OOP in the 3rd year of study. There is no possibility to choose another optional subject with programming topic.

3.4.1. Subject "Object oriented programming"

Table 16 - Object oriented programming subject

Topic	Game	Name	Hours	SW
1		Introduction to the subject, classroom rules, student obligations	1	
2		Basic concepts of object-oriented programming	41	
3		Principles of inheritance and polymorphism	70	
4		Written assignments	6	
5		Motivational game: Catch the fly	2	Greenfoot
6		Git	2	GitHub, SourceTree
7	Tower defense game	Greenfoot and the Java language - object, class	3	Greenfoot
8		Algorithm, Application Control	3	Greenfoot
9		Branching, enemy control	4	Greenfoot
10		Variable and expressions	3	Greenfoot
11		Association	4	Greenfoot
12		Inheritance	4	Greenfoot
13		Encapsulation	3	Greenfoot
14	Student games	Independent work of students - project	2	Greenfoot

Table 17 - Object oriented programming subject - list of OOP topics

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
2	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	N	Y	Y	N	Y	Y	Y
3	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	N	Y	Y	N	Y	Y	Y
4	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	N	Y	Y	N	Y	Y	Y
5	Y	Y	Y	Y	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
6	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
7	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
8	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
9	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N
10	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N
11	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	N	N	N	N	N	Y	N	N	N	Y	N	Y	N	N
12	Y	Y	Y	Y	Y	N	N	Y	Y	N	Y	N	N	N	N	Y	N	N	N	Y	N	Y	N	N
13	Y	Y	Y	Y	Y	Y	N	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
14	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	N	N	Y	N	N	N	Y	N	Y	N	N

Table 18 - Object oriented programming subject - list of LTA + support

	LTA									Teacher support			
	1	2	3	4	5	6	7	8	9	1	2	3	4
1	N	N	N	N	N	N	N	N	N	N	N	N	N
2	Y	Y	Y	N	Y	N	Y	N	Y	Y	Y	Y	Y
3	Y	Y	Y	N	Y	N	Y	N	N	Y	Y	Y	Y
4	Y	Y	Y	N	Y	N	Y	N	N	Y	Y	Y	Y
5	N	N	Y	N	Y	N	N	Y	N	Y	Y	Y	N
6	Y	N	Y	N	N	N	N	Y	N	Y	Y	N	N
7	N	N	Y	N	N	N	Y	Y	N	Y	Y	N	N
8	Y	N	Y	N	Y	N	Y	Y	N	Y	Y	N	N
9	Y	N	Y	N	Y	N	Y	Y	N	Y	Y	N	N
10	Y	N	Y	N	Y	N	Y	Y	N	Y	Y	Y	N
11	Y	N	Y	N	Y	N	Y	Y	N	Y	Y	Y	N
12	Y	Y	Y	N	Y	N	Y	Y	N	Y	Y	Y	N
13	Y	Y	Y	N	Y	N	Y	Y	N	Y	Y	Y	N
14	Y	Y	Y	Y	Y	N	Y	Y	N	Y	N	N	Y

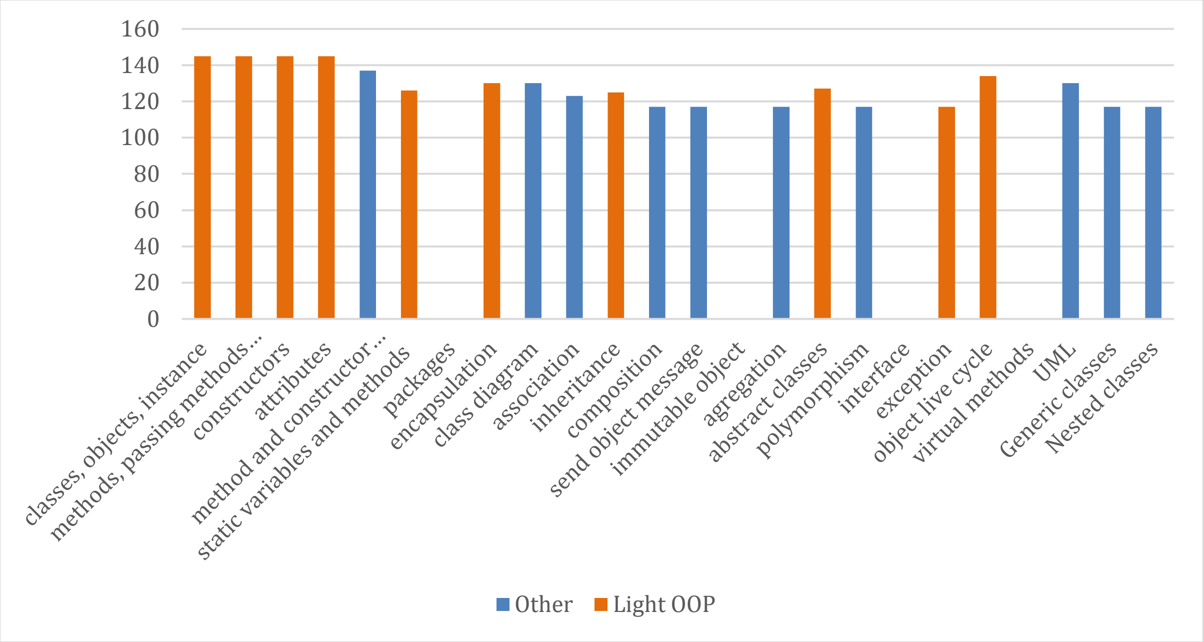

FIGURE 25 - OBJECT ORIENTED PROGRAMMING SUBJECT - OOP TOPICS HOURS

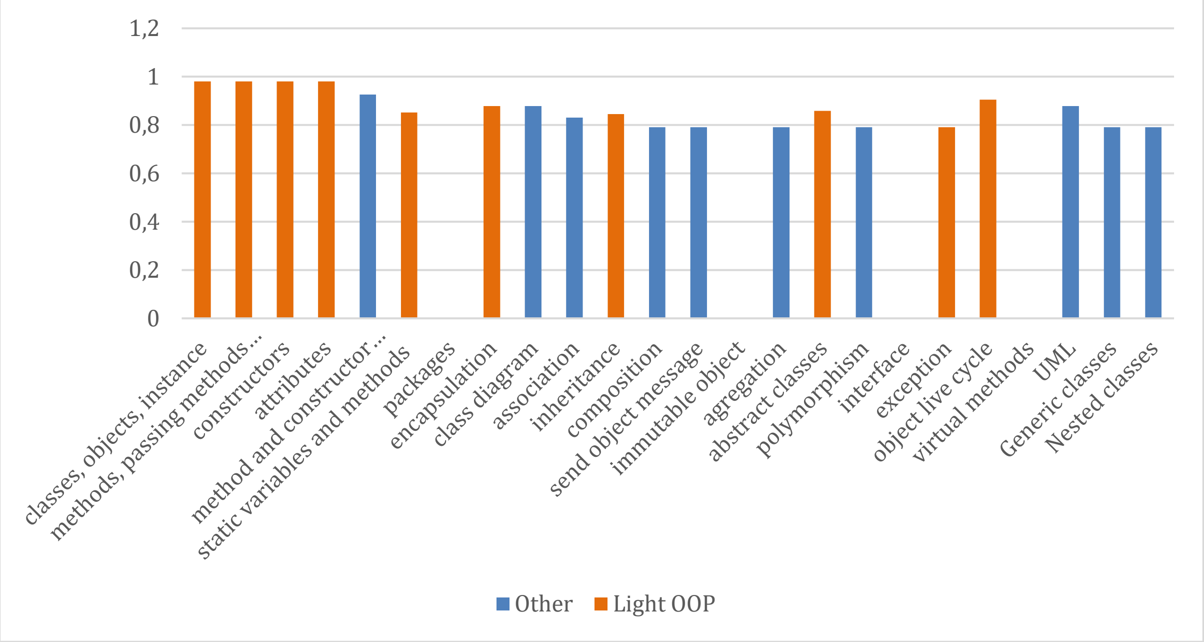

FIGURE 26 - OBJECT ORIENTED PROGRAMMING SUBJECT - PERCENTAGE OF OOP TOPICS HOURS

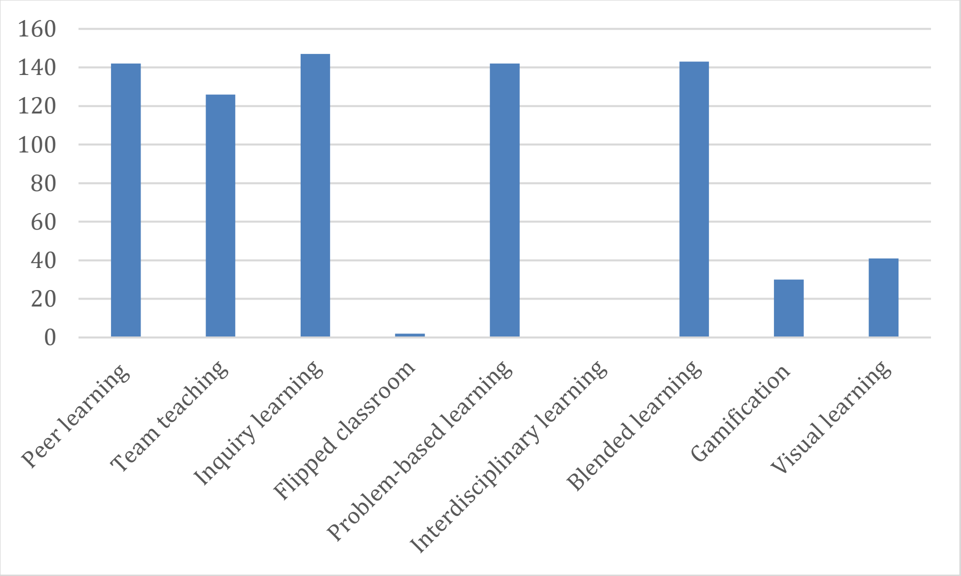

FIGURE 27 - OBJECT ORIENTED PROGRAMMING SUBJECT - LTA

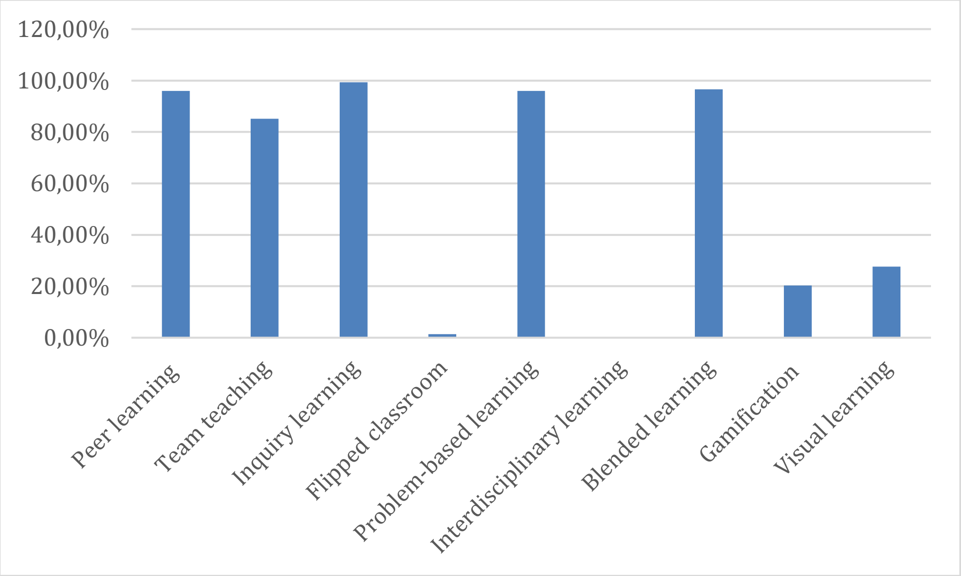

FIGURE 28 - OBJECT ORIENTED PROGRAMMING SUBJECT - LTA PERCENTAGE

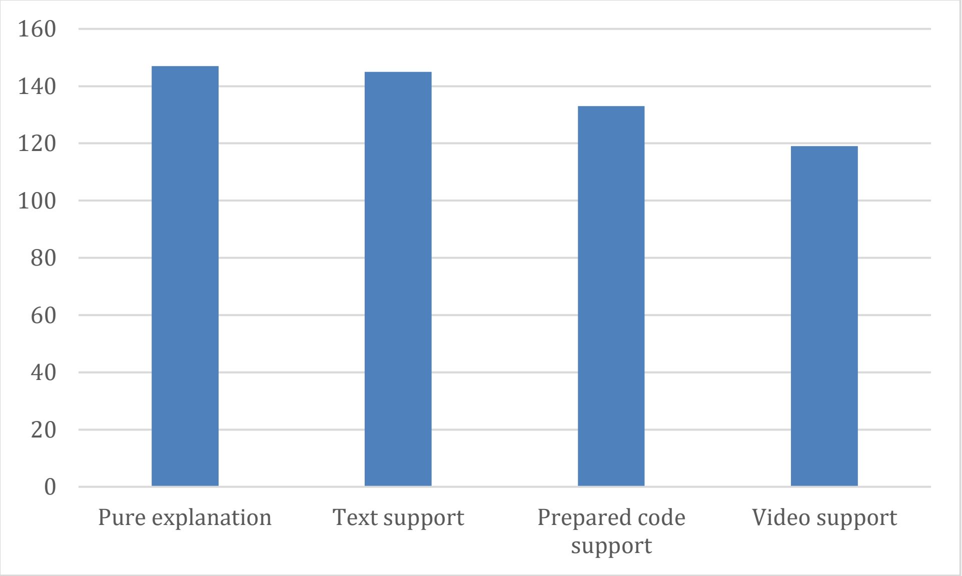

FIGURE 29 - OBJECT ORIENTED PROGRAMMING SUBJECT - SUPPORT

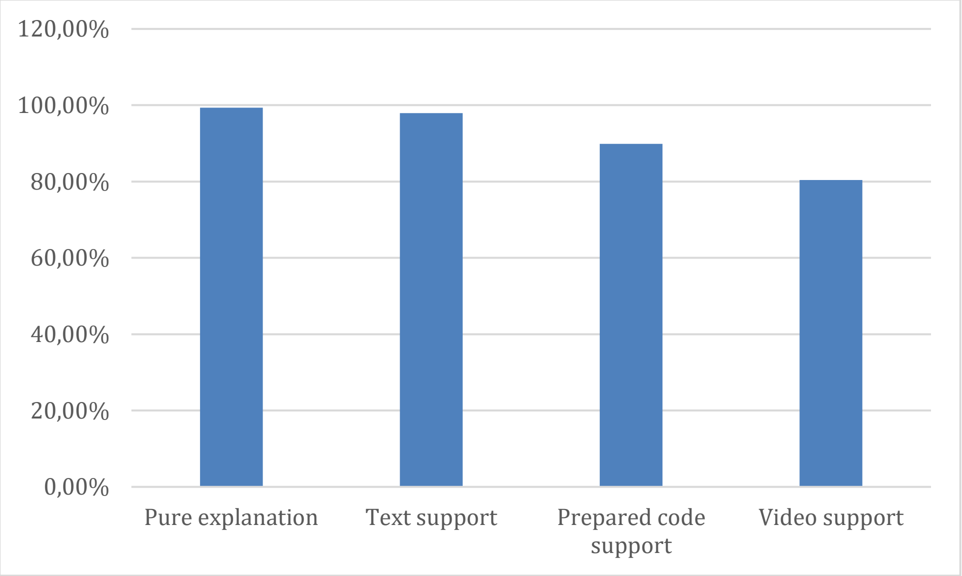

FIGURE 30 - OBJECT ORIENTED PROGRAMMING SUBJECT - SUPPORT PERCENTAGE

3.5. New syllabus in Slovakia

Mandatory subject “Applied informatics seminar” was created In Slovakia. It is a subject for 3rd a 4th year of study. Since it is a 2-year subject, the syllabus table will be divided by year. As in Ivanjica there is no possibility to choose optional subject with OOP topics.

3.5.1. Subject “Applied Informatics Seminar subject”

TABLE 19 - APPLIED INFORMATICS SEMINAR SUBJECT – FIRST YEAR

Topic		Name	Hours	SW
1.	Introduction to algorithmization	Introduction to the course	1	
2.		Logic tasks	1	
3.		The concept of algorithm	1	
4.		Formal notation of algorithms - flowchart	1	
5.		Solving simple algorithms	2	
6.	Introduction to OOP	Object, class, instance	2	
7.		UML - class diagram	1	Simple UML modeler
8.		Java language and Greenfoot environment	2	Greenfoot
9.		Example of a simple application	1	Greenfoot
10.		Working with Greenfoot environment - control components	2	Greenfoot
11.	Basics of algorithmization in OOP and Greenfoot environment	Creating a new project	1	Greenfoot
12.		Class World and Meadow	1	Greenfoot
13.		Documentation and its role in programming	1	Greenfoot
14.		Class Actor and Stone	2	Greenfoot
15.		Interaction of meadow with stone	1	Greenfoot
16.		Examining the internal state of an instance	1	Greenfoot
17.		Creating a meadow with stones - constructor	1	Greenfoot
18.		Class Player - attributes, constructor	2	Greenfoot
19.		Class Player - movement and act method	3	Greenfoot
20.		Class Player - user input	6	Greenfoot
21.		Class Player - solving collisions with stone and player	5	Greenfoot
22.		Class Meadow - adding player(s)	4	Greenfoot
23.		Generating a static meadow	4	Greenfoot
24.		Meadow parameterization	2	Greenfoot
25.		Class Food - attributes, constructor	2	Greenfoot
26.		Class Food - collision with a player	4	Greenfoot
27.	Randomness	Randomness from a programming point of view	1	Greenfoot
28.		Greenfoot - random number generation	2	Greenfoot
29.		Class Meadow - food position	3	Greenfoot
30.		Dynamic meadow generation	4	Greenfoot

TABLE 20 - APPLIED INFORMATICS SEMINAR SUBJECT - FIRST YEAR - LIST OF OOP TOPICS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
3	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
4	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
5	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
6	Y	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
7	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
8	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
9	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
10	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N
11	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
12	Y	N	Y	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
13	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
14	Y	N	Y	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
15	N	Y	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N
16	Y	N	N	Y	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
17	Y	N	Y	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N
18	Y	N	Y	Y	N	N	N	Y	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
19	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
20	N	Y	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
21	N	Y	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N
22	N	Y	Y	Y	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
23	N	Y	Y	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N
24	N	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
25	Y	N	Y	Y	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
26	N	Y	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	Y	N	N	N	N
27	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
28	Y	Y	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N
29	N	Y	Y	N	N	N	N	Y	N	N	N	Y	Y	N	N	N	N	N	N	N	N	N	N	N
30	N	Y	Y	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N

TABLE 21 - APPLIED INFORMATICS SEMINAR SUBJECT - FIRST YEAR LIST OF LTA + SUPPORT

	TLA									Support			
	1	2	3	4	5	6	7	8	9	1	2	3	4
1	N	N	N	N	N	N	N	N	N	N	N	N	N
2	N	N	N	N	N	N	N	N	N	Y	N	N	N
3	N	N	N	N	N	N	N	N	N	Y	N	N	N
4	N	N	N	N	N	N	N	N	Y	Y	Y	N	N
5	Y	N	Y	N	N	N	N	N	Y	Y	N	N	N
6	N	N	N	N	N	N	N	N	N	Y	N	N	N
7	N	N	N	N	N	N	N	N	Y	Y	Y	N	N
8	N	N	N	N	N	N	N	N	N	Y	N	N	N
9	N	N	N	N	N	N	N	N	N	Y	N	Y	N
10	N	N	N	N	N	N	N	N	N	Y	N	N	N
11	N	N	N	N	N	N	N	N	N	Y	N	N	N
12	N	N	Y	N	N	Y	N	N	N	Y	N	N	N
13	N	N	N	N	N	N	N	N	N	Y	N	N	N
14	Y	N	Y	N	N	N	N	N	N	Y	N	N	N
15	N	N	Y	N	N	N	N	N	N	Y	N	N	N
16	N	N	N	N	N	N	N	N	N	Y	Y	N	N
17	N	N	Y	N	N	N	N	N	N	Y	N	N	N
18	N	N	Y	N	N	N	N	N	N	Y	N	N	N
19	N	N	Y	N	N	N	N	N	N	Y	Y	N	N
20	N	N	Y	N	N	N	N	N	N	Y	Y	N	N
21	N	N	Y	N	N	N	N	N	N	Y	N	N	N
22	N	N	Y	N	N	N	N	N	N	Y	N	N	N
23	Y	N	Y	N	N	N	N	N	N	Y	N	N	N
24	N	N	Y	N	N	N	N	N	N	Y	N	N	N
25	N	N	Y	N	N	N	N	N	N	Y	N	N	N
26	Y	N	Y	N	N	N	N	N	N	Y	N	N	N
27	N	N	N	N	N	Y	N	N	N	Y	Y	N	N
28	N	N	N	N	N	N	N	N	N	Y	N	Y	N
29	N	N	Y	N	N	N	N	N	N	Y	N	N	N
30	Y	N	N	N	Y	N	N	N	N	Y	N	N	N

TABLE 22 - APPLIED INFORMATICS SEMINAR SUBJECT - SECOND YEAR

Topic		Name	Hours	SW
1.	Repeating and deepening the knowledge from the 3rd year	Getting acquainted with the pre-prepared project	4	Greenfoot
2.		Inheritance - basics	1	Greenfoot
3.		Class Fruit and Worm	1	Greenfoot
4.		Class Food - common ancestor of Fruit and Worm	1	Greenfoot
5.		Class Fruit - method act	2	Greenfoot
6.		Class Worm - method act	4	Greenfoot
7.		Class Meadow - new types of food	2	Greenfoot
8.	Finalization of project and Greenfoot options	Class Meadow - multiple instances of classes Fruit and Worm	4	Greenfoot
9.		Class Player - score management	2	Greenfoot
10.		Greenfoot and Counter class	2	Greenfoot
11.		Goal of the game and time in Greenfoot environment	4	Greenfoot
12.		Class Menu	6	Greenfoot
13.	Custom project	Work on the project	11	Greenfoot
14.		Presentation of the project	2	Presentation software, Greenfoot

TABLE 23 - APPLIED INFORMATICS SEMINAR SUBJECT - SECOND YEAR - LIST OF OOP TOPICS

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Y	Y	Y	Y	N	N	N	Y	Y	N	N	Y	Y	N	N	N	N	N	N	Y	N	Y	N	N
2	N	N	N	N	N	N	N	Y	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
3	Y	N	Y	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
4	Y	N	Y	Y	N	N	N	Y	N	N	Y	N	N	N	N	N	Y	N	N	N	N	N	N	N
5	N	Y	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	Y	N	N	N	N	N	N	N
6	N	Y	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	Y	N	N	N	N	N	N	N
7	Y	N	Y	Y	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N	N	Y	N	N	N	N
8	Y	N	Y	Y	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N	N	N	N	N	N	N
9	N	Y	Y	Y	N	N	N	Y	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N
10	Y	Y	Y	Y	N	N	N	Y	N	N	N	N	Y	N	N	N	N	N	N	Y	N	N	N	N
11	N	Y	Y	Y	N	N	N	Y	N	N	N	N	Y	N	N	N	N	N	N	Y	N	N	N	N
12	Y	Y	Y	Y	N	N	N	Y	N	N	N	Y	Y	N	N	N	N	N	N	Y	N	N	N	N
13	Y	Y	Y	Y	N	N	N	Y	Y	N	Y	Y	Y	N	N	N	Y	N	N	Y	N	Y	N	N
14	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N

Table 24 - Applied Informatics Seminar subject - second year - list of LTA + support

	TLA									Support			
	1	2	3	4	5	6	7	8	9	1	2	3	4
1	N	N	N	N	N	N	N	N	Y	Y	N	Y	N
2	N	N	N	N	N	N	N	N	N	Y	Y	N	N
3	N	N	Y	N	N	N	N	N	N	Y	N	N	N
4	N	N	Y	N	N	N	N	N	N	Y	N	N	N
5	N	N	Y	N	N	N	N	N	N	Y	N	N	N
6	Y	N	N	N	Y	N	N	N	N	Y	N	N	N
7	N	N	Y	N	N	N	N	N	N	Y	N	N	N
8	N	N	Y	N	N	N	N	N	N	Y	N	N	N
9	N	N	Y	N	N	N	N	N	N	Y	N	N	N
10	N	N	N	N	N	N	N	N	N	Y	N	Y	N
11	N	N	Y	N	N	N	N	N	N	Y	N	Y	N
12	N	N	N	N	Y	N	N	N	N	Y	N	Y	N
13	Y	Y	N	Y	Y	Y	N	Y	N	N	N	N	N
14	Y	Y	N	Y	N	Y	N	Y	N	N	N	N	N

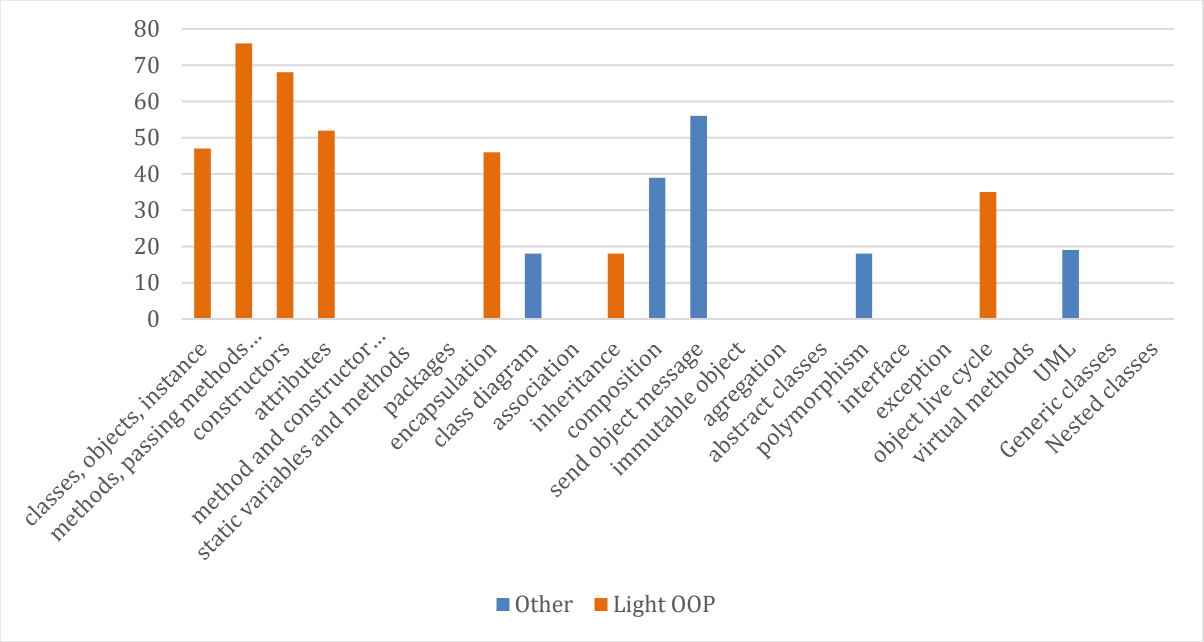

Figure 31 - Applied Informatics Seminar subject - OOP topics hours

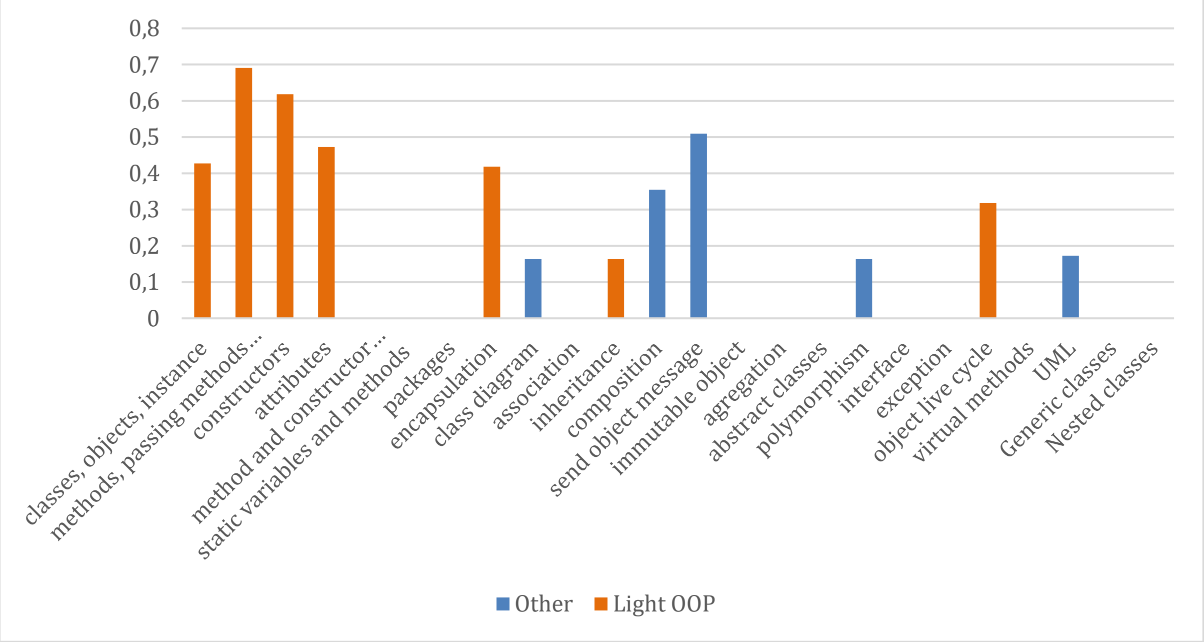

Figure 32 - Applied Informatics Seminar subject - percentage of OOP topics hours

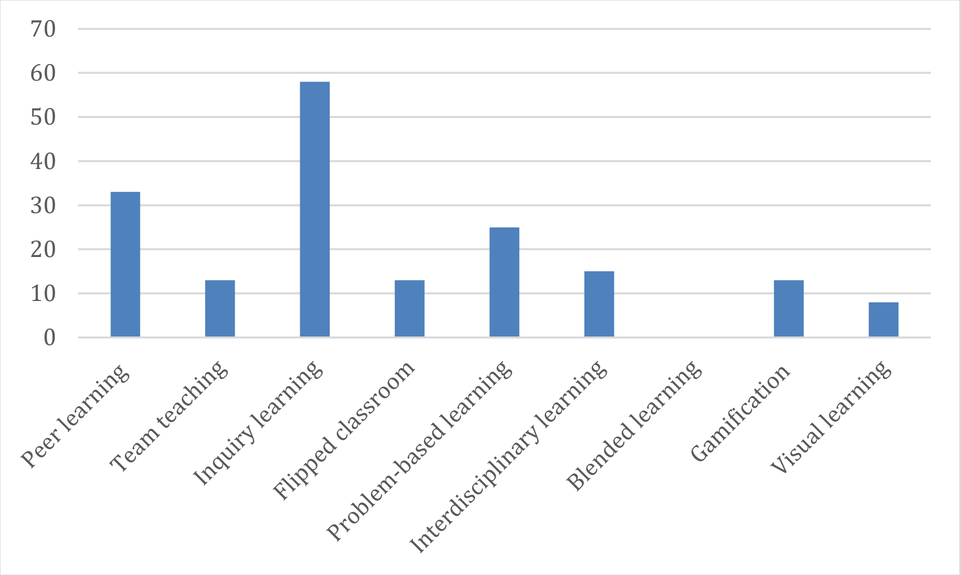

FIGURE 33 - APPLIED INFORMATICS SEMINAR SUBJECT - LTA

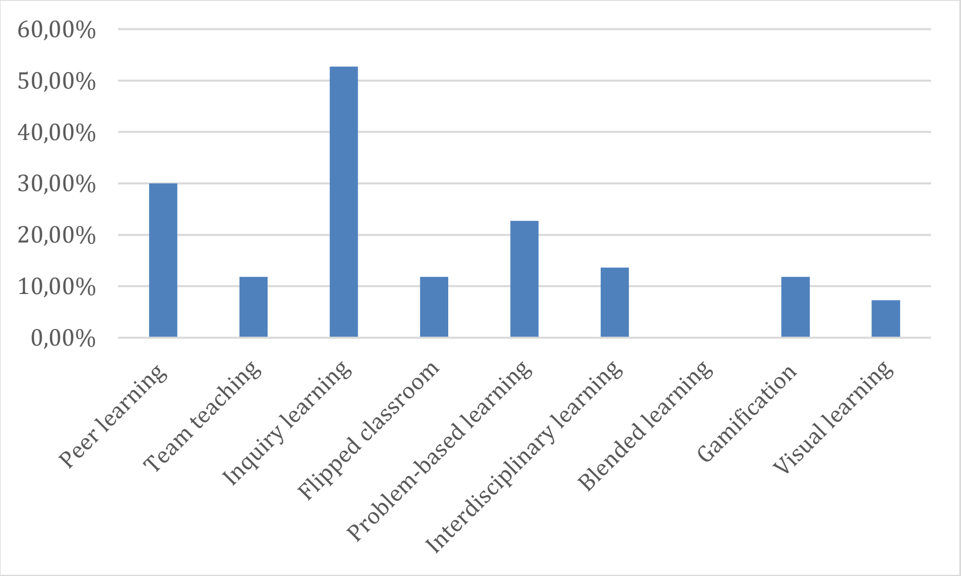

FIGURE 34 - APPLIED INFORMATICS SEMINAR SUBJECT - LTA PERCENTAGE

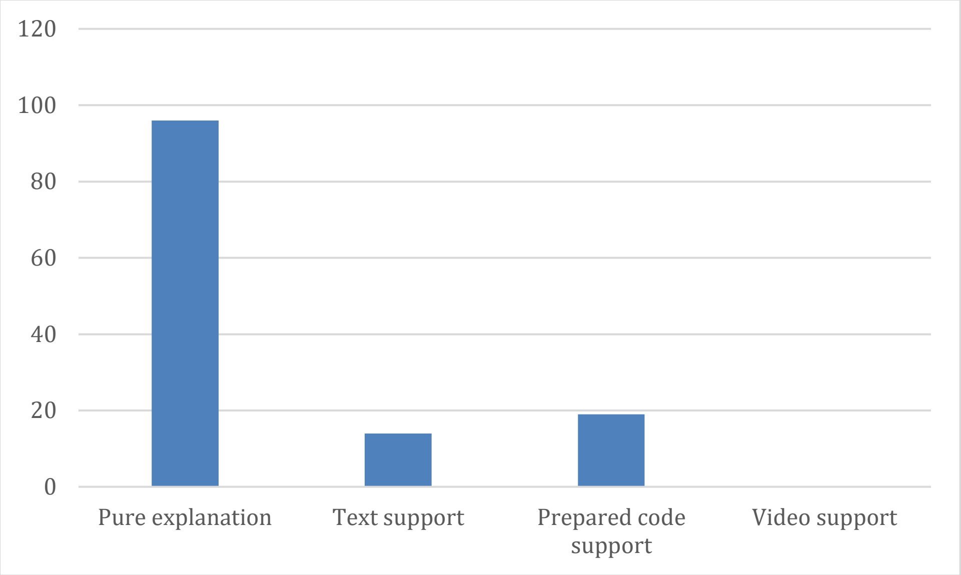

FIGURE 35 - APPLIED INFORMATICS SEMINAR SUBJECT - SUPPORT

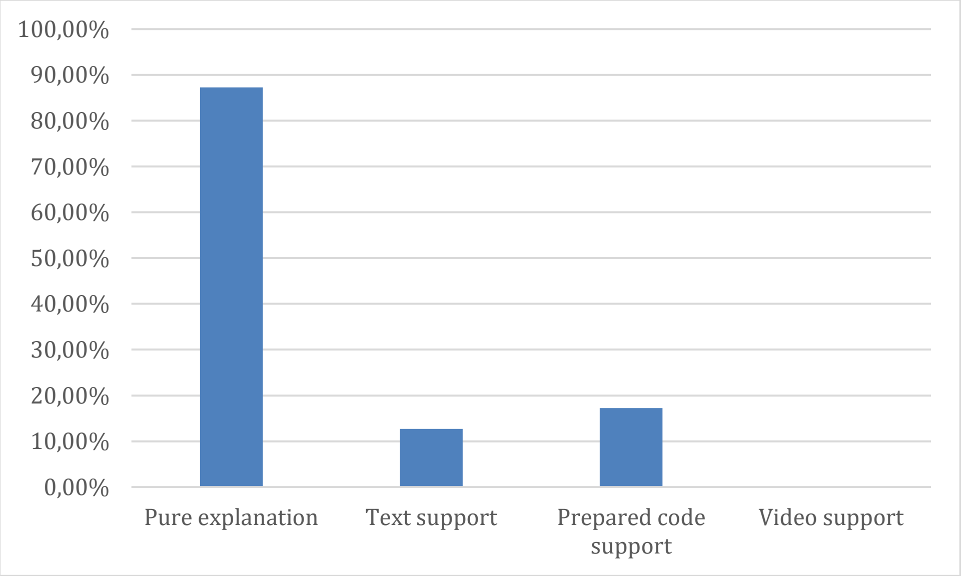

FIGURE 36 - APPLIED INFORMATICS SEMINAR SUBJECT - SUPPORT PERCENTAGE

3.5.2. Comparing of new syllabuses and practices in each country

There are different situations in participating countries. In Croatia and Germany there is a possibility to create optional subject related to project results.

High school Ivanec has curriculum for gymnasia in Croatia. There is one mandatory subject with teaching basics of programming in the first year and in the rest of three years there can be optional informatics subjects. As schools can decide about representation of optional subjects by themselves depending on material and personnel capabilities, there was only optional subject in the second and in the third year before the OOP4FUN project with no OOP content.

The proposed solution was to create new optional subject for the fourth year of study called “OOP through game development”. The proposed solution can be adapted to any school of the same type in Croatia. The aim of created subject is to expand students' programming skills with object-oriented approach.

More complicated possibilities are in Germany due to different rules in each state and each type of school. Gymnasium Dresden Plauen has conditions for grammar school (German "Gymnasium"). But also in this school, there was found a possibility to apply OOP4FUN project results with creating optional subject in learning area "8C - Practical Computer Science - Advanced Programming, which includes the basics of object-oriented programming (inheritance, polymorphism, encapsulation) and their application to the solution of a complex problem. However also in Saxony there are subjects related to programming skills – but without mandatory OOP skills. The aim of innovation is here the same as in Croatia – expanding programming skills with object-oriented approach. Therefore, the project results of OOP4FUN project can be applied for same type of schools in Saxony.

In Serbia there are many teaching hours related to programming – even with OOP skills in mandatory subjects. For that reason, the project OOP4FUN results were only used for improving the syllabus of existing mandatory subjects. Since in Serbia there is big time dotation for OOP there is no problem with implementing project results by using created syllabus and Greenfoot developing environment in OOP teaching hours.

In these three countries benefit of OOP4FUN results is same. Using the Greenfoot developing environment will ensure that student's knowledge is expanded in a playful way and that the OOP principle is understood by large number of students in all countries. The Greenfoot developing environment was used for extension of programming skills with OOP principles.

Schools in the Czech Republic and Slovakia could create their own subjects with their own names and content. The limitation is that it is necessary to comply with the RVP (in Czech Republic) and SEP (in Slovakia).

Within the Czech Republic for the type of school Gymnázium, Pardubice, Dašická 1083 has RVP for secondary education. It includes programming skills in code-based programming language. It is very easy to implement OOP4FUN results. The easiest way is to change content of existing subjects by innovated one. Subject renaming or moving contents of more subjects to one subject is also possible. As was said the only limit it the compatibility with student skills defined by RVP.

In Gymnázium, Pardubice, Dašická 1083 there are optional subjects in the 3rd an 4th year of study. Students can choose them according to their future fields of study. Since programming skills are required in the 2nd year of study, there was made an innovation of subject "Informatics" to show advantages of programming to students and to motivate them to continue in technical education in universities. For technical based students was innovated subject "Seminar of programming".

In subject "Informatics" the project results in Greenfoot environment were used as a bridge between block-based and code-based programming. Students prepare game is block based programming language – Scratch and teacher makes same game in Greenfoot developing environment. Then the teacher will provide the code to the students and explains it. Student can make some changes or can try to make something new. By this way students can learn OOP skills in a playful way. This way can be easily adapted to students with different programming skills and different algorithmic thinking - just adapt what the students have to do to their skills.

For technical students was innovated subject "Seminar of programming". The new concept is based on object first principal. The first part is based on Greenfoot developing environment and game creation where students create new games by themselves.

Object first principal was also used in Slovakia school Obchodná akadémia Považská Bystrica. It was in Greenfoot developing environment in some way even before the start of the project. Project results were used to improve the syllabus. Since the fact, that in Slovakia schools need to create their syllabuses to be compatible with standards defined by SEP, schools are free to choose programming language, developing environment and order of topics, all schools in Slovakia can implement OOP4FUN project results.

The teaching of code-based programming in Czech Republic and Slovakia is started with Greenfoot developing environment. It is the second possible usage of Greenfoot as first developing environment. As it was mentioned, in the Czech Republic, Greenfoot is also used as a bridge between block-based and code-based programming. It is its next possible usage.

Following two graphs shows numbers and percentage of teaching hours related to OOP topics in each country.

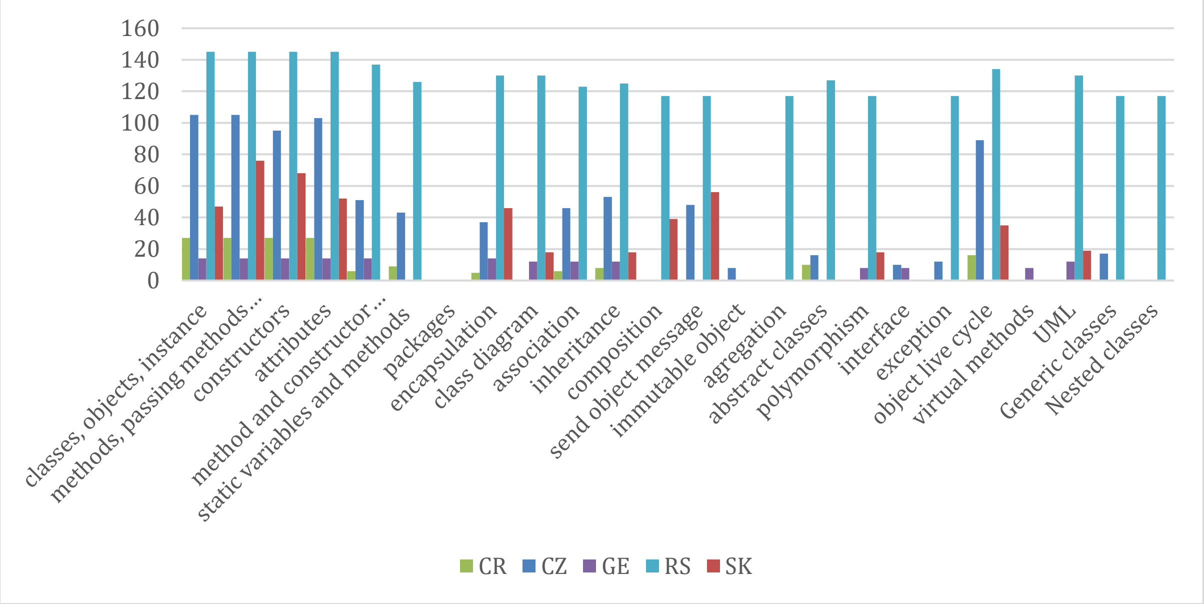

FIGURE 37 - OOP TOPICS - HOURS COMPARE

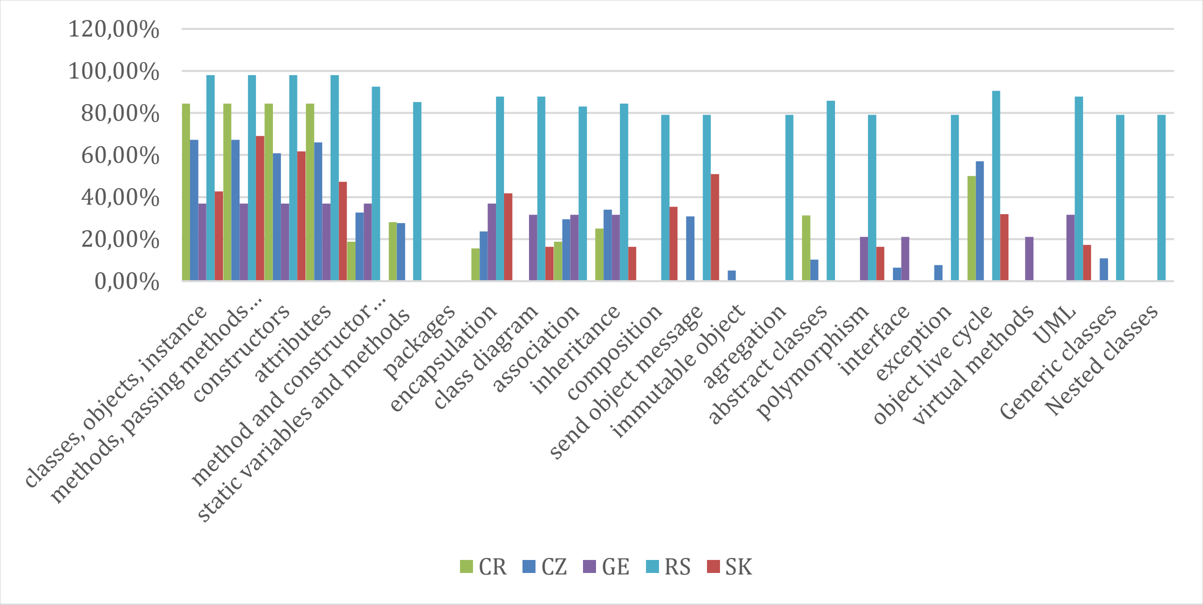

FIGURE 38 - OOP TOPICS - PERCENTAGE OF HOURS COMPARE

As there were selected “light OOP topics” in PR1, let us show same graphs for only “light OOP topics” to compare different practices in each country.

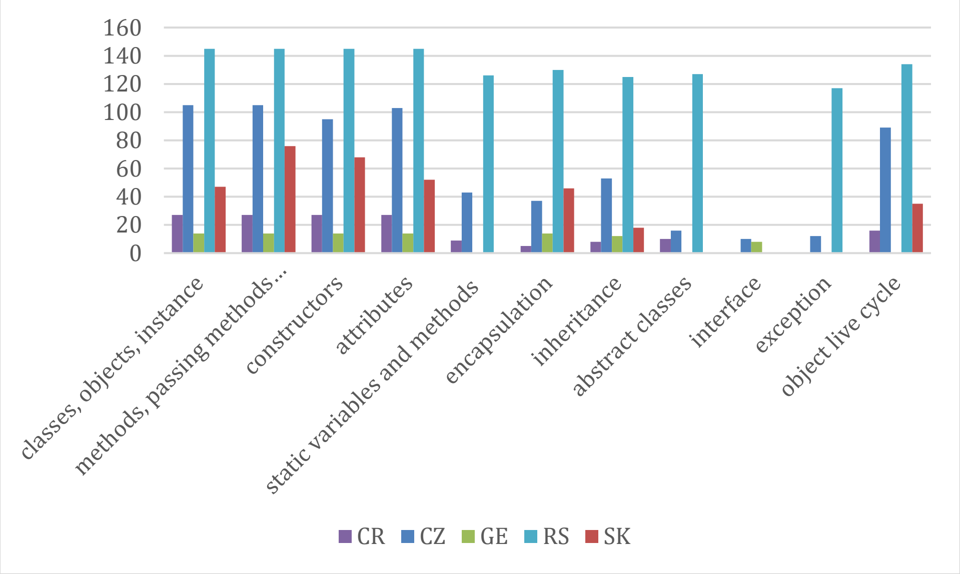

FIGURE 39 - LIGHT OOP TOPICS - HOURS COMPARE

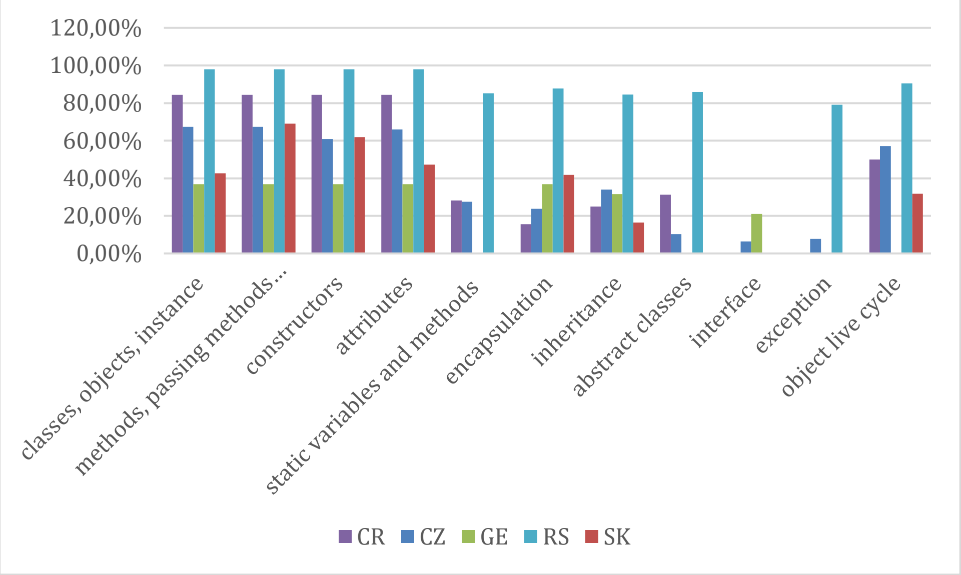

FIGURE 40 - LIGHT OOP TOPICS - PERCENTAGE OF HOURS COMPARE

In PR2 there were explained different learning and teaching activities (LTA). Let us show graphs with number and percentage of hour of using LTA in each country for comparing different practices in each country.

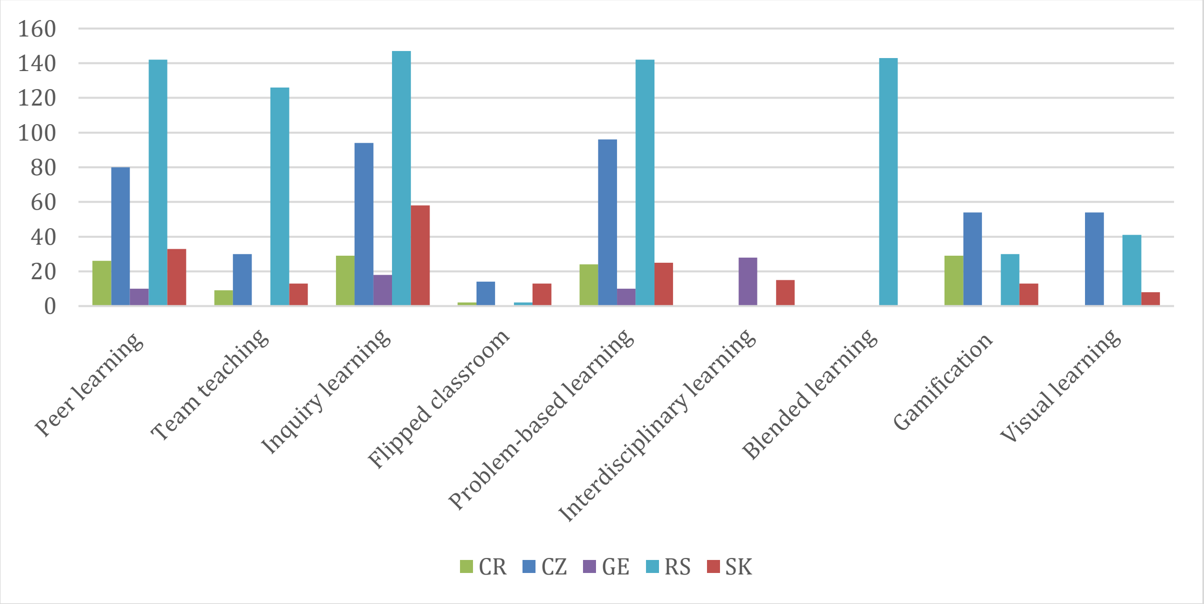

FIGURE 41 - LTA - HOURS COMPARE

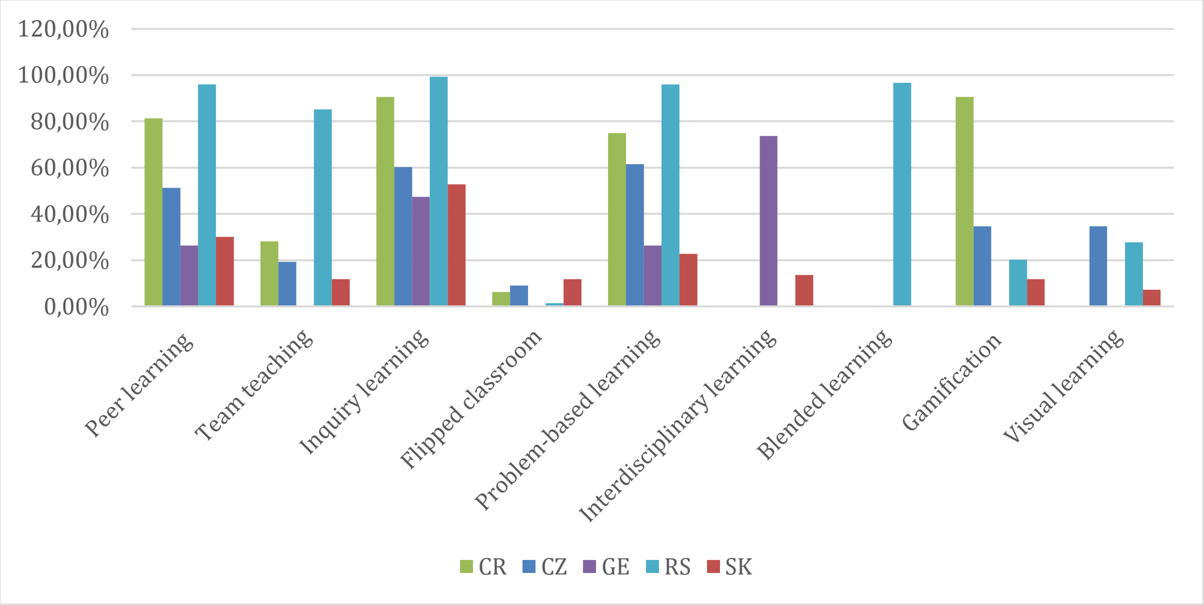

FIGURE 42 - LTA - PERCENTAGE OF HOURS COMPARE

New syllabus requires teaching support. For example materials for students. In PR5 there were defined following types of support - pure explanation with no teaching material, text material support, prepared code support and video support. At the end of this section, let us compare type of support in each country by graphs with number and percentage of hour of using them.

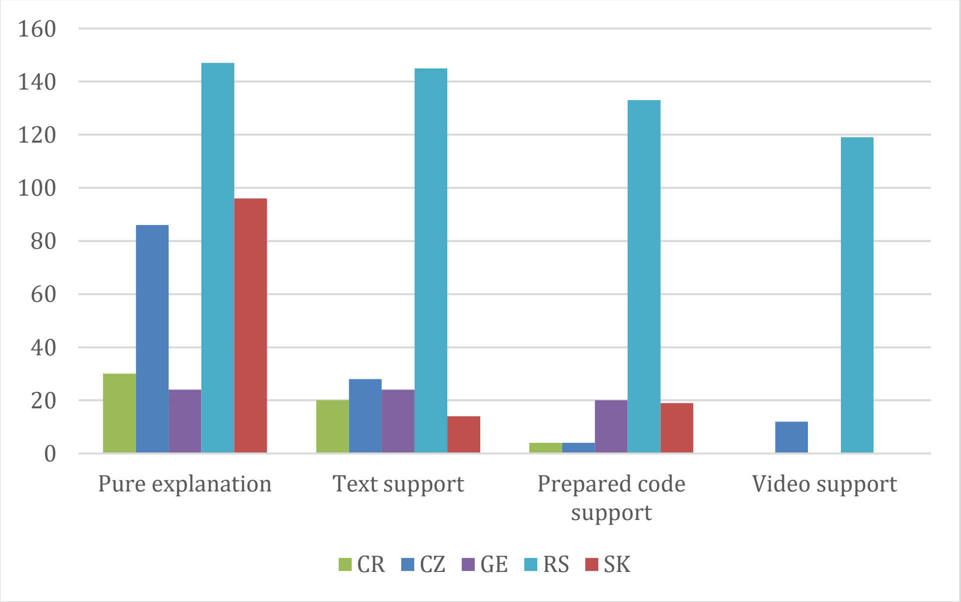

FIGURE 43 - SUPPORT - HOURS COMPARE

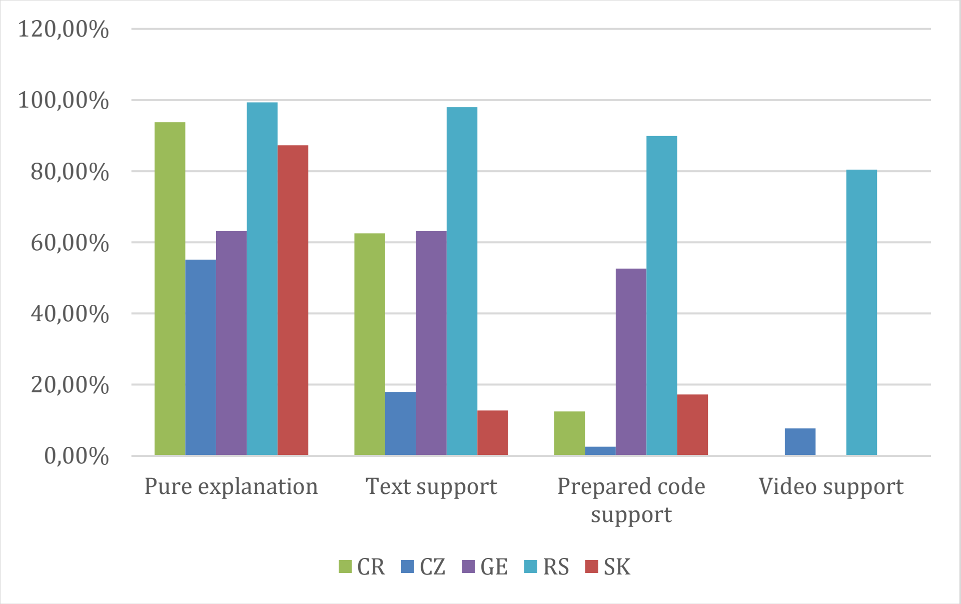

FIGURE 44 - SUPPORT - PERCENTAGE OF HOURS COMPARE

4. Conclusion

The PR5 should describe syllabus and literature accreditation process as well as different practices in each country – LTA, study materials and project results usage. The goal of the project was to show that its results are transferable over different schools in each country and between countries. From text above it was satisfied. Let us show graphs to compare the number of hours of the new syllabus in mandatory and optional subjects in each state and number of hours in Greenfoot developing environment.

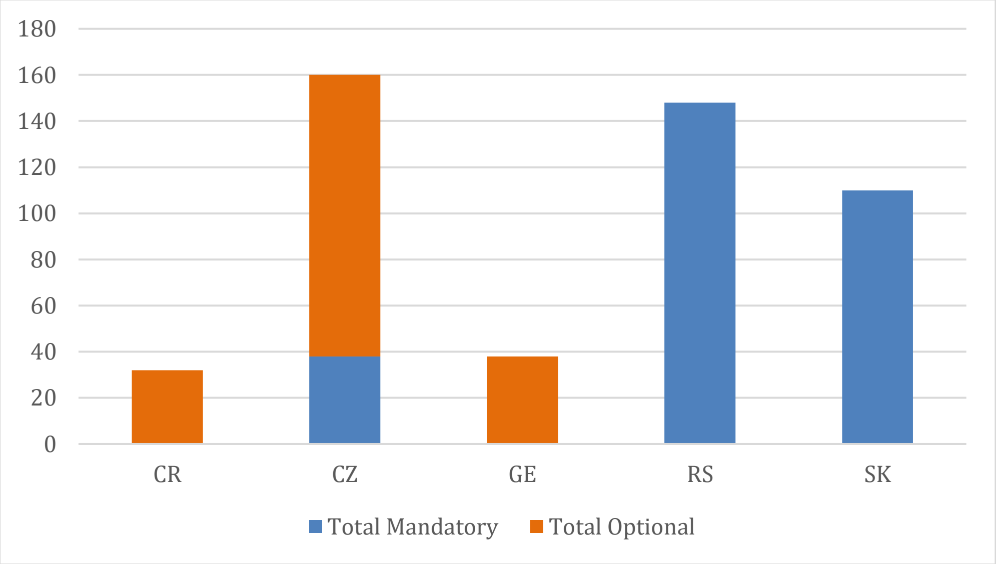

FIGURE 45 - NEW SYLLABUS - NUMBER OF HOURS IN EACH COUNTRY

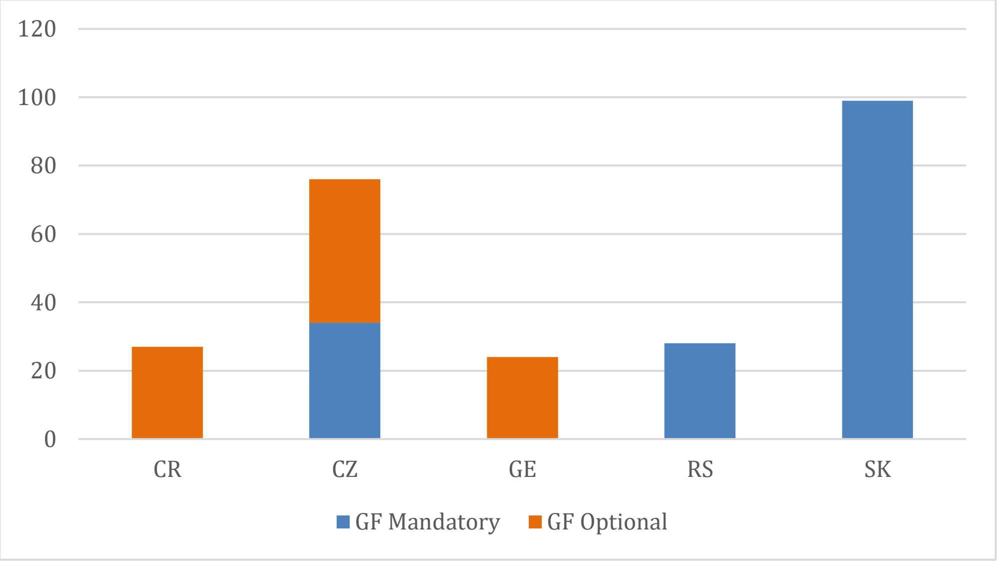

FIGURE 46 - NEW SYLLABUS - NUMBER OF HOURS IN EACH COUNTRY IN GREENFOOT ENVIRONMENT

Last graph shows percentage of teaching hours with Greenfoot developing environment.

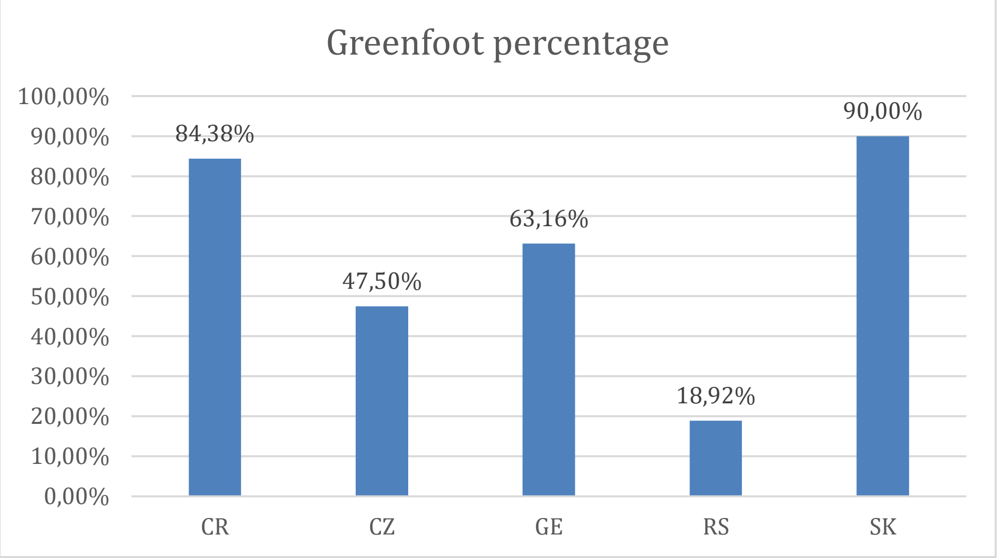

FIGURE 47 - PERCENTAGE OF HOURS WITH USING GREENFOOT ENVIRONMENT IN EACH COUNTRY

Greenfoot and project results were used in different way in each country. In Croatia, Germany and Serbia they were used for extension of programming skills with an object approach. In Slovakia whole teaching of programming was based on Greenfoot and project results. In Czech Republic it was used as a bridge between block-based and code-base programming language and for explaining basics of programming, OOP basics before using different developing environment for more complicated topics.

The project results in PR3 and PR4 with Greenfoot developing environment can be successfully used in international level in following three ways.

1. Extension of programming skills with an object approach
2. Main developing environment in teaching programming with OOP principals
3. Bridge between block-based and code-based programming

This fact shows the potential in the international use of the project results in different countries, different type of schools as well as in different phase of teaching programming skills.

5. List of Tables

6. List of figures