Project co-funded by European Union under Erasmus+ Programme

CodeInnova Curriculum

Authors
Jukka Lehtoranta, Janne Fagerlund, Marika Peltonen, Denis Zhidkikh, Juho Liedes & Mikko Vesisenaho

Publisher
University of Jyväskylä / Erasmus+ project 'CodeInnova - Teaching programming in primary school: curriculum, didactic methods, textbooks, online support' (project number 2019-1-PL01-KA201-065077)

Consultants / Reviewers
Emilia Ahlström
Bogusława Denys
Dulce Freire
Katarzyna Garbacik
Bogusław Klimczuk
Maria José Rodrigues
Ivana Ružić
Vítor Silva
Marko Šolić
Siniša Stričak
Ana Trucek

Design and Illustrations
Jukka Lehtoranta
Martti Minkkinen

Language Translators
Tanja Djaković (Croatian)
Maria José Rodrigues (Portuguese)
Bogusława Denys (Polish)

Erasmus+ project 'CodeInnova'

Teaching programming in primary school: curriculum, didactic methods, textbooks, online support'

(project number 2019-1-PL01-KA201-065077)

CodeInnova curriculum

Curriculum structure

The curriculum is divided into three levels. Additionally, it offers learning goals for planning instruction (under the title “To support teaching”). There are also vocabularies of the most relevant keywords.

General knowledge

General knowledge level aims to provide tools and skills to identify programming as part of a societal impact, to develop multiliteracy skills and to provide resistance to forms of computing as an integral part of lifelong learning.

Computational thinking and programming skills

Programming level offers basic skills for programming (related to core concepts, practises and views of programming) but it also offers computational thinking and abstracting skills in a general level.

Content creation

The level of digital content creation provides topics for programming and introduces the most essential ways of digital content and technologies that are most relevant in the coming decades.

To support teaching

Objectives are provided to support the planning of activities, in particular to support thinking and learning, communication and group work and the teaching of multiliteracy skills. All goals also provide tips on age-appropriate teaching methods. Teachers have a lot of autonomy to use this curriculum, they can vary order of content and modify difficulty level.

Vocabulary

This section provides definitions of relevant words.

Multiliteracy skills Set of thinking and communication skills, which are needed to interpret and produce messages in different situations and environments. Most essential element is an ability to acquire, modify, produce, present, assess and appraise information

Basic concepts of computational thinking

Computational thinking is based on research, there are several principles of computational thinking.. These areas have been taken into account in the design of the curriculum, with reference to the principles of programming thinking at each grade level, with particular emphasis.

These principles are:
Abstraction Programming languages, programs and data are abstractions of the real world phenomenon
Algorithms Number of systematic executable instructions or commands to perform some task
Automation Performing the tasks to automatic programmed instructions using
Collaboration Working together and sharing responsibility
Creativity Creating a project is always a form of creative expression, programming requires finding and using different options.
Data A plurality of different data consisting of sources members and using of it.
Figures and generalization of figures Repetition of the general form solutions that solve similar problems
Iteration The original idea is improved through design, testing and error correction until the ideal situation is achieved.
Logic Logic programs include a variety of logic elements such as conditional statements, Boolean logic and arithmetic operations.
Modeling and Design Programming includes design and algorithmic design models that can be later programmed. Programming involves taking care of the structure, layout and functionality of the system.
Partitioning problems Problems can be divided into smaller and simpler parts that can be solved separately
Performance Algorithms do not include unnecessary or extra steps
Reconciliation and similarity Programs can perform multiple operations at the same time, timing requires control.
Testing and debugging Programmers follow code, design and execute test plans and cases, and isolate and fix problems.

Additional information: *Fagerlund, J., Häkkinen, P., Vesisenaho, M. & Viiri, J.* (2020). Computational thinking in programming with Scratch in Primary schools: systematic review. Computer Applications in Engineering Education, In press. DOI: 10.1002/cae.22255

Grade 1

General knowledge

Pupils understand the meaning of programming related words such as “programming” and “an application”.

Computational thinking and programming skills

Pupils learn step-by-step troubleshooting and the basics of programming in a graphic programming environment by programming people and tangible objects.
Highlighted concepts of computational thinking: *Abstraction, algorithms, iteration, modeling and design, testing and debugging*

Content creation

Pupils become familiar with tangible objects and non-digital programming.

To support teaching

Pupils are offered various opportunities for individual and group work.

From the everyday experience of the pupils, issues related to the topic are emphasized and spoked in appropriate situations.

The development of multi-literacy is supported by a multi-sensory, holistic and phenomenal approach.

Vocabulary

Graphic programming environment Programming is based on visual symbols in graphic programming environment, for example giving moving instructions with arrows
Holistic Comprehensive approach
Non-digital programming Programming via games, symbols, instructions and other non-digital things

Grade 2

General knowledge

Pupils learn to recognize different codes around them and understand the purpose of programming. Furthermore, they understand that each code has its creator and purpose.

Computational thinking and programming skills

Pupils explore, design, and create step-by-step and creative instructions to solve a specific challenge or problem.
Highlighted concepts of computational thinking: *Abstraction, algorithms, creativity, data, iteration, logic, modeling and design, testing and debugging*

Content creation

Pupils introduce programming games and animations in a graphical programming environment. At the same time, they understand the potential of programming as a means of creative expression.
Other content: *tangible objects, non-digital programming*

To support teaching

Pupils gain experience from different programming-related working habits using digital communication tools that support learning, communication and teamwork.

Pupils are encouraged to ask questions, to listen, to make detailed observations, to find information developing ideas that already exist and to come up with new ideas, and to present ideas.

Pupils are guided to develop their multiliteracy skills by enabling them to interpret, produce and develop various age-appropriate codes.

Vocabulary

Graphic programming environment Programming in these environments takes place in different blocks instead of words, for example Scratch junior or Scratch

Grade 3

General knowledge

Pupils learn how human decisions affect the performance of technology.

Computational thinking and programming skills

Pupils solve problems, organize information and learn about concepts of algorithms.

Pupils learn how to visualize problems with different charts and generalizations.

Pupils design, code and develop programs using sequential commands, selections and repetitions.

Highlighted concepts of computational thinking: *Abstraction, algorithms, automation, creativity, data, iteration, logic, modeling and design, partitioning problems, testing and debugging*

Content creation

Pupils become familiar with advanced mobile devices and build and program physical or virtual robots.
Other content: *tangible objects, non-digital programming, simple games and animations*

To support teaching

Pupils gain experience working in a secure group and creating things together.

Pupils are guided to identify the most appropriate ways of learning and to develop their learning and innovation techniques.

Multiliteracy skills are promoted by analyzing different codes from the perspective of the author and the user, taking into account context and situation.

Vocabulary

Algorithm A set of systematically executable instructions for performing a particular task
Command Command to control the device
Repetition Repeating the same thing several times
Selection Taking one or more of the options
Virtual robot Simulator that lets you program robots without a physical device.

Grade 4

General knowledge

Pupils understand the potential of programming for automatic and simultaneous operations.

Computational thinking and programming skills

Pupils design and program in a graphical programming environment using input values.
Pupils introduce simple variables that include numerical values and texts with different kinds of materials.
Pupils can solve more complex logical problems with and without technology.
Step-by-step and conditional instructions and events are utilized in problem solving. Pupils learn to use iterative working habits.

Highlighted concepts of computational thinking: *Abstraction, algorithms, collaboration, creativity, figures and generalization of figures, iteration, logic, modeling and design, partitioning problems, testing and debugging*

Content creation

Pupils learn about the concept of artificial intelligence and different kinds of practical use of it.
Pupils create games for different platforms.
Other content: *tangible objects, non-digital programming, simple games and animations, physical and virtual robots*

To support teaching

Pupils are guided to evaluate and develop their own communication and teamwork skills.
Interactive learning, especially peer learning, is used in many ways, thus reinforcing the pupils' teamwork skills.
Thinking skills are practiced using problem solving and reasoned tasks and working methods that utilize and support curiosity, imagination, inventiveness, and learning.
Critical programming literacy is developed in a cultural context relevant to pupils and near their everyday experiences.

Vocabulary

Artificial intelligence A program capable of doing what is considered intelligent
Input For example, a number entered in a program or pressing a button can also be an auxiliary device connected to the device, eg sensor or button
Iterative Displaying frequently occurring items only once, ie avoiding unnecessary repetition of code
Variable Storage location for programming

Grade 5

General knowledge

Pupils begin to understand how code is always a mathematical problem-solving exercise and how it can possibly lead to ethical issues.

Computational thinking and programming skills

Pupils design and program software that prints values which includes numbers or texts. Pupils check the correctness of the code, and detect and correct errors. Pupils create simple variables. Pupils get introduced to outcomes prediction, testing and explaining existing programs.

Highlighted concepts of computational thinking: *Abstraction, algorithms, automation, collaboration, creativity, figures and generalization of figures, logic, partitioning problems, performance, reconciliation and similarity, testing and debugging*

Content creation

Pupils design and create simple internet of things and wearable technology devices and at the same time learn to express themselves through design and creation.
Other content*: tangible objects, non-digital programming, simple games and animations, physical and virtual robots, artificial intelligence*

To support teaching

Pupils are encouraged to look for ways of expression that are suitable for collaboration.
Pupils are encouraged to use their imagination to find creative solutions.
Pupils learn to analyze different features of games, programs, and applications to distinguish between them.

Vocabulary

Internet of things Various devices connected to the Internet, eg measuring sensors or household appliances
Output Value provided by a program or by functioning accessory connected to a device
Printed output Value that program gives, for example after mathematical operations.
Wearable technology For example smartwatch or intelligent clothing

Grade 6

General knowledge

Pupils are directed to discuss the role of programming as affection. Pupils learn to identify the author values that the code reflects.

Computational thinking and programming skills

Pupils plan, anticipate, monitor, create and adjust programs, which are suitable for age-level. The pupil will be able to code a workable program and solve more complex problems by dividing them into smaller subproblems. Pupils learn more about using different kinds of variables and they create them.

Highlighted concepts of computational thinking: *Abstraction, algorithms, automation, collaboration, creativity, data, figures and generalization of figures, iteration, logic, modeling and design, partitioning problems, performance, reconciliation and similarity, testing and debugging*

Content creation

Pupils design and create a working game or program for a specific purpose.

Other content*: tangible objects, non-digital programming, simple games and animations, physical and virtual robots, artificial intelligence, internet of things (IoT) and wearable design*

To support teaching

Pupils gather experiences to make their own skills available to the group in the best possible way.
Making observations and using a variety of information sources and tools strengthens the pupil's skills in asking questions and finding answers, both independently and with others. Pupils are directed to compare and evaluate the appropriateness of the code used for a particular purpose.

Grade 7

General knowledge

Pupils deepen their understanding of the use of various software and policies.

Computational thinking and programming skills

The pupil will be able to design and create programs that utilize subroutines, appropriate structures and data types, expressions, variables and iterative and conditional commands. General programming languages are used to create programs which are suitable for age-level. The pupil understands the different ways to use simulations and step-by-step organization algorithms to solve problems.

Highlighted concepts of computational thinking: *Abstraction, algorithms, automation, collaboration, creativity, data, figures and generalization of figures, iteration, logic, modeling and design, partitioning problems, performance, reconciliation and similarity, testing and debugging*

Content creation

Pupils create a more complex game, application, or mobile application that solves a particular problem from specific subject or topic. Pupils learn how to outline the operation of a more complex program into various patterns and generalizations.
Other content*: games and animations, physical and virtual robots, artificial intelligence, internet of things (IoT) and wearable design*

To support teaching

Pupils understand different ways to collaborate and different group roles. In addition, they gain experience working in these different group roles.
Pupils are guided to identify and continuously develop their own programming strategies for programming.
Pupils develop their programming-related analytical, critical, and cultural literacy.

Vocabulary

Data type A concept used to define variables that defines a variable to contain, for example a string or integer
General programming language Programming is done by a programming language (for example python or javascript)
Iterative Displaying only one repetition of things, ie avoiding unnecessary repetition in the code
Sorting algorithm An algorithm that arranges the list in a specific order for ease of processing
Subroutine An independent part of a program that performs a specific function

Grade 8

General knowledge

Pupils deepen their understanding of the meaning, potential and risks of programming at a society level. Pupils learn to use artificial intelligence.

Computational thinking and programming skills

Pupils are able to design, create, document, and present programs and robots that solve a particular real-life problem. Created programs include search algorithms, tables and automatic functions. Several simultaneous events happen in these programs.

Highlighted concepts of computational thinking: *Abstraction, algorithms, automation, collaboration, creativity, data, figures and generalization of figures, iteration, logic, modeling and design, partitioning problems, performance, reconciliation and similarity, testing and debugging*

Content creation

Pupils create more complex games, applications or mobile applications that simulate subject matters. Pupils learn about the potential and features of more advanced microcontrollers.
Other content*: games and animations, physical and virtual robots, artificial intelligence, internet of things (IoT) and wearable design*

To support teaching

Pupils know and use different methods for physical and virtual communication, collaboration and collaboration in programming projects.
Pupils are encouraged to express their own experiences and their importance in their own way of thinking and encourage them to listen to themselves and others and see things from the perspective of others.
Pupils are encouraged to use their multi-literacy skills while participating in different situations.

Vocabulary

Microcontroller A system that combines analog electronics components (for example leds) with user-programmed code
Search algorithm An algorithm that organizes a list into a specific order for ease of processing

Grade 9

General knowledge

Pupils have a broad understanding of programming, programs and their role in the modern society. In addition, they understand the importance of programming in influencing and expressing themselves.

Computational thinking and programming skills

Pupils design and implement various automation solutions as well as analyze automation solutions for various hardware and software applications. Pupils will explore opportunities to develop mobile operating systems via practical examples.

Highlighted concepts of computational thinking: *Abstraction, algorithms, automation, creativity, figures and generalization of figures, iteration, logic, partitioning problems, performance, reconciliation and similarity, testing and debugging*
Applied Computational Thinking Principles: 1, 2, 3, 5, 7, 8, 9, 11, 12, 13, 14

Content creation

Pupils become familiar with blockchain technology and its applications, they understand the working principles of solutions that simulate block chains and they get acquainted with simple cryptographic principles. Pupils get familiar with mobile devices and their operating systems working principles.
Other content*: games and animations, physical and virtual robots, artificial intelligence, internet of things (IoT) and wearable design, microcontrollers*

To Support teaching

Pupils will have a good command of the use of various programming aids (for example programming environment) and different working methods.
Pupils can express their own way of thinking and see things smoothly from the perspective of other people.
Pupils have the skills to utilize their own multi-literacy skills and actively develop them independently.

Vocabulary

Blockchain technology Technology that allows unknown entities to maintain distributed databases, for example used in connection with virtual money
Programming environment A program or a set of programs used to code, the environment can translate the code into a program and automatically correct errors in the code (for example keystrokes)
Virtual money Digital virtual money based on an encryption system, all transfers and ownership of which are stored in the money block chain

Sources

Scientific Research

Angeli C, Voogt J, Fluck A, Webb M, Cox M, Malyn-Smith J, Zagami J. A K-6 Computational Thinking Curriculum Framework: Implications for Teacher Knowledge. *Educational Technology & Society.* in 2016; 19 (3): 47-57

Barr V, Stephenson C. Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? ACM Inroads. 2011; 2 (1): 48-54

Czismadia A, Curzon P, Dorling M, Humphreys S, Ng T, Selby C, Woollard J. Computational thinking. A guide for teachers. 2015
https://community.computingatschool.org.uk/resources/2324/single

Fagerlund, J., Häkkinen, P., Vesisenaho, M. & Viiri, J. (2020). Computational thinking in programming with Scratch in Primary schools: systematic review. Computer Applications in Engineering Education, In press. DOI: 10.1002/cae.22255

Grover S, Pea R. Computational thinking: A competency whose time has come. In: Sentance S, Barendsen E, Schulte C, ed. Computer Science Education: Perspectives On Teaching And Learning In School. London: Bloomsbury Academic; 2018: 19-37.

Hsu TC, Chang SC, Hung YT. How to learn and how to teach Computational thinking: Suggestions based on a review of literature. Computers & Education. 2018; 126: 296-310.

Settle A, Perkovic L. Computational Thinking Across the Curriculum: A Conceptual Framework. 2010; Technical Reports, Paper 13
https://www.researchgate.net/publication/254582838_Computational_Thinking_across_the_Curriculum_A_Conceptual_Framework

Shute VJ, Sun C, Asbell-Clarke J. Demystifying Computational Thinking. Educational Research Review. 2017; 22: 142-158 Reports

National Curriculums and summaries

Croatia: https://eacea.ec.europa.eu/national-policies/eurydice/content/croatia_en
Portugal: https://eacea.ec.europa.eu/national-policies/eurydice/content/portugal_en
Poland: https://eacea.ec.europa.eu/national-policies/eurydice/content/poland_en
Finland: https: //eacea.ec.europa.eu/national-policies/eurydice/content/finland_enReports
and https://www.oph.fi/sites/default/files/documents/based_curriculum_country_2014.pdf

Future reports

Adams Becker, S., Cummins, M., Freeman, A., and Rose, K. (2017). 2017 NMC Technology Outlook for Nordic Schools: A Horizon Project Regional Report. Austin, Texas: The New Media Consortium. https://eric.ed.gov/?id=ED593946
Educause horizon report, 2019
https://library.educause.edu/resources/2019/4/2019-horizon-report