Exploring the Computer Science Curriculum

Minisymposium ter ere van proefschriftverdediging SLO'-er Nataša Grgurina

Valentina Dagienė, Vilnius University

Opening passage

- Computer Science (CS) or Informatics or Computing education has a long way in upper secondary school in the Netherlands.
- Many interesting issues can be found in the reformed Dutch CS curriculum.
- I sincerely dedicate this presentation to Nataša Grgurina – to her huge work as a practitioner and as a researcher in the field of Informatics education and her contribution to Computational Thinking (CT) implementation in schools.

Prof. dr. Valentina Dagienė

INTERESTS

teaching and learning of informatics, informatics curricula for schools, olympiads in informatics, gamification, puzzle-based learning, intelligent technologies for education

https://www.mii.lt/en/structure/staff/408-dagiene-valentina-en-gb

MAIN RESULTS

- over 300 research articles
- over 60 textbooks for schools
- editor of two international journals: „Informatics in Education" (WOS emerging list) and „Olympiads in Informatics" (Scopus)
- established *Bebras* Challenge on Informatics and Computational Thinking in 2004, network of 66 countries

www.bebras.org

Terminology

STILL CONFUSING …

From: Shut down or restart?
The Royal Society, January 2012

- **Computing (Education)**

The broad subject area; roughly equivalent to what is called ICT in schools and IT in industry, as the term is generally used.

- **Computer Science**

The rigorous academic discipline, encompassing programming languages, data structures, algorithms, etc.

- **Information Technology**

The use of computers, in industry, commerce, the arts and elsewhere, including aspects of IT systems architecture, human factors, etc.

- **ICT**

Information and Communication Technology

- **Digital literacy**

The general ability to use computers.

M. Webb at al. Computer science in K-12 school curricula of the 2lst century: Why, what and when?

- **Informatics** – the entire set of scientific concepts that make information technology possible
  (Informatics education: Europe cannot afford to miss the boat..., 2013)
- **Computing** is the broad subject area incorporating IT, CS, digital literacy and problem solving in this context deploying CT. *Computing* is the title for UK curriculum. In Australia and N. Zealand *Digital Technologies* is the equivalent term used in curricula.
- **Programming** – a process involving: analysis and understanding of problems; identifying and evaluating possible solutions; generating algorithms; implementing solutions in the code of a particular programming language; testing and debugging, in order to formulate solutions into executable computer programs.

Regarding education in schools

- **Digital literacy** – office, presentation, media, communication, etc.
- **Applications** – using the potential of computers and the internet for the teaching and learning of other disciplines
- **Computer Science** / **Informatics** / **Computing** – the scientific discipline
- **Computational thinking** – recognizing aspects of computation in the world that surrounds us, and applying tools and techniques from Computer Science to understand, reason and solve problems in relation to both natural and artificial systems and processes
  (adapted from The Royal Society 2012)
- M. Webb et al. paper: **Computer science in K-12 school curricula of the 2lst century: Why, what and when?**
  (*Education and Information Technologies*, 2017, 22(2), 445–468)

Computational Thinking

Understanding of Computational Thinking

- Despite an intensive research history of around fifteen years in CT field, researchers have ***not reached a consensus on the definition of CT***. That is why researchers elaborate on various CT definitions depending on their investigations and hold different perspectives when applying, interpreting, and assessing CT concepts
- **CT can be understood as a way of thinking** used to develop solutions in a form that ultimately **allows executing those solutions**. An executor - a *computational agent* (i.e., computer or robot) should be guaranteed to achieve results by doing them automatically
- CT is defined as a kind of thinking skills that could be *transferred and applied in the process of solving real-world* and significant problems (Román-González & Robles, 2019)
- CT is a way of thinking (thought process) for problem-solving (Grover & Pea, 2018; Hazzan et al., 2020; Zhang & Nouri, 2019)
- However, they all specify that it is not just problem solving: the solution of the problem must be expressed in a way that *allows a computational agent to carry it out*.

Categories of CT definitions

- Main categories of CT definitions (summary from the literature):

1. **Generic definitions** that focus on CT as a thought process - as it was initially proposed;
2. **Operational or model definitions** that describe what CT entails based on fundamental elements;
3. **Educational and curricular definitions** derived from different frameworks:

- CT involves systematically approaching problem-solving
- CT as a problem-solving method that used various techniques and strategies
- CT is used for developing knowledge and understanding of concepts in CS
- Thinking computationally means being able to approach and solve problems efficiently based on the principles and methods of Computer Science/Computing.

Definitions of Computational Thinking

Categories of CT definitions	Examples of CT definitions in the analysed literature
Generic definitions	CT is the thought processes involved in formulating a problem and expressing its solution(s) so that a computer-human or machine – can effectively carry it out (Grover & Pea, 2018); CT is about thinking processes, and its implementation is independent of technology (Hazzan et al., 2020); CT is defined as a thought process, ***through skills that are fundamental in programming*** (CT skills), to solve problems regardless of discipline (Zhang & Nouri, 2019) ;
Operational or model definitions	The framework of CT involves solving problems, designing systems, and understanding human behaviour by drawing on the concepts fundamental to computer science (Jocius et al., 2020) ; CT is a set of broadly applicable problem-solving skills, including abstraction, decomposition, pattern recognition, and algorithmic thinking, among others (Huang & Looi, 2020) ; 8 aspects of CT as its core: ***Abstraction, Algorithm Design, Evaluation, Generalization, Iterative Improvement, Information Representation, Precise Communication, and Problem Decomposition*** (Komm et al., 2020); CT definition classified into four major categories: ***data practices, modelling & simulation practices, computational problem-solving practices, and systems thinking practice***s (Weintrop et al., 2016) ;
Educational and curricular definitions	CT involves systematically approaching problem-solving (e.g., algorithmically) in a manner that results in solutions that can be reusable in different contexts (Shute et al., 2017); ACARA defines CT as a problem-solving method that involves various ***techniques and strategies*** that can be implemented by digital systems (Australian Computing Academy, 2019); CT has been recognised for developing knowledge and understanding of concepts in CS as well as for significant contribution to general-purpose problem-solving skills (Israel-Fishelson & Hershkovitz, 2020) ; Thinking computationally means being able to ***approach and solve problems efficiently*** based on the principles and methods of CS (Arfé et al., 2020) .

Computational Thinking

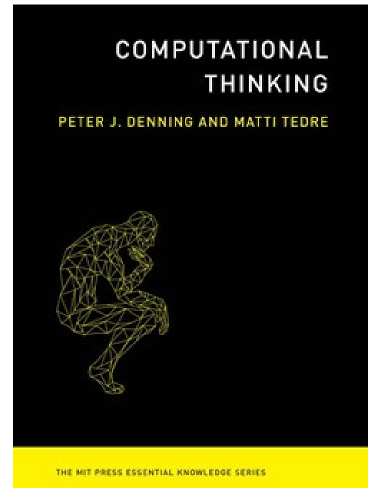
- **Programming and CT are deeply intertwined**. A dual association between CT and programming is noticed. Programming supports the development of CT while CT provides to programming a new updated role (Metcalf et al., 2021; Tikva & Tamboris, 2021)
- The distinction between CT and programming is subtle in principle. CT does not require programming at all, although in practice, representing a solution to a problem as a program provides a perfect way to evaluate the solution, as the computer will execute the instructions to the letter, forcing the student to refine their solution so that it is very precise (Webb et al., 2017)
- **CT is not necessarily about programming but rather, the emphasis is on problem solving** which promote learning experiences (Hazzan et al., 2020).

Concepts of Computational Thinking

CT concepts related to generic problem solving	CT concepts related to program ming and computing
Abstraction Data Analysis Data Collection Data Representation Decomposition Efficiency Evaluation Generalisation Logics and Logical Thinking Modelling Patterns & Pattern Recognition Repeating Patterns Simulation System Thinking Visualisation	Algorithmic Thinking Algorithm Design Automation Boolean Logic Computation Computational Modelling Conditionals Data Types Events Functions Iteration Loops (repetition) Modularisation Parallelisation Sequencing Testing and Debugging Threads (Parallel Execution)

Understanding of CT and related concepts #9

The definitions of CT that have emerged from the post-2006 CT movement have moved conspicuously into the public view. But many public definitions, especially as interpreted to us by policymakers, are quite narrow compared to the notions of CT developed over the earlier centuries of computing. Mainstream media occasionally give a misinformed view of the scope and influence of computing. They have led people unfamiliar with computing to make inflated claims about the power of CT that will mislead students and others into making promises about computers they cannot deliver.



P J Denning & M Tedre. Computational Thinking, The MIT press, 2019.

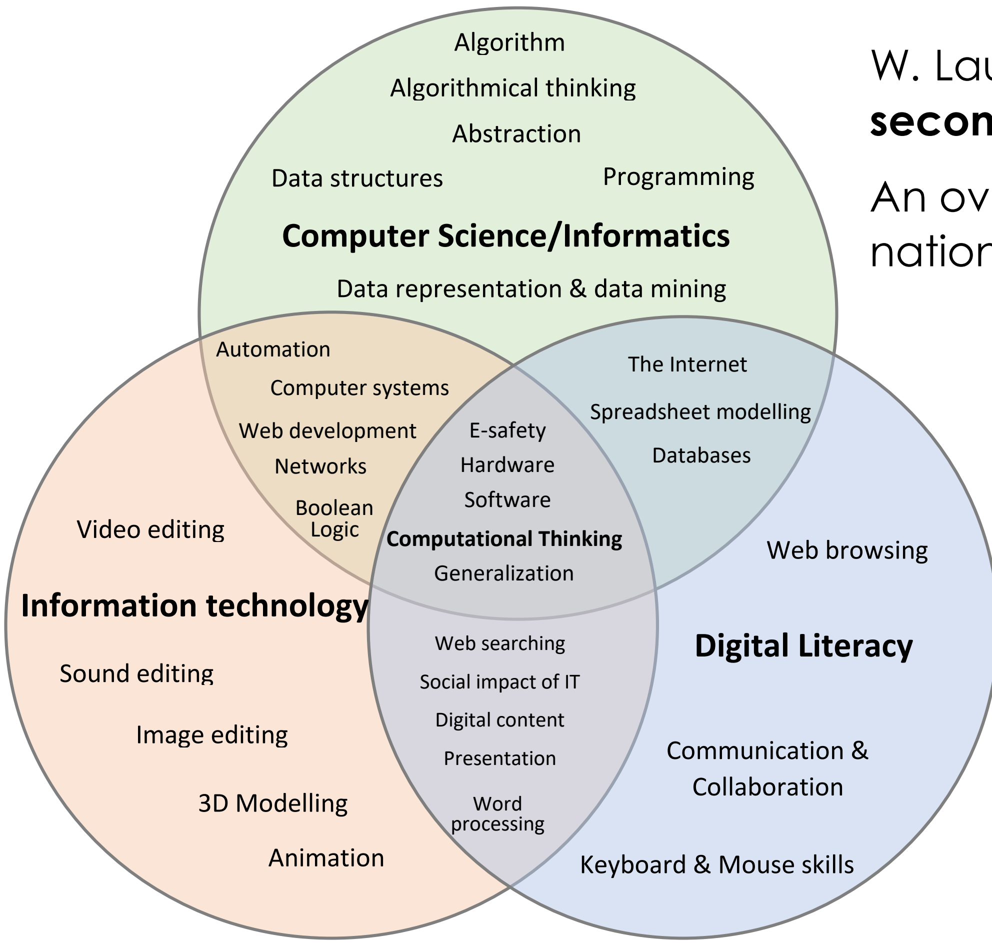
CT compromises the mental skills and practices for

- ***designing*** computations that get computers to do jobs for us, and
- ***explaining*** and interpreting the world as a complex of information processes.

The development of CT has opened Computing Education!

Informatics curriculum

Constructs of Informatics curricula



W. Lau. **Teaching computing in secondary schools**, Routledge, 2018.

An overview of the Informatics national curricula - adapted

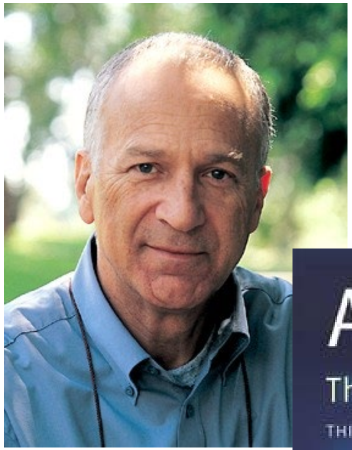
The heart of the scientific discipline

Algorithmical Thinking

- State the problem clearly and unambiguously;
- Write an algorithmic solution that:
  - takes into account all boundary conditions (robustness);
  - determines that the algorithm produces eventually the right answer;
  - tests that the solution is efficient (complexity considerations).
- Translate the algorithm into a programming language;
- Run it on a computer.

Computational Thinking

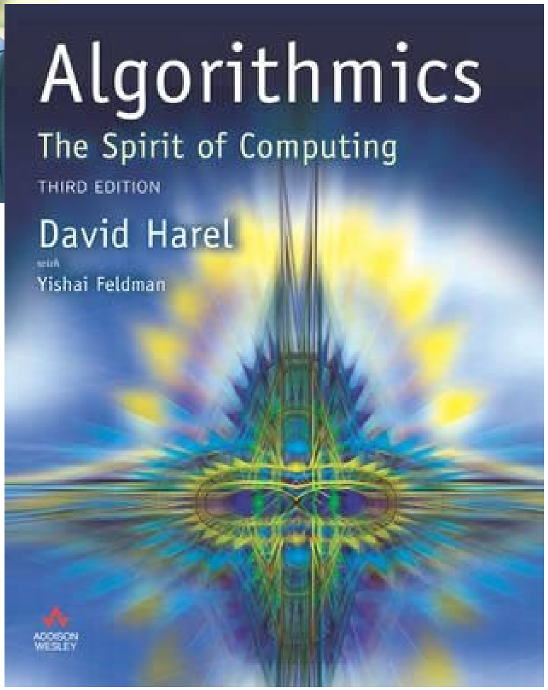
The Spirit of Computing



David Harel. Algorithmics:
The Spirit of Computing

Three complexities:

- Computational complexity,
- Behavioral complexity,
- Cognitive complexity.



Informatics Europe

Informatics education: Europe cannot afford to miss the boat

Report of the joint Informatics Europe & ACM Europe Working Group on Informatics Education, April 2013

Informatics covers the science behind information technology.
Informatics is a **distinct science**, characterized by its own concepts, methods, body of knowledge and open issues.
It has emerged, in a role similar to that of mathematics, as a cross-discipline field underlying today's scientific, engineering and economic progress.

- CS teaches **problem solving**,
- CS supports and links to other sciences,
- CS is **intellectually important**,
- CS leads to multiple career paths.

Role of CT in Curriculum

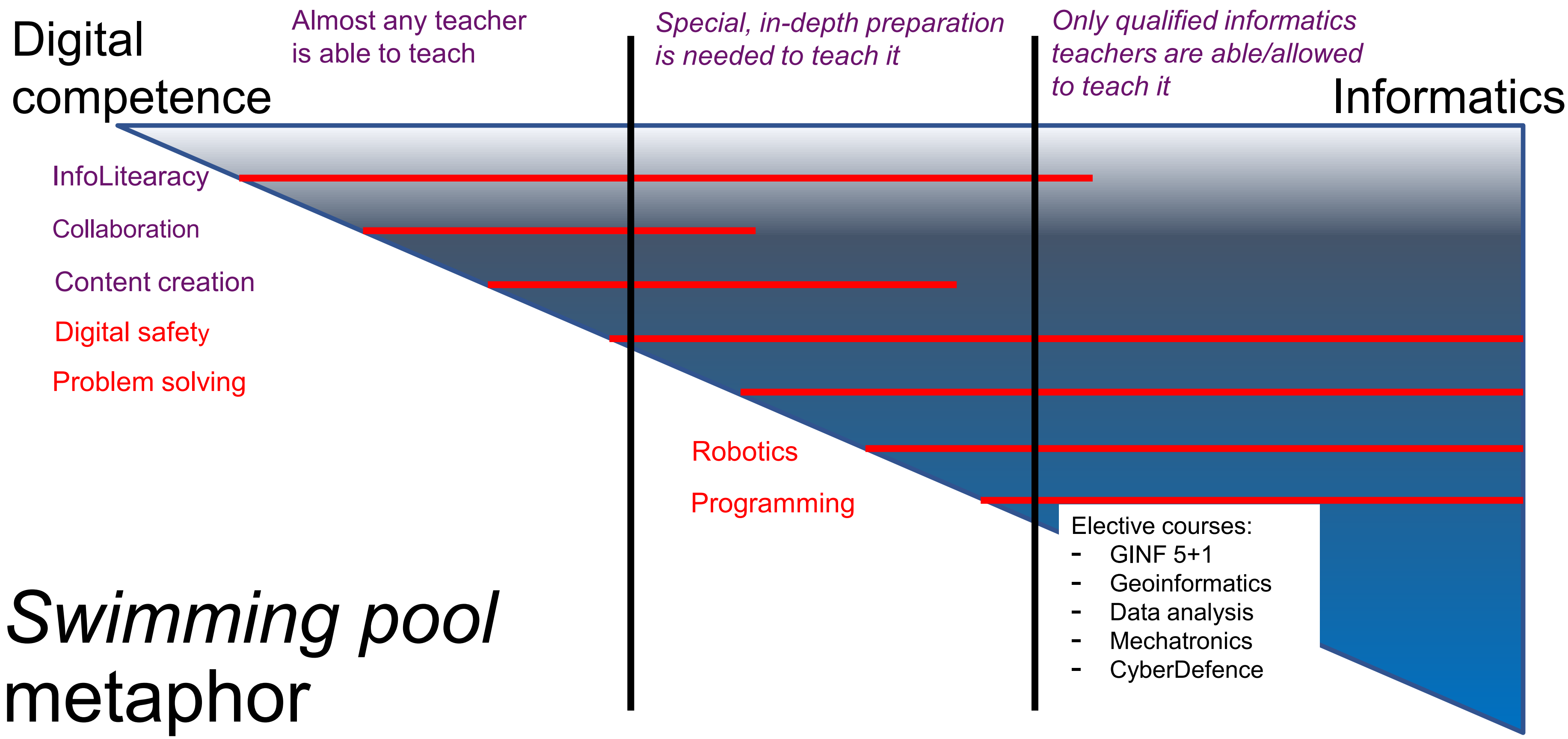
- **Make use of CT to generate innovation from pre-primary to secondary education level**, establishing a continuum-based approach to CT development and renewing practices in all education levels
- **Weave CT (as part of Informatics and digital skills) into all subjects** to foster students' problem solving and logical thinking skills
- **Implement CT to support gender balance and foster students' creativity**, paving the way to strengthen women's employment in the IT sector and making room in the curriculum for all pupils' self expression and creativity

CT implementation in curriculum

- **Make room for a minimum of one lesson per week for CT implementation**, facilitating (e.g.) the development of students' fluency in programming
- **Re-arrange / increase the number of hours of Maths and related subjects beyond the minimum requirements** to make adequate room for implementing CT
- **Strengthen synergies between subjects where CT is integrated** to make the acquisition of CT competence more effective
- **Use comprehensive approaches for assessment of CT skills,** defining tools and criteria to help teachers in formative and summative assessment
- **Include CT in summative assessment at the end of the lower secondary cycle,** thereby underlining its significance

CSTA K-12 standards

- The curriculum should prepare students to understand the nature of CS and its place in the modern world.
- Students should understand that CS interweaves **concepts and skills**.
- Students should be **able to use CS skills (especially CT)** in their problem-solving activities in other subjects.
- The CS standards should complement Information Technology and Advanced Placement CS curricula.

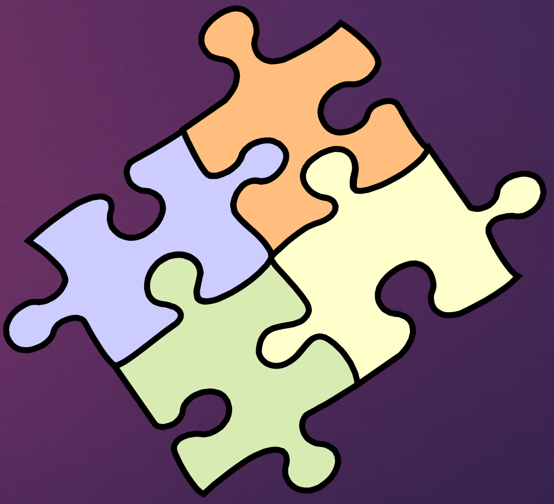

Digital competence
Almost any teacher is able to teach
Special, in-depth preparation is needed to teach it
Only qualified informatics teachers are able/allowed to teach it
Informatics
InfoLitearacy
Collaboration
Content creation
Digital safety
Problem solving
Robotics
Programming
Elective courses:
- GINF 5+1
- Geoinformatics
- Data analysis
- Mechatronics
- CyberDefence
Swimming pool metaphor

Should computing be taught earlier than middle school?

- Every junior (middle) school student should be exposed o a broad introductory CS course;
- Following the broad junior school course, students should be able to decide whether to take the discipline in high school;
- High school will serve as pipe line to higher education.

- Concerning primary school CS education, a deep and thorough research is needed:
  - How early can students – children handle abstraction?
  - If computing is taught working only with concrete objects – technical coding only – what is the effect of such and introduction to the image of computing?
  - Is this image fixed? Could it be changed at an older age?

Dutch Informatics curriculum



First decade on Informatics in Dutch high schools

- Grgurina, N., & Tolboom, J. (2008). **The First Decade of Informatics in Dutch High Schools.** *Informatics in Education*, *7*(1), 55–74.
- Barendsen, E., & Zwaneveld, B. (2010). Informatica in het Voortgezet Onderwijs Voorstel voor vakvernieuwing [Notitie voor KNAW].
- van DIEPEN, N., PERRENET, J., ZWANEVELD, B. (2011). Which Way with Informatics in High Schools in the Netherlands? The Dutch Dilemma. *Informatics in Education* 10(1), 123-148.
- Grgurina, N. (2013). Computational Thinking in Dutch Secondary Education. Informatics in Schools: Local Proceedings of the 6th International Conference ISSEP 2013–Selected Papers, 119.
- Tolboom, J., Kruger, J., & Grgurina, N. (2014). Informatica in de bovenbouw havo/vwo: Naar aantrekkelijk en actueel onderwijs in informatica. SLO.

CT and modelling

- Grgurina, N., Barendsen, E., Zwaneveld, B., Veen, K. van, & Stoker, I. (2014). Computational Thinking Skills in Dutch Secondary Education: Exploring Pedagogical Content Knowledge. Proc of the 14th Koli Calling International Conference on Computing Education Research, 173–174.
- Grgurina, N., Barendsen, E., Zwaneveld, B., Veen, K. van, & Stoker, I. (2014b). Computational Thinking Skills in Dutch Secondary Education: Exploring Teacher's Perspective. Proc of the 9th Workshop in Primary and Secondary Computing Education, 124–125.
- Grgurina, N., Barendsen, E., Veen, K. van, Suhre, C., & Zwaneveld, B. (2015). Exploring Students' Computational Thinking Skills in Modeling and Simulation Projects: A Pilot Study. Proc of the Workshop in Primary and Secondary Computing Education, 65–68.
- Barendsen, E., Mannila, L., Demo, B., Grgurina, N., Izu, C., Mirolo, C., Sentance, S., Settle, A., & Stupurienė, G. (2015). Concepts in K-9 Computer Science Education. Proc. of the 2015 ITiCSE on WG Reports, 85–116.

Second decade on Informatics in Dutch high schools

- Grgurina, N., Barendsen, E., Zwaneveld, B., Veen, K. van, & Suhre, C. (2016). Defining and Observing Modeling and Simulation in Informatics. International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, 130–141.
- Barendsen, E., & Tolboom, J. (2016). Advisory report (intended) curriculum for informatics for upper secondary education. SLO.
- Barendsen, E., Grgurina, N., & Tolboom, J. (2016). A New Informatics Curriculum for Secondary Education in The Netherlands. International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, 105–117.
- Grgurina, N., Tolboom, J., & Barendsen, E. (2018). **The Second Decade of Informatics in Dutch Secondary Education**. 271–282.
- Grgurina, N., van der Veen, R., & Velthuizen, V. (2019). Informatica 2019, Domein R: Computational Science. Lesmateriaal. https://ieni.github.io/inf2019/themas/r-computational-science

2019 Dutch Secondary CS Curriculum

- **A. SKILLS**
  - A1. Using information skills
  - A2. Communicating
  - A3. Reflecting on learning
  - A4. Orientation on study and profession
  - A5. Researching
  - A6. Modelling
  - A7. Appreciating and deciding
  - A8. Designing and developing
  - A9. CS as a perspective
  - A10. Cooperation and interdisciplinarity
  - A11. Ethical conduct
  - A12. Using CS tool set
  - A13. Working in contexts
- **B. BASICS**
  - B1. Algorithms
  - B2. Data structures
  - B3. Machines
  - B4. Grammars
- **C. INFORMATION**
  - C1. Objectives
  - C2. Identifying
  - C3. Representing
  - C4. Standard representations
  - C5. Structured data

2019 Dutch Secondary CS Curriculum

- **D. PROGRAMMING**
  - D1. Developing
  - D2. Inspecting and adapting
- **E. ARCHITECTURE**
  - E1. Decomposition
  - E2. Security
- **F. INTERACTION**
  - F1. Usability
  - F2. Social aspects
  - F3. Privacy
  - F4. Security

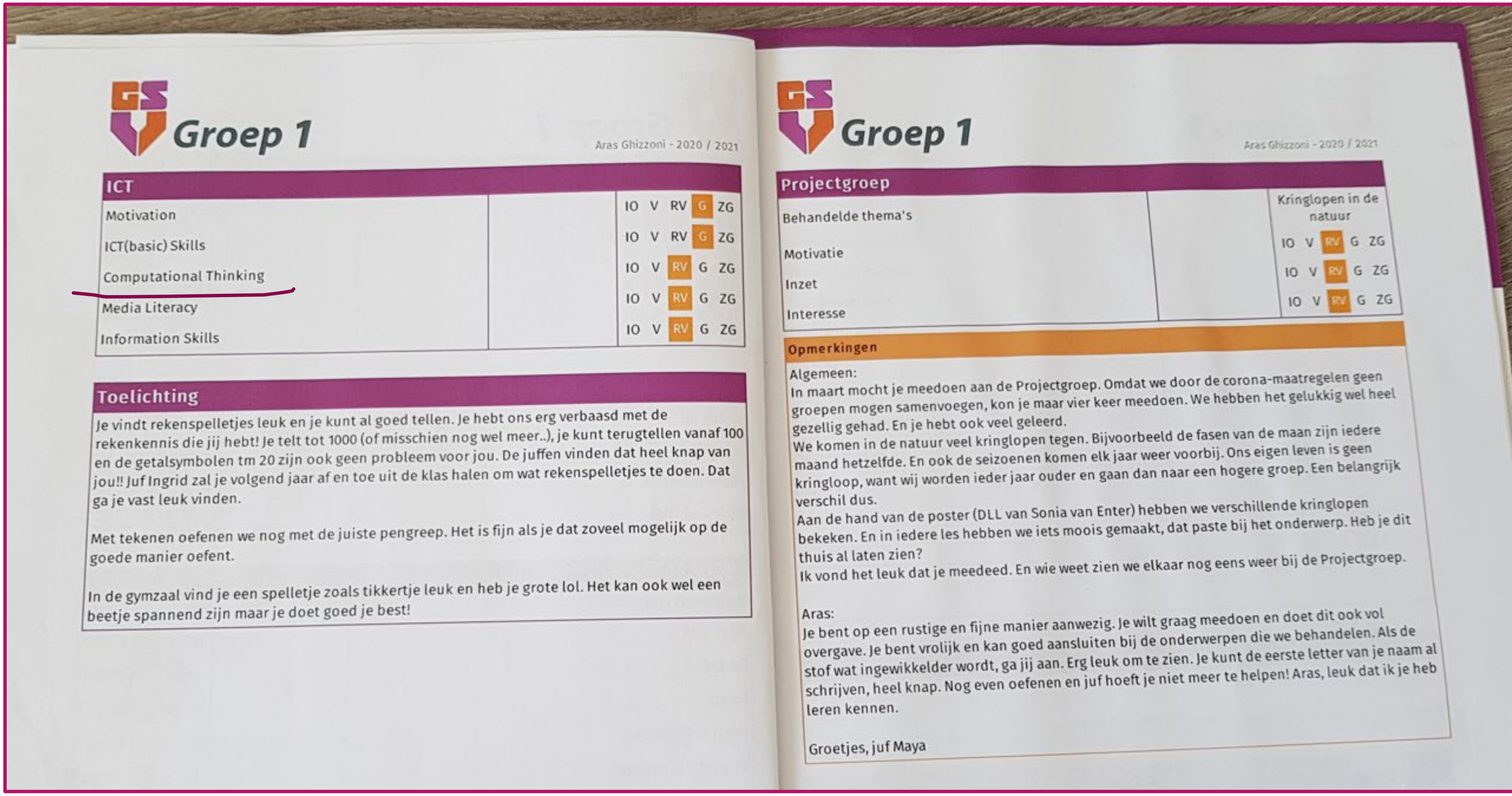
2019 Dutch CS Curriculum - electives

- **G. Algorithms, computability and logic**
  - G1. Algorithm complexity
  - G2. Calculability
  - G3. Logic
- **H. Databases**
  - H1. Information modelling
  - H2. Database paradigms
  - H3. Linked data
- **I. Cognitive computing**
  - I1. Intelligent behavior
  - I2. Cognitive computing
  - I3. Application of cognitive computing
- **J. Programming paradigms**
  - J1. Alternative programming paradigm
  - J2. Selecting a programming paradigm
- **K. Computer architecture**
  - K1. Boolean algebra
  - K2. Digital circuits
  - K3. Machine language
  - K4. Variation in computer architecture
- **L. Networks**
  - L1. Network communication
  - L2. Internet
  - L3. Distribution
  - L4. Network security

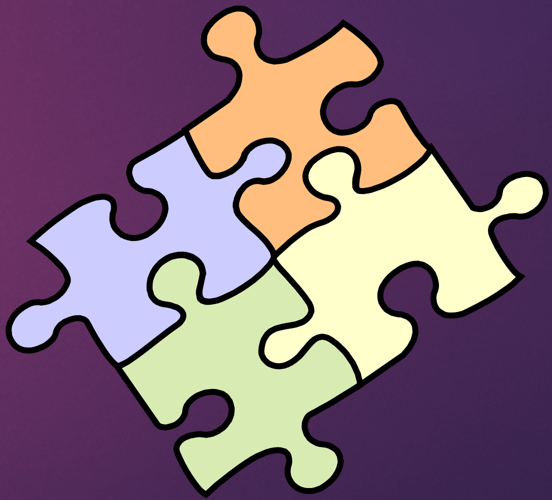
2019 Dutch CS Curriculum - electives

- **M. Physical computing**
  - M1. Sensors and actuators
  - M2. Development of physical computing components
- **N. Security**
  - N1. Risk analyses
  - N2. Measures
- **O. Usability**
  - O1. User interfaces
  - O2. User research
  - O3. Design
- **P. User Experience**
  - P1. Analysis
  - P2. Design
- **Q. Social and individual** impact of informatics
  - Q1. Social influence
  - Q2. Legal aspects
  - Q3. Privacy
  - Q4. Culture
- **R. Computational Science**
  - R1. Modelling
  - R2. Simulating

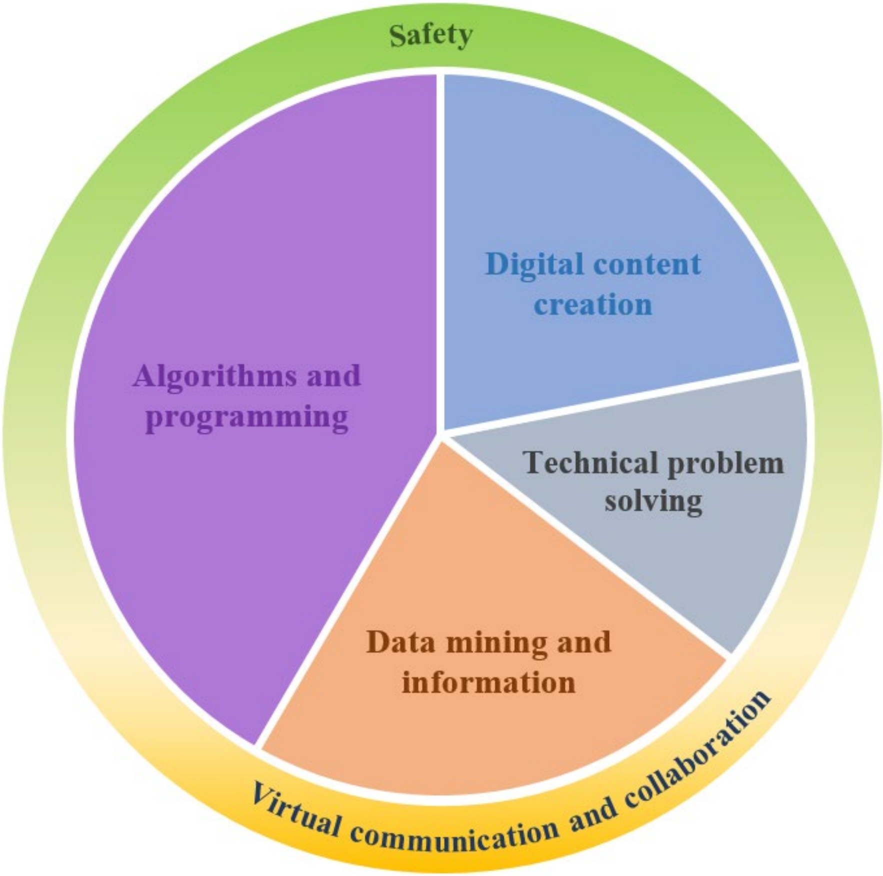
CT in one of Dutch school: Grade 1



Lithuanian Informatics curriculum



Informatics curriculum: Areas of achievement



Reformed Informatics curriculum in schools in Lithuania:

- Integrated in grades 1-4
- Mandatory in grades 5-10 (approx. 1 h per week)
- Selected in grades 11-12

Six areas of Informatics achievements (aligned to DigComp)

Informatics curriculum in Lithuania

Algorithms & programming	• Understand the benefits of an algorithm, a program, recognize and use informatics concepts • Apply programming commands, logical operations and call programming interfaces (APIs) • Create (code) and execute programs • Detect bugs, test and improve programs. Use IDEs
Data mining & information	• Understand the importance of data and information, make data analyses • Perform various actions with data: collect, store, group, search, visualize • Evaluate relevance and reliability of information
Digital content creation	• Know various digital content for learning, recognize concepts • Create various digital content: draw, write, compose, record, film, create mind maps, tables, diagrams • Evaluate and improve, shares digital content

Informatics curriculum in Lithuania

Technical problem solving	● Investigate hardware, see problems arising from the use of digital technologies, use properly technical concepts ● Select and combine various digital technologies ● Self-educate and self-evaluate own digital skills
Virtual communication & collaboration	● Collaborate, share experiences and resources, communicate using digital technologies ● Assess the dangers of virtual communication, protects software and hardware
Safety	● Protect health and environment ● Behave safely in virtual space

KURIU IR MĄSTAU. Integruotos informatikos užduotys – smagių uždavinių rinkinys, padedantis žingsnelis po žingsnelio ugdyti informatinio mąstymo įgūdžius, reikalingus būsimiems kompiuterių, robotų, naujų išmaniųjų įrenginių kūrėjams.

Pagal projekto „Informatika pradiniame ugdyme" gaires parengtas leidinys yra susijęs su integruoto ugdymo vadovėliu „Vaivorykštė", todėl jame esančias užduotis galima įtraukti į pasaulio pažinimo, matematikos, gimtosios kalbos ar kitų mokomųjų dalykų turinį.

Leidinį papildo metodiškai paaiškinti užduočių sprendimai ir atsakymai, kuriuos galima rasti skaitmeninėje ugdymo priemonėje e.lankos (www.e-lankos.lt) arba leidyklos svetainėje (baltulankuvadoveliai.lt/kuriu-ir-mastau/).

Pradinėms klasėms skirtas
integruoto ugdymo vadovėlis „Vaivorykštė"

www.vaivorykste.eu

9 786094 701214
www.baltulankuvadoveliai.lt

Valentina Dagienė

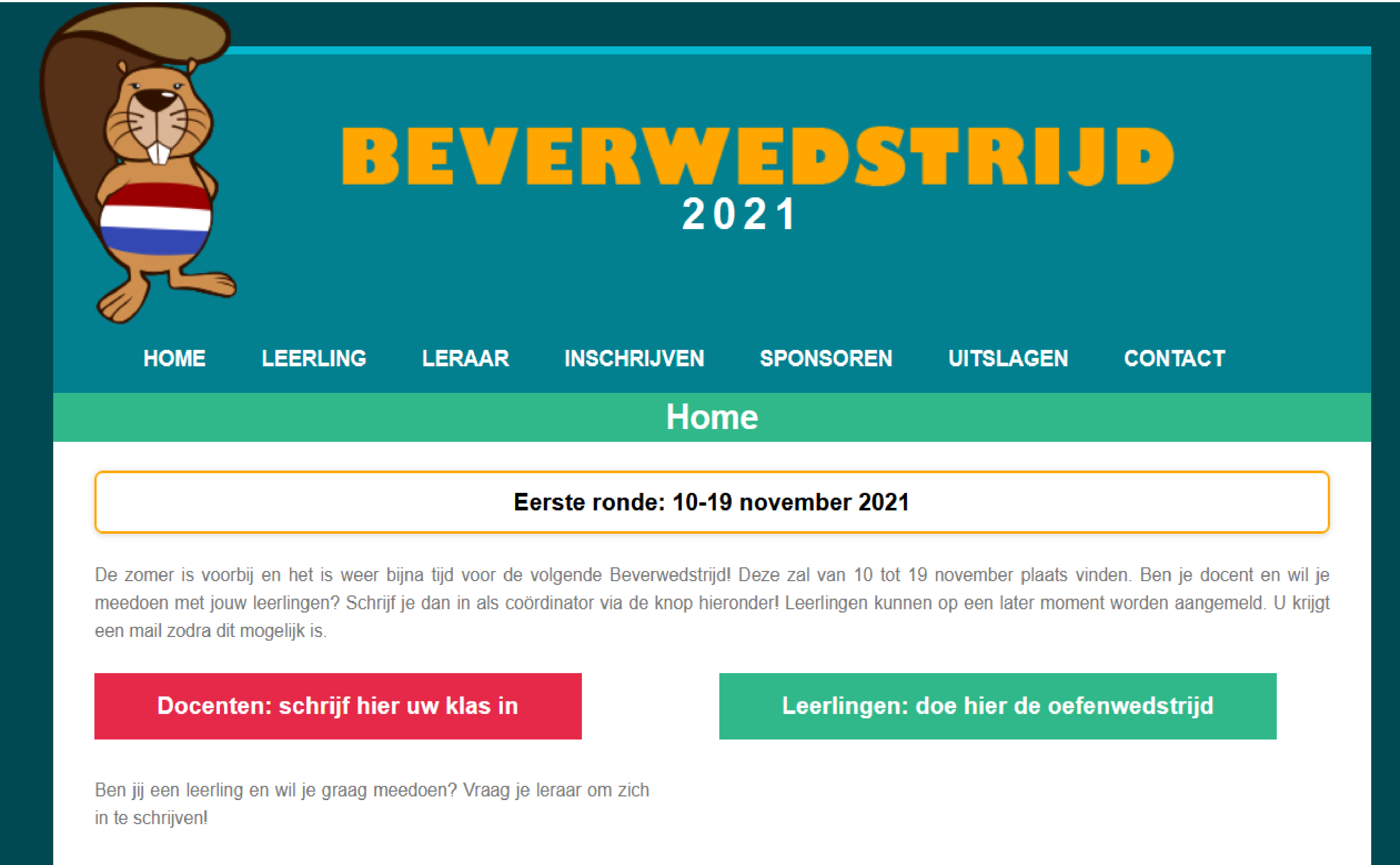
MĄSTAU IR KURIU

Integruotos informatikos užduotys 6–8 metų vaikams

baltos lankos

Informatics exercise books for primary grades

https://www.beverwedstrijd.nl



Thank You!

valentina.dagiene@mif.vu.lt

Questions for discussion

- Do teachers make difference between Digital Literacy and Informatics in their classes?
- What is the role of Computational Thinking in Informatics curriculum?
- What about Informatics education in primary schools?