From Algorithms to Computational Thinking in K-12: the Lithuanian Experience

Prof. dr. Valentina Dagienė
valentina.dagiene@mii.vu.lt
Vilnius University
Lithuania – LIETUVA

- Territory – 65 300 km²
- Population – about 3 mln.
- Vilnius – about 0.5 mln.
- Borders: with Belorussia, Latvia, Poland, Russia and Baltic sea
Short glance to Informatics/IT at School

- **1970**: Teaching about computer and algorithms
- **1980**: Creating programs
- **1990**: Focus on applications
- **2000**: Technology enriched learning
- **2012**: Deepening understanding of Informatics
„Prehistory“ of teaching programming

❖ ~40 years ago (in 1975) – the idea of nation wide teaching of programming in schools in Lithuania has emerged.
❖ Implementation: Teaching material was prepared.
❖ In 1979–1981 the Experimental School of Programming by Correspondence was organized.
❖ 34 years ago (January 1981) – Young Programmer’s School by Correspondence was established officially 
Jaunųjų programuotojų mokykla
Pirmoji pamoka

ALGORITMAI

Gyvenime labai dažnai suteikame iš anksto numatyti nurodymus, kuriuos reikia vykdyti norint atlikti konkrečias darbą. Pavyzdžiu, prie telefono automatų galima rašti instrukciją, kurioje trumpai ir aiškiai pasakyta, ką reikia daryti, norint paskambinti:

1. Imkskite dviejų kapeikių monetą į automato skylę.
2. Nukelkite ragejį ir laukite signalo.
3. Išgirdėtę ilgą, nepatrukiamą gaudėsį, sustokite reikiamai įspėjimą ir laukite atsakymo signalo.
4. Išgirdėtę ilgus gaudėsį, laukite, kol abonentas atsakys.
5. Išgirdėtę trumpus, dažnai pakeičiantis gaudėsį, pasakinkite ragejį ir išimkite monetą.


TRYLIKTOJI PAMOKA

Skyreli tvarko LTSR MA Matematikos ir kibernetikos instituto jaunesnioji mokslinė bendradarbi Valentina DANIENĖ

Programuotojai, rašantys neįtikėtinai kai kurių programų, mėgsta teksistis, kad programa skiriamasi kompiuteriui, nera žmogus. Be abejo, kompiuteriui programos aiškumas nesvarbus – jis mechaniskai atlieka veiksnius ir nesidomis programos valdymu. Tačiau kad ir kaip atrodytų keista, didžiausias programų skaitmenės vis dėlto yra žmogus, o ne kompiuteris. Skaitmenės programos, žmogus susipažins su kitų programuotojų idėjomis ir patirtimis, moksis pati sudarinti programas. Dažnai tenka tobulinti ir pačių su

mentarais galima paaškinti ne tik kintamųjų vartus, bet ir atskirų programos dalis, nurodyti, ką vienas ar kita sakyk yra atlieka ir panašiai. Komentarai galima iterpti visur tarp atskirų simbolų, žodžių, skaičių, vartus. Jie susiklaudžiamai skliaustais ('*').

Komentarai padeda greitai ir lengvalai skaityti programas. Čia jau nereikia piktnaudžių – komentarai turi būti jokio išsprendžių, greičiau, trumpai nusakantys pagrindinius dalykus, neuzgrūždantys programos teksto.

Paminėkime dar vieną programavimo kultūros elementą – programų redagavimą. Redagavimu vadinas programos teksto lūždymas popieriaus lape. Nekyla abejonų, kad žmogus kur kas lengviau skaitydavusiai lūždymą programą. Be to, tokioje programoje būna mažiau klaidų (pavyzdžiui, sunkiai pamiršti žodį eilę, tai jis rašomas, to

rojo mėnesio pabaigoje prielagu duos tik pirmoji pora, todėl turėsime tris poras, o dar po mėnesio prielagu duos ir pradinė pora, ir pora, gimusi prieš du mėnesius. Todėl viso bus 5 poros.

Simbolis F(n) pažymėkime trišių porų skaičių, kurį turėsime po n mėnesių. Matomu, kad n-ojo mėnesio pabaigoje turėsime tiek porų, kiek jų buvo prieš mėnesį, t.y. F(n-1) ir dar tiek naujų porų, kiek jų buvo prieš du mėnesius, t.y. F(n-2) o meno pabaigoje. Kitaip sakant, gauname tokį priklausomybę: 

$$ F(n) = F(n-1) + F(n-2) $$

Pateikiamo programą trišių porų skaičių po n mėnesio spausdinti:

```
program fibonnaci;
var fn, (* F(n) *),
fn1, (* F(n-1) *),
fn2, (* F(n-2) *),
n, (* mėnesio skaičius *),
k: integer;
begin
write ('* F(n) = ');
for s:= 1 to n do
begin
write (s);
for d:= 1 to n do
if s mod d = 0 then
write (' + ');
end
writeln;
end;
```

End.

Jis nėra efektyvus, tačiau kadangi iš uždavinio sąlygos nereikalingai, kad n būtų labai didelis, tai skaičiavimus bus naudų ir nėra reliau šių aspektų tobulinti programas.

KONTROLINIAI

UZDAVINIAI

13. Duota programa: program atsą pes:
var a, b, c;
 1, j: integer;
begin
  read (n);
  a := 1; b := 1;
  for i := 0 to n do
  begin
    for j := 1 to b do
    begin
      for k := 1 to i do
      begin
        a := a + b;
        write ('* ');
      end
    end
  end
  writeln;
end.

Ką ši spausdina kompiuteris, atlikę šią programą, jei pradinis duotuo yra 77? Kokio uždavinio sprendimas užrašytas ši programą (7 halai).
The curriculum and content

- Names, variables, values, assignment statement and sequence of statements
- Branches of actions
- Repetition (Loop)
- Program and its running by computer
- Logical values
- Functions and procedures
- Recursion
- Discrete data types
- Real numbers and records
- Arrays
- Programming style
- Program design
- Efficiency
Program reading was considered as important as writing

- Programmer must be able to read and investigate the programs designed by other programmers.
- Program analysis – reading makes a considerable part of home tasks.
- Pupils were asked to modify a program in order to adapt it to another similar problem, to restore the omitted parts of its text, etc.
Development periods of the Young Programmer’s School by Correspondence

5. Using new media (virtual learning environment) while learning algorithms (since 2005)
What is valuable: Lessons learned

- **Tasks**: Interesting, attractive tasks (small pieces) and problems (connected to real world) for students
- **Flexible learning**: Learning should occur anytime anywhere
- **Learning resources**: Divide learning material in pieces, combine practical problems with theoretical approaches
- **Teaching resources**: Well-prepared material for teachers
- **Face-to-face meetings**: summer camps, teacher training sessions...
- The successful introduction of Informatics is determined by a change of mental habit and promoting co-operation among researchers, policy makers, and teachers
Logo

- Learning Logo
- Logo contests
- Logo workshops for teachers
- Books for schools (project-based method)
- “Comenius Logo“ learning environment

Valentina Dagiene and Seymour Papert
Hanoi, Vietnam, 2006
Informatics in schools of Lithuania: 1986

- The official beginning of informatics as a subject in schools of Lithuania is dated back to 1986
- Main idea: introduce each pupil to a computer
- *Programming is the second literacy* - prof. Andrei Ershov, a founder of the Siberian School of CS
- We translated the textbook and added a chapter on Pascal programming language
Informatics in schools of Lithuanian: 1991

- In 1991 the first curriculum for teaching informatics in secondary schools was developed.
- An original Lithuanian textbook of Informatics was written just after Lithuania has regained independence.
- The course was lectured for two years in school-leaving grades 10th – 12th of upper secondary schools, composing it in different ways.
Since 1997 the teaching of informatics essentially changed:

- the **compulsory** course for the 9th and 10th grades was introduced
- Informatics remained **compulsory** for the 11th and 12th grades as well
- the possibility to take **advanced optional modules** was provided
Informatics in schools of Lithuanian: 2005

- Since 2005 the basics of informatics and IT as a separate subject has been introduced to grades 5 and 6
- Changing name: Informatics to Information Technologies or IT
- IT – a compulsory subject at 5 – 10 grades
  - 1 hour per week (35 hours per year) for grades 5 and 6
  - 7 or 8
  - 9 and 10
- Optional modules for grades 11 and 12 – programming, data base, desktop publishing – for upper secondary school
Lower Secondary School, grades 5-10

5-6 Grades
- Handling information by computer
- Drawing with computer
- Text processing and formatting
- Internet and services
- Modeling with computer

7-8 Grades
- Handling information by computer
- Text processing and formatting
- Data handling and spreadsheets
- Presentation

9-10 Grades
- Handling information by computer
- Text processing and formatting
- Internet and services
- Data handling and spreadsheets
- Introduction to programming
- Desktop publishing
- Webpage design
Upper Secondary School, grades 11-12

General course
- Formatting texts
- Spreadsheet
- Presentation
- Internet security, ethics

Extended course
- Programming
  - Desktop publishing
  - Data Base developing and management

9-10 Grades
- Introduction to programming
- Desktop publishing
- Webpage design
## IT in 5–6 grades

<table>
<thead>
<tr>
<th>Themes, subthemes</th>
<th>IT hours</th>
<th>Subjects, integration is addressed to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to computer application</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Principles of computer use</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Drawing with computer</td>
<td>4</td>
<td>Art; 10</td>
</tr>
<tr>
<td>Text and keyboard</td>
<td>14</td>
<td>Mother tongue; 10</td>
</tr>
<tr>
<td>Internet and electronic mail</td>
<td>10</td>
<td>Mother tongue; 4 Foreign language; 10</td>
</tr>
<tr>
<td>Modeling <em>(Logo or Scratch)</em></td>
<td>24</td>
<td></td>
</tr>
</tbody>
</table>
Textbook for 5-6 grades

- Introduction to computer
- Drawing
- Text processing
- Internet and emails
- Modeling (Logo)
## Contents of Informatics and IT subjects

<table>
<thead>
<tr>
<th>9-10 grades (Compulsory course)</th>
<th>11-12 grades (Optional course)</th>
<th>11-12 grades (Advanced modules)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer (principles of the work)</td>
<td>Advanced elements of text editing</td>
<td>Data base</td>
</tr>
<tr>
<td>Text processing</td>
<td>Presentation</td>
<td>Multimedia</td>
</tr>
<tr>
<td>Information (basics of information handling)</td>
<td>Web and email</td>
<td>Programming</td>
</tr>
<tr>
<td>Algorithms (main concepts and commands)</td>
<td>Social and ethical issues of using IT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Spreadsheet</td>
<td></td>
</tr>
</tbody>
</table>
Matriculation in Lithuania

- All high school students are required to take matriculation exams in main subjects studied in high school: at least two and not more than 6.
- Informatics (IT + Programming) is one of them
- The exams are external nation wide exams - National Examination Centre is responsible.
- Evaluation of an exam is by points from 16 to 100.
## Informatics exam structure

<table>
<thead>
<tr>
<th>Two parts</th>
<th>Questions</th>
<th>Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer literacy tasks</td>
<td>1-2 multiply choice questions</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2-3 short answer questions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2-3 open-ended questions</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>1 task with text processing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 task with spreadsheet</td>
<td>20</td>
</tr>
<tr>
<td>Programming tasks</td>
<td>2 or 3 practical tasks to be programmed</td>
<td>50</td>
</tr>
<tr>
<td>Algorithms</td>
<td>Data structures</td>
<td>Control structures</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>------------------------------------------------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>Calculation of the sums (of product, quantity, and arithmetical average).</td>
<td>Integer and real, char, boolean, and string.</td>
<td>Program structure.</td>
</tr>
<tr>
<td>Search of the maximal (minimal) value.</td>
<td>Text file.</td>
<td>Comments.</td>
</tr>
<tr>
<td>Data input/output.</td>
<td>One-dimension array.</td>
<td>Variables.</td>
</tr>
<tr>
<td>Data sorting.</td>
<td>Record.</td>
<td>Assignment and sentence.</td>
</tr>
<tr>
<td>Modification of algorithms according to the particular data structures.</td>
<td>Development of data structures.</td>
<td>Logical operations, if statement.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Loops.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compound statement.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Procedure and function.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Parameters and arguments.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Standard files.</td>
</tr>
</tbody>
</table>

Programming tasks

- **A first task** is intended to examine the students’ abilities
  - to use the procedures or functions,
  - to use the data types,
  - to realize the algorithms for work with data structures,
  - to manage with input and output in text files.

- **A second task** is intended to examine the students’ understanding and abilities to implement data structures. The core of the task is to develop the appropriate structures of records together with arrays. Students usually are asked:
  - to input data from the text file to arrays containing the elements of record type,
  - to perform operations by implementing algorithms,
  - to present the results in a text file.
Evaluation system

- Both tasks intentionally requires to write batch style programs, as they are more suitable for blackbox testing.
- Semi-automatic evaluation system with blackbox testing is developed for evaluation of programs.
- Exam evaluation system has different requirements than systems used in programming courses or programming contests.
# Evaluation schema of a programming task

<table>
<thead>
<tr>
<th>First task: evaluation criteria</th>
<th>Points</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tests</td>
<td>20</td>
<td>If the program provides correct outputs to all tests.</td>
</tr>
<tr>
<td>Correct reading from file</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>The result is outputted correctly</td>
<td>2</td>
<td>Evaluated only if the program scores no points for the tests.</td>
</tr>
<tr>
<td>The function, which calculates the number of chess sets that can be</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>collected from the pieces brought by the students is created</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other functions, procedures (if there are ones) and the main program are</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The data type of array is declared correctly</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>The function is created</td>
<td>1</td>
<td>Always evaluated.</td>
</tr>
<tr>
<td>Meaningful names of the variables. Program parts are commented, spelling</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>is correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Programming style is consistent, no statements for working with the</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>screen.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>25</strong></td>
<td></td>
</tr>
</tbody>
</table>
IT and Informatics in school education of Lithuania

Pre-school year

Primary education

Informatics introduced as a stand-alone subject from grades 5 to 12!!!

From 2015: Modern Informatics/IT for all children with elements of programming

IT and Informatics for all with elements of algorithmics

Secondary education

Informatics and IT advanced – elective modules

Tertiary education – University

IT and Informatics as a stand-alone subject in 1986 for HS has never been removed !!!

Informatics related to computer science in school

IT and Informatics in school education of Lithuania

Pre-school year

Primary education

Secondary education

Tertiary education – University
Modern curriculum: Informatics and IT

1. **Understanding and analysis of problems** based on logical and abstract thinking, algorithmic thinking, algorithms and representations of information.

2. **Programming and problem solving by using computers** and other digital devices – designing and programming algorithms; organizing, searching and sharing information; utilizing computer applications.


4. **Developing social competences** – communication and cooperation, in particular in virtual environments; project based learning; group projects; equity.

5. **Observing law and security principles and regulations** – respecting privacy of personal information, intellectual property, data security, netiquette; positive and negative impact of technology on culture, social live and security.
Supporting activities

- **Teacher preparation:**
  a teacher is the most important „technology”!
- Standards, evaluation and **support in a classroom**
- In-service training at universities – based on standards
- **Web service** – materials, MOOCs
- Comments to the curricula of other subjects – how to use **computational thinking** in solving problems
- **PBL and flipped learning** – extra hours of school learning
- Gamification
- Contests: *Bebras*, Olympiads...
- Informatics oriented tasks in national school tests
Challenges

- How to motivate and engage students through K-12, for 12 years, e.g. learning programming requires constant practice
- The role of coding – programming
- When and how to switch from visual to textual programming?
  - Visual – for beginners, non-professional
  - Textual – for those who seriously think about CS – we don’t want to loose them
The curriculum – general comments

Informatics ≠ programming
Concepts before tools, before programming

There are plenty of ways to introduce/teach informatics concepts ... without computers:

- CS unplugged
- Bebras tasks

When appropriate, we can extend unplugged CS by adding ... a computer

- Grades 7-9 – focus on real world problems and applications which are meaningful for pupils
- Grades 10-12 and vocational schools – CS/ICT specializations
The curriculum – the role of programming

- Remember: Informatics ≠ programming
- How to use extra curricular coding activities (e.g. the Hour of Code) in the classroom?

Programming

- Programming is a tool, not a goal
- Which programming language? – there are 3000
  - any, which can be used to introduce and illustrate concepts
  - introduce new constructs when needed
  - a program is a message for a computer and also to other people
  - different languages different programming methods
  - visual versus textual languages and programming
Methods of introducing Informatics concepts

❖ Use all three forms of activities:
  ❖ **visual** learning: pictures, objects, abstract and physical models, ...
  ❖ **auditory** learning: exchange ideas, discussions, group work, ...
  ❖ **kinesthetic** learning – physical activities

❖ Learn/teach in environments of three stages:
  ❖ **cooperative games** and puzzles that use concrete meaningful objects – discovering concepts: Bebras tasks, The Hour of Code
  ❖ **computational thinking** about the objects and concepts – algorithms, solutions
  ❖ **programming** – Scratch, The Hour of Code, Logo

❖ **Bebras tasks** – the source of problem situations
❖ **The Hour of Code** – introduction to (visual) programming with puzzles
The Hanoi Towers

- The Hanoi Towers story
- *In the beginning:* ask kids to play and try to find „an algorithm” and calculate the number of moves for different numbers of rings
- *Expected:* algorithms and tables with the number of moves
- *Then:* kids play with (against) a computer program
- *Finally:* they verify initial findings
- *Extra (MS, HS):* recursive solution, minimum number of moves

Concepts:
- game
- algorithm
- efficiency (complexity)
- recursion
Shortest path – introduction

Kids are working with real situation – motivates them:

- **Computer:** Find your house and your school on the Google map. Find your way to/from school.
- Find **shortest paths** (distance and time) to/from school by different transportation means: on foot, by bicycle, by car, public transportation.

- **Paper and pencil:** Table to compare which is the shortest path (time/distance) to school?
Typical approach, a greedy type: the nearest neighbor method. It doesn’t work!

However it works when you go from Diamond to Einstein !!!

Think: Dijkstra’s algorithm is a greedy method and optimal
Shortest path – Beaver task

From START to FINISH

Find the quickest path from START to FINISH. The hours in the figure indicate how much time must be spend to travel between the adjacent stations.
Conclusions

With the modern Informatics/IT curriculum:

- Students acquire a broad overview of informatics/IT and applications.
- Teaching informatics focuses on problem solving and CT.
- IT/Informatics is taught independently of application software, languages, environments – students are free to make their own choice.
- IT/Informatics is taught using problem situations coming from school subjects and real-world applications.
- IT/Informatics education provides a background for the professional use of computers in other disciplines.
- Students experience a solid foundation in CT through problem solving with computers.
- Students experience that programming is a creative process.
- Students learn how to collaborate on projects.
- IT/Informatics enables innovation also in other fields.
- PBL and flipped methods contribute to personalization of learning.
Thanks you for your attention!

valentina.dagliene@miii.vu.lt