Algorithmic thinking: what set of Logo constructions should be used to develop it?

Valentina Dagienė

Institute of Mathematics and Informatics
Akademijos St. 4, Vilnius 2600, Lithuania
Tel. (370-2) 729606, fax (370-2) 729209,
E-mail: dagiene@ktl.mii.lt

Abstract

This paper discusses how to teach algorithms in the school curriculum of informatics, how to develop algorithmic thinking. Teaching of algorithms is included in the compulsory course of informatics in Lithuanian comprehensive schools. Mostly the Pascal programming language is chosen as tool of introducing algorithms. Alternative ways of teaching algorithms, e. g., flowcharts, are also recommended for visualisation of algorithms. However, Logo is a very suitable tool for developing algorithmic thinking as well, especially for basic education. The way of using Comenius Logo for teaching algorithms in the curriculum of informatics is discussed.

Keywords: Teaching algorithms, Developing algorithmic thinking, Pascal, Comenius Logo

1. Historical view of teaching algorithms in Lithuanian schools

Since 1986 a separate course of informatics has been delivered at comprehensive schools in Lithuania. When generalizing the experience available, we may make out some stages of its evolution:

Stage 1 1986–1991
An academic, scientific trend of informatics. Fundamental knowledge of computer science and theoretical programming principles are prevailing here.

An algorithmic (practical) trend of informatics. Fundamental knowledge of informatics is taught: computer systems, logics, algorithmization. Attention is paid to developing of logical and algorithmic thinking. Also, more and more attempts are made to link with practice.

Stage 3 1999–
The following phases of informatics take a clear standing: at a younger age constructive education, at a medium age systematic knowledge and skills of an obligatory nature, orientation towards training of practical skills by computer, development of algorithmic thinking; at an older age, – differentiation of teaching by means of a modular system; attention is paid to training technological skills and developing thinking related with that, - implementation of information and communication technology into general education (Dagiene, Markauskaite, 1997).

In the first stage of development, the main attention was paid to teach algorithmization. That is quite natural, because two decades ago there were almost no computers in schools (and those present in some schools were so poor that that only the simplest algorithms may be run). There was no possibility of training skills of practical work with computer, therefore the whole attention was focused on fundamental concept of informatics – algorithmization. Besides, teachers of informatics were mostly mathematicians (about 90%). So algorithms were quite acceptable to them.
With a prosperity of information technologies and a reduction in prices of personal computers, ever more and more attention was paid to the solution of problems of an applied nature, first of all, to processing of textual and visual information as well as acquiring some conception of modern information and communication technologies. At the initial period (grades 5-8) of teaching informatics, the Logo trend was gaining more and more flourishment.

Since 1999, after reform of education system the compulsory course of informatics was lowered by a step, – the time limit was fixed to teach it only in school-leaving grades (9-10) of the basic school.

The compulsory course of informatics in the 9th and 10th grades in Lithuanian schools is focused on four main topics (Dagiene, 1997): 1) information, 2) algorithms, 3) computer, 4) information technologies. The whole compulsory course involves approximately 70 hours of classroom time over one or two years in total.

Note again that, regardless of whether the school has computers or not, this course is compulsory. The main goal of the informatics course is to develop students’ thinking skills. The informatics course in the schools with computer labs is based on a more extensive curriculum.

The algorithm course includes the following subjects.

- **The concept of algorithms.** Algorithms in nature and society.
  - An algorithm is the basis of computer information processing.
  - The features of algorithms: discreteness, clearness, mass character, resultativity.

- **Ways of algorithm presentation and writing.** Structured schemes (flowcharts). Algorithm and program. Running a program by computer.

- **Data.** Their types.

- **Main algorithmic constructions (control statements).** Sequences. Branching of actions.
  - Iteration of actions (loops)

- **Stages of algorithmic problem solving.** The principle of splitting a problem into parts.

- **Creator, performer, and user of the algorithm.** Their interrelation.

As one can see from the contents of the algorithmization course, students are acquainted with and taught to use the main algorithmic constructions. It is of utmost importance to teach students to write complete programs of small problems and to be able to document them.

At the stage of a secondary school (grades 11-12), the curriculum of informatics is made up basing on the modular principle. Three modules are offered for the compulsory general course of informatics for all profiles: ‘Information technology’, ‘Information resources’, and ‘Elements of programming’. After coordinating the wishes of students, their parents and teachers, schools can compose these modules arbitrarily, only with regard to the total number of hours allotted to the course (about 70 hours in total in two years). So teaching of programming is not compulsory for everybody, it can be selected only regarding to the needs.

2. Developing of algorithmic thinking

The key aim of teaching of informatics in our comprehensive schools is to develop students’ information literacy in a broad sense. This goal is timely, urgent, and of constant value. With a lapse of time the conception of information literacy may change.

In our opinion, the information literacy (or information culture, in a broader sense) is a wide concept, considerably wider than information skills or abilities to work by computer. We consider that at present the conception of information literacy covers various abilities and skills, one of which is to develop logical and operational thinking, operation planning skills, creativeness, ability of improvising, self-confidence.

Different subjects at school develop different modes of thinking. It is of importance for experts of informatics that there exists a specific way of thinking, called an algorithmic one, that is closely related to computers and information technology. Developing of an algorithmic style of thinking is one of the problems
of teaching informatics. We refer here to the general regulations of our education that a contemporary school must educate a man of versatile thinking. Thus, if we isolate a particular style in man’s thinking, its developing is important as to be promoted. The significance of developing the algorithmic style of thinking was emphasized by Papert (Papert, 1980).

What does algorithmization give our students?
By algorithmization it is attempted to teach to comprehend the meaning and power of formal notation, to be able to use that for describing the action to be performed. This part of the course is in close relation with mathematics. Undoubtedly, comprehension of algorithmization requires a great deal of mathematical knowledge. However, in mathematics formalization of algorithms, mastering of the ways of solution, as usual, specialized models of problems are stressed rather than getting solution of real problems.

In algorithmization, formalization of the representation of not only operations but also descriptions, and the language of communication selected as well as product presentation are emphasized. Attention is paid to the conception of formal language constructions. It is also attempted to convey the methods for solving real problems and capabilities of solving them by means of computers.

The part of algorithmization in the general compulsory course of informatics is rather small (only a quarter of hours of the entire course is devoted for that). During teaching this subject, it is necessary to find students gifted at programming and to encourage them for further studies in this direction.

Development of algorithmic thinking, however, is an extremely difficult problem. If the students in mathematics, physics, or linguistics usually are supplied with ready-made algorithms to solve their problems, in informatics they have to create algorithms themselves. And this is creation that is associated with scientific activities, mastering of which requires a lot of effort and inventiveness. That is why algorithmic thinking is worthwhile developing from childhood, at a younger age of pupils, in order to train and develop universal skills.

Several methods are proposed to train the algorithmization style. One of them is a permanent mental activity, i.e., solving lots of problems that are advisable to be presented to students setting according to their complexity (Dagiene, 1999b). The whole attention should be paid to get the gist of algorithms themselves rather than the means of their description (programming language). Therefore one should do this with a minimal set of the most indispensable algorithmic constructions.

Usage of the structured method (top-down) plays an important part in the development of algorithmic thinking. This method can be perceived and applied on a much wider range than pure programming: to plan our everyday activities, to solve complex problems, etc. The application of this method, however, requires a large set of problems, best prepared and easy run by computer leaving only certain clear activity for students. There are no problems of this kind so far. Therefore, the efficiency of this method is not so high, as the students are able to complete only one or other more complex problem during a small number of hours allotted for the course. Thus, they will not train up their skills of algorithmic thinking.

It is very important for students to observe the process of algorithm performance. Therefore it is desirable that the students perform algorithms using a pencil and paper or, with more initial data, using step-by-step algorithm performance routine.

3. Alternative ways of teaching algorithms
The principal conceptions of procedural programming that lay as a basis at the start of teaching of algorithmization fundamentals are data types and control structures. In 1986-1998, the course of informatics acquainted students with the following data types: integer, real, boolean, array (for an advanced level) and control structures, namely, assignment, conditional (if statement), loops (while and for), procedure and function.

After transferring the course of informatics into the basic school, there were attempts to simplify algorithmization. In grades 9-10, only three types of data (integer, real, boolean) and the main control structures (assignment and a conditional statement as well as two types of loops) are used. Since procedures
and functions were refused (algorithm is described as a program), for data input and output it is necessary to consider read and write procedures.

Algorithms can be taught by various methods: using formal languages, flowcharts, computer programs, etc.

Since the very emergence of informatics it has been proposed to teach algorithms in Lithuanian schools using Pascal, – the school textbook has been based on it for many years (Dagiene, 1992). It should be noted that the Pascal language was used only to describe algorithms, to perceive and execute algorithmic constructions. But no effort was put to teach the Pascal programming language itself, no going into technical details or language subtleties.

No concrete language was proposed for teaching algorithms in the general curriculum of informatics of the reformed basic school. However, a part of the textbook was prepared (Dagiene, 1999), based on the usage of minimal constructions of the Pascal language: it included integer and real numbers, boolean type, assignment, conditional, and loop statements.

Rather often some graphical means – flow charts and structured schemes, etc. – were employed as a teaching aid of algorithms. That makes it easier to perceive operations written as algorithmic constructions. However, visual schemes are usually used along with some programming language, and not only as a form of algorithm presentation.

Structured schemes are very convenient to describe solutions of applied problems that require more complex Pascal constructions, and their learning would be too difficult.

At present, there is a world-wide tendency of developing structured flowcharts to represent algorithms. Assembly of algorithms out of components is especially attractive, it looks like a construction of Lego games (Roschelle, 1999).

This kind of algorithmization might be much more attractive to schools. It is of great interest now in the world, while nothing is being done in this direction in Lithuania.

4. **Logo – an alternative to Pascal**

It is as long as several years that teachers of informatics in our schools acquaint the students of grades 9-10 with algorithms by using the Pascal programming language. The majority of polled teachers of informatics state that at the basic school students can hardly perceive formal notation of the Pascal programming language and even more difficult it is for them to comprehend an entire algorithm and to run it by computer. The major part of reasons is stipulated by the software itself, – it is not friendly-disposed to a beginner (Turbo Pascal used by us relies on a bit old-fashioned DOS operational system).

A nice Logo teaching tradition has survived for many years at some of our schools, this language is employed in the extra classroom teaching, but as usual, in lower grades (5-8).

The question is as follows – is there any possibility to use Logo for training the main algorithmization skills in older age? There is some experience of this kind in the world (Blaho, 1995; Blaho, 1997; Brusilovsky, 1997). Slovakian scientists have prepared abundant nice methodological material for work with the Comenius Logo program. Their book ‘Super Logo: Learning by Developing’ (Blaho, 1998) is a depository of wonderful algorithms. The whole experience of other countries might be applied in the training medium of our country.

It is important for the beginners to see the result yielded by algorithms as soon as possible. In this respect, the Logo language is more advantageous: commands are executed by a dynamic object (a turtle), they are interpreted and followed by an immediate result (not only numbers, but also graphics, texts, forms of windows, buttons and so on).

Let us consider to what extent constructions of the Comenius Logo program suit to describe algorithms according to the course of informatics at our schools (Table 1).
Table 1: How Pascal and Comenius Logo are suitable to express algorithms in the course of informatics of the basic school

<table>
<thead>
<tr>
<th>Teaching of concepts of algorithms (following by the informatics curriculum)</th>
<th>Pascal</th>
<th>Comenius Logo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data. Real and integer numbers. Input and output. Control data.</td>
<td>Deeper understanding of data conception.</td>
<td>No special effort necessary to perceive integer and real types.</td>
</tr>
<tr>
<td>Basic control structures of algorithms. Sequence of statements. Conditional branch. Repetition (loop).</td>
<td>Control structures a bit overloaded by additional details; hardly conceivable formal notation (e. g., loop for).</td>
<td>Extremely visual and simple control structures.</td>
</tr>
<tr>
<td>Algorithmic style and layout of the algorithm text</td>
<td>More settled rules, a perfect representation of structured programming principles.</td>
<td>Almost a free style, rather often lacking text delivery visuality.</td>
</tr>
</tbody>
</table>

Let us analyse more thoroughly how the basic algorithmic constructions, used in the course of informatics in grades 9-10, would be written in the Logo language.

**Data types.** As usual, in Logo there is no need to describe data types, which is hardly perceivable for the beginners of programming Integer and real numbers are used naturally. The is no boolean type in an explicit form.

**Variables and assignment of values.** The notion of a variable is also hardly perceivable by a beginner. In the Pascal language, the attributes of a variable (name, type, value) are defined by a name and type definition, and by assignment statement or reading procedures. The type definition may be not presented for the beginners, while descriptions of names are dispensable.

In the Logo language the same command gives the name and value to a variable, e. g., make "a 15 or name 15 "a. It is clear and simple.

In the Logo language the problem of local and global variables is solved simply and visually: if we wish to describe a local variable (in the procedure), we use a command let "a 15 (the variable a is declared as a local one and thereby it is assigned a value). There is another possibility for a variable to become local, - by using a special command local.
Conditional statement. The Logo language possesses some kinds of conditional statements (*if, iftrue, iffalse*). The most common conditional statements are:

- `if <condition> [list of instructions]`
- `if <condition> [list of instruction 1] [list of instruction 2]`

These statements are quite analogous to those of the Pascal language (only the word *else* that indicates the opposite actions is missing).

Loops. It is not so simple to convey the essence of a loop, therefore it would be desirable that its formal expression were as simple and visual as possible. It is hardly possible to simplify the expression of a loop more than it is in the Logo language:

- `repeat N [list of instructions]`

This loop is obvious: the commands written in brackets should be repeated N times.

In Comenius Logo there are also other loop instructions *while* (in Pascal it corresponds to *while*), *for* (corresponds to the *for* loop in Pascal but even more general: one can indicate the size of a step of the loop variable), *foreach* (mostly for processing lists).

Procedures. Our course of algorithmization of the basic school does not provide for teaching procedures. It is not so easy to explain Pascal procedures (and especially functions). In the Logo language, however, every algorithm is described by procedure – naturally and simply. Therefore, the students using the Logo language ought to easily perceive procedures and be able to apply them in solving problems.

In order to teach algorithmic thinking using Logo you need a set of Logo instructions. We have discussed a core set of Logo constructions (Table 2).

Table 2: Main Logo instructions needed to develop algorithms in the introductory course of informatics in the basic school
<table>
<thead>
<tr>
<th>Group of instructions</th>
<th>Names of instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turtle Geometry: moving</td>
<td>back, forward, heading, hideturtle, home, left, pos, random, right, setheading, setpos, setx, setxy, sety, showturtule, towards, xcor, ycor,</td>
</tr>
<tr>
<td>Turtle Geometry: drawing</td>
<td>background, clearsreen, fill, fillcolour, pencolour, pendown, penerase, penreverse, penup, penwidth, setbackground, setfillcolour, setfillmode, setfillpattern, setpencolour, setpenwidth,</td>
</tr>
<tr>
<td>Control structures</td>
<td>case, foreach, if, iffalse, iftrue, let, local, make, name, output, play, print, repcount, repeat, run, stop, to,</td>
</tr>
<tr>
<td>Data structures</td>
<td>abs, ascii, butfirst, butlast, char, count, div, first, fput, item, last, lput, mod, round, sqrt,</td>
</tr>
<tr>
<td>Creating Turtles</td>
<td>allturtles, ask, each, eraseturtle, maketurtle, tell, who,</td>
</tr>
<tr>
<td>Animation</td>
<td>emptyimage, frame, getimage, getshape, hotspot, image, loadimage, putimage, savelimage, setframe, setframemode, sethotspot, setshape, setshapecolour, stamp, wait,</td>
</tr>
<tr>
<td>Controlling by keyboard and mouse</td>
<td>key?, mouse, mousestate, readchar, readkey, readlist, readword, setpattern, setmousecursor, sound, touched</td>
</tr>
</tbody>
</table>

5. Conclusions and future prospects

1. The key goal of teaching informatics at our comprehensive schools is education of students’ information culture. The notion of information culture is broad, – it embraces a lot of capabilities, one of which is development of logical, operational, algorithmic thinking.

2. Only a small part of the course of informatics - approximately a quarter of it – is allocated to algorithmization. This course is an attempt to convey to students a perception, what algorithm or program stand for as well as what programming means, and to acquaint students with the principal constructions of algorithmization.

3. Algorithms are usually expressed by a subset of the Pascal programming language. Despite that it is stressed to pay more attention to perceive algorithms themselves, the notation of the Pascal language is specific and one has to allot a lot of time to it.

4. An alternative way of expressing algorithms is proposed – a Logo language, or to be more precise, the Comenius Logo software. After exploring the equivalents of the basic algorithmization constructions to those in the Logo language, it has turned out that they are convenient and much more suitable for the beginners to learn programming.

5. We have gained a great experience of teaching algorithms in the Pascal language, many methodical means and books of problems have been prepared. To teach algorithms in the Logo language, we have nothing in this kind. It is clear, we have to do that. There is a particular need to prepare a collection of problems that develop algorithmic thinking: from simplest to complex.
6. References


Dagiene V (1999) Informatikos pradmenys (Elements of informatics), textbook for grades 9-10, in Lithanian, Vilnius: TEV.


Dagiene V and Markauskaite L (1997) Information Technology in the School Subject of Informatics in International Conference and Exhibition ‘Information Technologies and Telecommunications in the Baltic States’, Riga, 2-5 April, 137-142.
