INFORMATICA, 1990, Vol.1, No.1, 156–166

SOME ASPECTS OF TEACHING THE ART OF PROGRAMMING BY CORRESPONDENCE

Gintautas GRIGAS

Institute of Mathematics and Cybernetics,
Lithuanian Academy of Sciences,
232600 Vilnius, Akademijos St.4, Lithuania

Abstract. The role and goals of computer programming in the education of young people are discussed. The activity of the Lithuanian Young Programmers School by correspondence is described. Motivation of its curriculum and the way of teaching are given. Relations with the competitions of programmers and some other forms of teaching are considered.

Key words: teaching of programming, problem solving, teaching by correspondence, competition, creative thinking.

Introduction. Three institutions concerned with teaching programming by correspondence were organized in the last decade in Lithuania. Namely, the Lithuanian Young Programmers School by correspondence and two nationwide competitions: the (professional) young programmers competition and TV competition of informatics.

The School was set in 1981. In the early 80s it was the only educational institution,enabling to get a primary acquaintance with computer programming for the major part of school children in Lithuania, especially in the provinces. Recently the situation has changed. Computers are available both at schools and home. The subject of informatics is introduced into secondary schools compulsory teaching program. However, the Lithuanian Young Programmers School successfully coexists with other ways of teaching due to its particular role in teaching the art of computer programming and its orientation towards the stimulation of creative thinking and skills of problem solving (Grigas, 1989). To gain the greatest benefit,the studies at the School and practical work with personal computer are recommended to be combined.

The two competitions contain elements of activity by correspondence as well, the first rounds being performed by correspondence. All three institutions are aiming to develop a good programming style and habits of problem solving.

Goals and tools. We consider programming not as an ultimate goal,but rather as an intermediate goal or tool to achieve another goal–to teach creative thinking, intelligence and work discipline. Such aspects of programming are discussed in the paper.

The main goals and means (tools) of teaching concerned with computers and informatics in a school are the following:

1. Creative thinking.
2. Problem solving.
3. Programming.
4. Programming languages.
5. Computers and software.

Each item may be considered as a goal. Each item but the first one may be considered as a mean (tool) to achieve the goal indicated by the item next above as a goal.

A computer together with its operating system and other software may be considered as an ultimate goal or as a tool for all its application. It is a tool for execution of the programming language commands. A language itself may be considered as an ultimate goal (by many teachers of programming, unfortunately) or as a tool for another goal–programming as such. Programming itself may be treated not only as a goal, but also as a tool to achieve yet another goal–problem solving or creative thinking.

The subject and the way of its teaching may be built with any of these goals in mind, using the items next below them as tools. The higher goal the greater possibility to develop intellectual capabilities of children, the lower goal the more possibility to convey empirical or technical knowledge.

The well known system of active teaching on computer is Logo developed by Seymour Papert (Papert, 1980). We discuss another system based on the conventional programming with Pascal as a tool language and point out the main differences of the considered systems. Logo is more suitable for young, while programming in Pascal for older children. Logo is based on experiments with computer and needs a computer for all the time of work. The Pascal oriented teaching is based on problem solving by algorithms and needs less computer time, for tests of programs only. These objectives determined the choice of Pascal in the Lithuanian Young Programmers School.

Programming as a subject of teaching. We are considering the programming in the broad sense, including problem formulation, evaluation of the ideas for problem solving, choosing suitable methods for its solution and motivation, algorithmization and writing program texts. So creative thinking is developed through programming and problem solving.

A clear distinction between a computer programmer and a computer user is made. The essential difference between them lies in the end product of their work. The product of a user are results of the execution of a program by a computer. The product of a programmer is a program text itself. A user writes a program for himself only and cannot make a single step without a computer. A computer is necessary for a programmer as well. However, it is indispensable for the last step of his work only, i.e., for program testing. The major part of programming work may be done merely with paper, pencil and a clever brain.

It is possible to learn by correspondence to develop an algorithm but not to use a computer or its software. For many children brainwork is less attractive than say, computer games. Thus, the main task was to find an alluring way to teach programming and make to think.

Now let us look at programming as such from different points of view. Programming may be considered as a science, an art or a craft. To teach a craft one needs an instrument (a computer). Thus, from what remains, we would like to think of programming as an art, because science requires a lot of theory and theory is a poor attraction for school children.

Problems for programming. The attractiveness of programming may be achieved mainly by choosing a set of interesting programming problems, interesting due to an attractive problem formulation, an unusual way of its solving (i.e., programming) or unusual results. The choice of a proper set of problems is of great importance. The examples of such problems may be as follows: to find the minimal set of post stamps to pay for a letter of a given weight, to calculate and print a calendar table, calendar events, biorhythms, to find a number or a set of numbers owing the given property, to find an optimal strategy in some situations of popular games, etc. In order to find out the degree of attractiveness and difficulty of particular problems,the students of the School are regularly asked to fill in forms/questionnaires on the subject.

About a thousand of such problems are considered at the School. A small subset of the problems along with the corresponding solutions are available for the students (Dagienė, 1986).

In order to convey the foundations of contemporary programming methodology to the School students a certain dose of theory, however unwanted, is absolutely necessary. Thus, our decision was to deliver the basic principles of the theory of programming by an indirect way through problem solving. The set of programming problems is chosen with respect to theoretical requirements and in a good programming style, but at no expense of attractiveness.

The programming language. The main tool for a programmer is a programming language. It is well known that the first programming language may be considered as a native language for a programmer. One learns to think in the terms and structures of that language. These habits remain for a long time and sometimes make a decisive impact on further skill and career of a programmer.

Teaching of programming languages is an auxiliary goal and serves as a tool to master the programming. It is enough for a student to know only the basic constructs of a programming language for active use. But the selection of such constructs and the language itself is of great importance.

We are using a subset of Pascal (Dagienė, 1983) without labels, go to statement, packed data structures, pointers, file definitions, cases in records, functional or procedural parameters, some built-in functions (get, put, reset, rewrite, new, dispose, pack, unpack).

The curriculum. All teaching material is divided into 12 chapters as follows:

1. Names, variables, values, assignment statements and sequence of statements.
2. Branches of actions.
3. Repetitions of actions.
4. Program and its run on computer.
5. Logical values.
6. Functions and procedures.
7. Recursion.
8. Discrete data types.

9. Real numbers and records.
10. Arrays.
11. Programming methodic.
12. Program design.

These topics are discussed in two books (Dagienė, 1989; Grigas, 1986) devoted primarily for the students of the School.

Only integers are used in the first four chapters. Reals appear in chapter ten. Such a sequence of teaching numerical data types may be motivated by the following reasons:

1. Results of operations with data types of countable (discrete) values may be always exactly defined.

2. Arithmetics of integers is easily understood by younger children.

3. Only integers are used in the majority of interesting problems. The operations div and mod of Pascal are widely used in the programs of such problems. Many problems are short and simple (e.g., to find perfect numbers, friendly numbers, etc.), though their solutions without a computer is cumbersome. The computer advantages for solving such problems may be also demonstrated.

A special attention is given to the boolean data type and recursion. The motivation is that the logic is a base of the programming as a whole and the recursion may be considered as a bridge between procedural and nonprocedural programming.

Program writing is not the only job of a programmer. He must be able to read and investigate programs designed by other programmers as well. For this reason, the problems of program analysis make a considerable part of home tasks. Students are asked to modify a program in order to adapt it to another similar problem, to restore the omitted parts of its text, etc.

There are no separate topics of a program verification in the curriculum of the School. However, the students are asked to solve some problems with indirect elements of verification, e.g., to find the range of possible values of output data if the ranges of input data are given, or vice versa.

In about one-third of required home works, the students must deliver program texts together with an exhaustive description of an idea of the solution, its motivation, an informal and may be incomplete proof of the correctness of a program, and a set of the selected input data for testing a program.

Every student is asked to carry out a test for every chapter of the curriculum. The solution of every problem is graded by points. The grading depends on the quality of a program text. A program must be correct and of high readability as well, a program text must be carefully structured and idented.

In order to achieve the high readability, it is recommended to observe the following rules:

1. The constructs of a language must be chosen so as to make the text of a program comprehensible.

2. Otherwise, the comprehensibility may be improved by comments included in the text of a program.

3. Otherwise, the comprehensibility may be improved by an additional text supplemented to a program.

The rules must be used in accordance with the above described order.

If a student fulfils his test work successfully, he receives a test of the subsequent chapter. If not, he receives another test variant of the same chapter. Duration of the whole cycle of every test ranges from 4 to 7 weeks. The schedule forces a student to recapitulate separate chapters he has not coped with successfully at once and allows him to make breaks in studies. At every moment of the studies all 12 chapters of the curriculum are in action in the School.

The number of participants of the School is not stable. It consists of about 200-600 active students. (The total number of population in Lithuania is around 3.7 millions).

During summer holidays the best school children are invited to the summer camps for competitions of programming, lectures, and practical lessons with computers.

The competition of young programmers. Since 1985 the competition of young programmers is organized annually. All residents of Lithuania up to the age of 30 may enter the competition.

The competition is performed in two rounds. The first round is passed by correspondence. Conditions of the competition and five problems for the solution are published in the "Komjaunimo tiesa" newspaper in January every year. Solutions are evaluated by experts according to the correctness, readability and perfection of programs. The original ideas of solutions and the style of programming have a considerable influence on the results of evaluation.

Winners of the first round and are invited to go forward in the second final round. They are given three problems. The correctness of programs is established by computers while the human qualities, such as readability, lucidity, perfectness, by experts.

As many as 60-70 youngsters participate in the competition every year. There are many graduates from the Lithuanian Young Programmers School among the participants and winners of the competition. The competition may be considered as a fine event for graduates to demonstrate the skills of problem solving.

The TV competition. Since 1986 the Lithuanian TV provides regular broadcasts "Informatics and computer science". It makes some 20-25 half an hour broadcasts yearly. Three competitions were already carried out in the school years 1986/87, 1987/88 and 1988/89.

The competition is divided into two rounds as well. Exercises of the first round are scattered among the broadcasts according to the topics of the broadcasts. Naturally, some of

TV watchers may miss some TV programs and so loose some of the exercises. Such cases are foreseen. To be admitted into the second round, it is enough to solve correctly about one-third of exercises. Thus, the first round may be considered as a collection of applications for the real competition in the second round. These applications are supported by a demonstration of the minimal knowledge in the field of informatics.

Conditions and problems of the second round are broadcasted by TV and mailed to the winners of the first round.

Authors of the best solutions of every exercise or problem are invited to the TV studio in order to discuss the ideas of solutions and solutions themselves.

Exercises and problems of the TV competition are simpler than those of the competition of the young programmers. Though every TV watcher may participate in the competition, however, skilled programmers as a rule don't take part in it and let the beginners to have a success.

About 100-200 school children take part in this competition every year.

Conclusions. The Lithuanian School of Young Programmers and two competitions of young programmers are distinct forms of activities. Different media for the communication is used as well. However, all three events have the following common features and goals:

1. The most part of problems in control works are unusual and require creative thinking to solve them.
2. Youngsters are independent in their studies and have enough time to analyze and to solve a given problem.
3. Creativity, intelligence and culture of programming are prefered in the evaluation of problems solutions in all stages of all events.

The features named show the orientation of the described activities towards a common goal–to develop the creative thinking and self-dependent attributes of a personality among youngsters. The activities were founded bearing these goals in mind. The expressed ideas and experience have a considerable influence in the making of the curriculum, textbooks and software for informatics as a subject in Lithuanian secondary schools.

Such a teaching comes very close to the programming as an art, while practical programming skills with computer may be considered as a craft. It was observed, that the best results in the competitions of programmers outside the country gain children who are both graduates from the School and have a good practical experience of the work with computer. So we draw a conclusion, that success may be best achieved mastering both the art and the craft of programming.

Acknowledgments. The author thanks to Mrs. Valentina Dagienė, Mr. Kęstutis Augutis and Mr. Viktoras Dagys for the great help and valuable suggestions in carrying out the described ideas into practice.

REFERENCES

Dagienė,V., G.Grigas and A.Petrauskienė (1983). *Programming Language Pascal.* Mokslas, Vilnius (in Lithuanian).

Dagienė, V., G.Grigas and K.Augutis (1986). *A Hundred of Programming Problems.* Šviesa, Kaunas (in Lithuanian).

Dagienė,V. (1989). *First Steps in Programming.* Šviesa, Kaunas (in Lithuanian).

Grigas, G (1987). *The Fundamentals of Programming.* Šviesa, Kaunas (in Lithuanian).

Grigas, G. (1987). Informatics and creative thinking. In *Third International Conference "Children in the Information Age",* Preprints, Sofia, Part I. pp. 229–240 .

Papert, S. (1980). *Mindstorms: Children, Computer and Powerful Ideas.* Basic Book, New York.

Received March 1990

G.Grigas received the Degree of Candidate of Technical Sciences from the Kaunas Polytechnic Institute, Kaunas, Lithuania, in 1970. He heads the Department of Systems Programming at the Institute of Mathematics and Cybernetics. His research interests include abstract data types, programming methodology and teaching.