A novel didactical approach of the decision structure for novice programmers

Katerina Glezou, *kglezou@di.uoa.gr*
Dept of Informatics & Telecommunications, National and Kapodistrian University of Athens

Maria Grigoriadou, *gregor@di.uoa.gr*
Dept of Informatics & Telecommunications, National and Kapodistrian University of Athens

Abstract

This paper presents a novel didactical approach of the decision structure for novice programmers. More specifically, it introduces a pedagogic intervention, using the multimedia programming environment MicroWorlds Pro (MicroWorlds Pro 1.1 - Greek version and the incorporated Logo programming language) for the teaching of decision structure in its single (if…then…end if) and double (if … then … else… end if) form. The educational goal of the particular topic is to enable students to understand the concept, the features and the various forms of the decision structure, in order to be able to apply it properly.

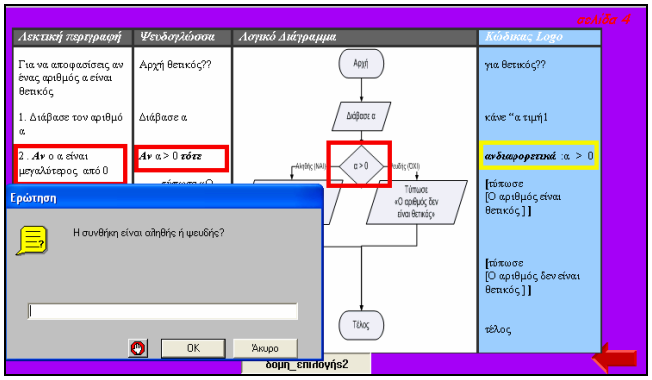

Figure 1. A specific double decision structure example: Visualization of the algorithm and Logo code's flow, and entry of the condition value

The paper's aim is to discuss the basic parameters of an effective alternative didactical approach, in the level of planning, development and implementation, as well as to support the teaching community with particular suggestions for implementation. The particular didactical approach, for which we created microworlds, lesson plans, student worksheets and additional teacher worksheets, was implemented in the framework of teaching computer programming to students (aged 14 years old) of the third grade of Junior High in Greek educational system. The students' response to the teaching process was positive. The students showed interest in their interaction with the MicroWorlds Pro environment and remained active during the lessons, especially in the phase where they had to build their own programs. The students, in many cases, were enthusiastic after being exposed to the "purely" programming part of the lesson and admitted having a "live" interaction with the environment. The findings of this trial implementation reaffirm the view that the use of an open software, like MicroWorlds Pro, which allows the development, reuse and adaptation of microworlds, as well as alternative teaching interventions, promotes the growing of creativity and pedagogical freedom of the teacher, as well as the active involvement of students.

Keywords

Didactical approach; decision structure; Logo; MicroWorlds Pro

Introduction

The international bibliography is full of various approaches for the teaching of the basic programming structures, with a double aim: to deal with the difficulties encountered by students and to ensure the active participation of the latter in the teaching-learning process (Grigoriadou et. al., 2002, Pane and Myers, 2000, Du Boulay, 1989, Soloway, 1986, Spohrer & Soloway, 1986).

The Logo programming language is a powerful tool for the development of algorithmic thought and the visualization of algorithms, especially for primary and secondary education (Glezou et al, 2005, Mikropoulos, 2004, Dagiene, 2003, diSessa et al, 1995, Harel and Papert, 1991, Papert, 1980). Logo's most important feature, which distinguishes it from the other programming languages, is its orientation as an analytical tool for thought and learning processes (Komis, 2005, Papert, 1980). Logo's ability to represent visually the program performance contributes to the understanding of the program function and smoothes out the debugging process (Papert, 1980).

The educational software MicroWorlds Pro, one of the most popular Logo-like environments, is a powerful multimedia environment for programming and an authoring tool and application environment for the development, management and exploration of microworlds.

The study we report here is part of our effort to extend our experience in designing learning environments that support learning through exploration, expression, construction and negotiation. We attempt to gain some more insight into the potential of using MicroWorlds Pro (MicroWorlds Pro 1.1 - Greek version), as a teaching tool for introducing the basic programming concepts to novice programmers in a systematic way, as well as to contribute to the discussion about the main parameters of planning, developing and implementing an effective alternative constructionist didactical approach.

In this paper we present a novel didactical approach of the decision structure in its single (if…then…end if) and double (if … then … else… end if) form for novice programmers.

Theoretical framework

A microworld is an incubator of knowledge since, due to its ability to simulate real world, it offers students the ability to explore a cognitive subject from the inside, aiming at the development of high level cognitive skills that can be extended in various situations (Papert, 1980). The best microworlds have an easy-to-understand set of operations that students can use to engage tasks of value to them, and in doing so, they come to understanding powerful underlying principles (diSessa, 2000). A microworld must be defined at the interface between an individual user in a social context and a software tool possessing the following five functional attributes: a) it is domain specific; b) it provides a doorway to the domain for the user by offering a simple example of the domain that is immediately understandable by the user; c) it leads to activity that can be intrinsically motivating to the user-the user wants to participate and persist at the task for same time; d) it leads to immersive activity best characterized by words such as play, inquiry, and invention; and e) it is situated in a constructivist philosophy of learning (Rieber, 2004).

Logo-like environments can be used to plan and develop microworlds, which offer students the possibility to express and exploit their thoughts, ideas and instincts and to support the process of building knowledge by creating learning environments rich in speculation and opportunities for experimentation (Noss, 1995, diSessa, 1995, Hoyles, 1995, Harel and Papert, 1991). The computational environment can function as a mental scaffolding that allows the structuring of more complex and composite commands, reinforcing, this way, the subtractive thought that progressively develops. When children try to write a program, they actually try to teach the computer how to "think", something that urges them to explore and probably get to the bottom of their own way of thinking, in order to teach it to the device (Harel and Papert, 1991).

In Ackermann's words: "A rich learning environment is one that offers the freedom for genuine exploration, reflection, expression and negotiation, while at the same time providing help and support, when needed. Needless to say, it is not easy to decide how much freedom or guidance makes for a nurturing and yet challenging learning experience. And different people need different kinds of feedback, at different times, in different situations! A good clinician, like a good teacher, is someone who masters the art of providing the "right" amount of elbowroom in each singular case" (Ackermann, 2003). Thus, the creation of interesting and demanding environments encouraging the active and constructive participation of students is a great challenge for teachers (Vosniadou, 2001).

Teaching subject: Decision Structure

The decision structure is one of the three basic logical structures of programming found in every contemporary programming environment. This structure allows the algorithm to chose the commands that are about to be executed, with respect to the verification results of a dual condition. We use it to decide between two alternative situations, one of which is true, while the other one false.

In this didactic unit, students face, for the first time, with the need to introduce the concept of the condition, the condition itself and its features. The possibility of experimenting with and exploring the various alternative routes in the program flow, according to the value of the condition, the recognition of the single and double-alternative form of the decision structure, as well as the ability of applying these forms in simple problems, are considered crucial points.

Learning Difficulties in using Decision Structure

The use of decision structure adds special cognitive difficulties in the general difficulties that have to do with understanding programming. These special difficulties (Komis, 2001) are mostly linked with the following:

- the logical content of the condition, the analyticity and exclusiveness, the logical operations of conjunction, disjunction, negation etc,
- the symbolic representations of these cases,
- the semantic and syntactic properties of the control structure in the programming language used (Du Boulay, 1989),
- the interactions with the representations of the sequential form of the task performance.

Other researchers (Ebrahimi, 1994), (Pane and Myers, 1996), (Tzimoyiannis and Georgiou, 1999) point out as the most important learning difficulties:

- the understanding of the decision structures function,
- the definition of the value (true/false) of the logical statement,
- the designation of the requisite logical statement, within the framework of a problem,
- the grouping of commands in nested decision structures.

In our teaching approach of the control structure, we should also take into consideration that the nature (endogenous: defined by the result of a calculation, or exogenous: defined by a user's interactive entry) of the conditions on which the control depends, as novice programmers encounter many difficulties in the endogenous rather than the exogenous conditions. In addition, the previous mathematical knowledge and knowledge of logic play an important role in understanding the concept of the command under certain conditions.

Research setting

The study we report here is part of a broader research on designing learning-rich settings and "objects-to-think-with" that foster exploration, expression, construction and negotiation. Our aim was to study the potential of using the multimedia programming environment MicroWorlds Pro (MicroWorlds Pro 1.1 - Greek version and the incorporated Logo programming language), as a teaching tool for introducing the basic programming concepts (variables, ways of representing

an algorithm, sequence structure, decision structure and repetition structure) to novice programmers in a systematic way, as well as to push forward some aspects of planning, developing and implementing an effective alternative constructionist didactical approach.

Firstly, we proceeded in a bibliographic research on learning difficulties recorded in relation to our studying object; we defined the cognitive obstacles and set out our teaching goals/objectives. Then, we formed a series of lessons in separate didactic units; we separated each unit in different stages and each stage in distinct steps. Afterwards, we developed a) microworlds in MicroWorlds Pro, b) activity worksheets-lesson plans, c) student worksheets and d) additional teacher worksheets, by making a formative-dynamic evaluation during their development. The research tools used were the above, as well as the Greek version of MicroWorlds Pro environment (version 1.1).

The suggested didactical approach was implemented in the framework of the didactic unit on programming in the Informatics course of the 3rd grade of Junior High of the 1st (4 classes) and 5th (2 classes) Zografou Junior High School in Athens during the school year 2004-2005. The students were separated in small groups of 2 per computer of their own choice. Due to the odd number of students in two classes, there were also two teams with three students. The participants were totally 6 classes of 18 students (10 girls - 8 boys), 19 students (11 girls - 8 boys), 20 students (11 girls - 9 boys), 21 students (12 girls - 9 boys), 22 students (13 girls - 9 boys) and 22 students (14 girls - 8 boys) respectively. The students had already been exposed to the educational software MicroWorlds Pro and had acquired a first familiarization with the environment and the basic commands of Logo language, during the lessons that had to do with the didactic unit Multimedia that preceded the introduction to programming. In addition, before the particular lesson about the decision structure (in the chapter of programming), the students had already been taught the different phases for building a program, the concept of variable, the ways of algorithm description, and had already been exposed to a first brief presentation of the three basic algorithmic structures, as well as to an analytical presentation of the sequence structure. The students' level was mostly heterogeneous. This lack of homogeneity was due to the students' different origin and linguistic competence, as well as to their cognitive level, the basic computer skills and programming skills. Apart from that, in four classes there was a significant 10% percentage of students with dyslexia and general learning difficulties that had to be handled with a lot of attention (e.g. oral examination, special support during the process of coding and debugging programs).

We collected data from the teachers' notes kept during each didactic hour, the students' microworlds, the filled in worksheets and the revision tests of the students, as well as from the semi-structured interviews of students after the conclusion of the teaching process. Then, the data underwent a qualitative analysis, whose results lead to modifications, interventions and changes in the ergonomics, the appearance and the function of microworlds, as well as in the gradual ameliorative reshaping of the worksheets and the lesson plan.

This paper presents the didactical approach for the decision structure, as it was implemented for one didactic hour. The basic investigative questions of the particular study are: a) how the learning environment is shaped during the introduction of decision structure to novice programmers and b) which are the special features of the Greek version of MicroWorlds Pro that contribute to or cause difficulty in the creation of an effective learning environment? It is a case study that uses ethnographic and action research elements.

Teaching goal

The teaching goal of the particular unit is to enable students to understand the concept, the features and the various forms of the decision structure, in order to be able to apply it properly.

Teaching objectives

In terms of knowledge and skills, the student is in position to:

- Distinguish the decision structure from the other basic algorithmic structures.

- Acknowledge the importance of introducing the single and double decision structure.
- Reproduce the two forms of the decision structure in verbal description, pseudo code and flowchart.
- Differentiate the condition in the decision structure and the different values (true/false) it can take, as well as the result according to the condition value.
- Track which commands will be executed according to the condition value.
- Collate the single and double decision structure.
- Develop algorithms and programs by applying the decision structure.

Description of the microworld

The particular microworld, which was designed for the teaching of the decision structure, is composed of 5 appropriately formed pages that attend to the stages of the process described analytically below.

In page 1 of the microworld (see Figure 2), we have a presentation of the flowcharts of the three basic programming structures (sequence, decision, repetition) in their general form. The students can execute the program that visualizes step-by-step the flowchart of the single decision structure (if … then … end if), assign a value to the condition in the dialog box that appears and observe the alternative program flow.

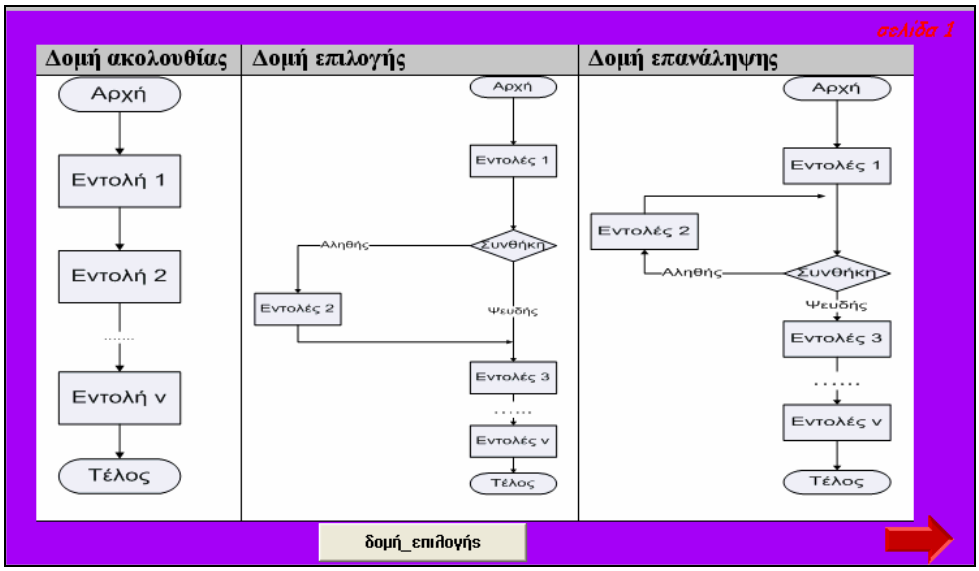

Figure 2: The flow charts of the three basic programming structures (sequence, decision, repetition) in their general form and visualization of the decision structure flowchart (Page 1 of the microworld)

In page 2 of the microworld (see Figure 3), we have a visualization of the three ways in which the algorithm can be represented (verbal description, pseudo code, flowchart), as well as of the Logo code of a specific single decision structure example, which is related with the problem described in Step 2 of the worksheet. The students can execute the program that visualizes the algorithm (in all three possible ways of its representation) and the Logo code's parallel flow of the specific single decision structure example. They can also assign a value to the condition in the dialog box that appears and observe the alternative flow of the program. The execution of this program helps the students to make a connection between the steps-commands that are used respectively in the three ways of the algorithm's representation and the code.

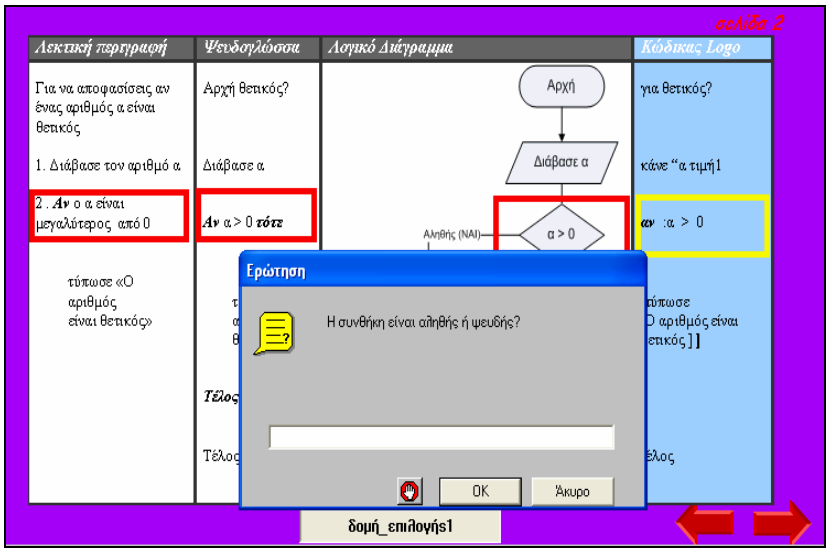

Figure 3: A specific single decision structure example: Visualization of the algorithm and Logo code' s flow and entry of the condition value (Page 2 of the microworld)

In page 3 of the microworld (see Figure 4), the students are asked to assign different values to a variable A, to execute and explore the response of two readily executable programs, positive? and positive??. The students realize the impossibility of applying the single decision structure to problems that require a double branching, they are introduced to the concept of the double decision structure (if … then … else… end if) and recognize its valuable use.

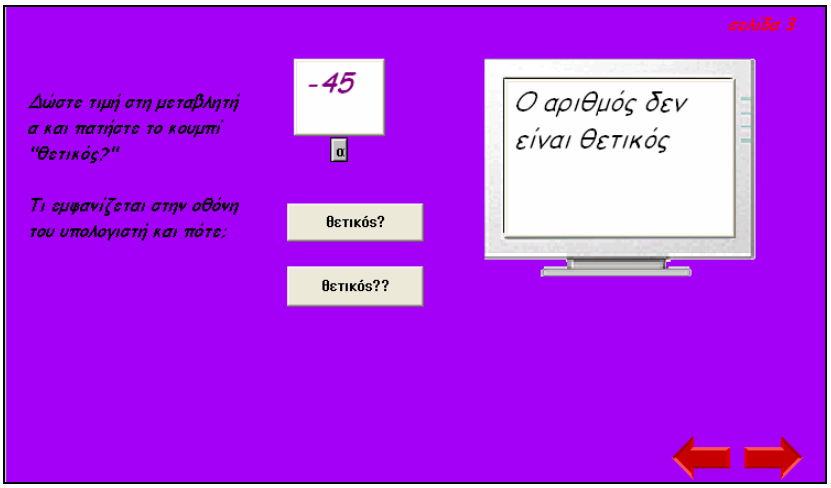

Figure 4: Response of the program "positive??" in case that the entered number is not positive (Page 3 of the microworld)

In page 4 of the microworld (see Figure 5), we have a visualization of the three ways in which the algorithm can be represented (verbal description, pseudo code, flowchart), as well as of the Logo code of a specific double decision structure example, in an analogous way with the presentation of a specific single decision structure example in page 2. The students can execute the program that visualizes the algorithm (in all three possible ways of its representation) and the code's parallel flow of the specific double decision structure example. They can also assign a value to the condition in the dialog box that appears and observe the alternative flow of the program (see Figure 1).

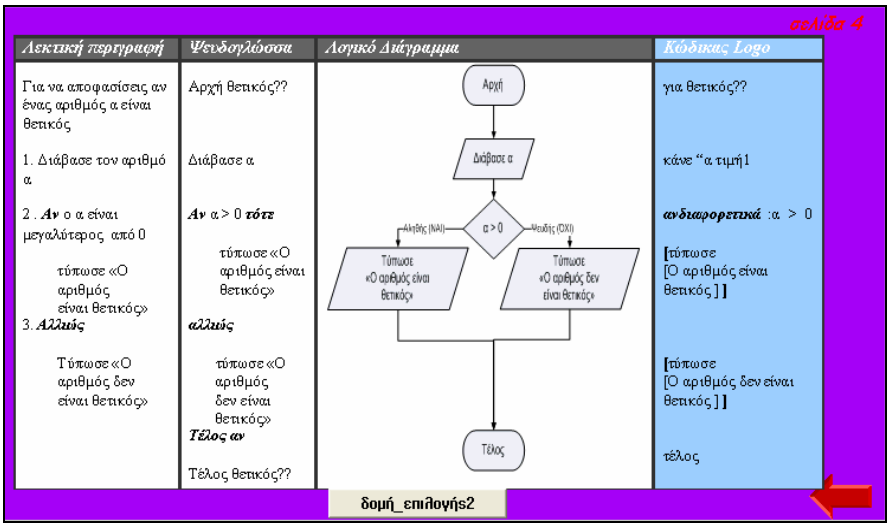

Figure 5: A specific double decision structure example: Visualization of the algorithm and Logo code's flow (Page 4 of the microworld)

In page 5 of the microworld (see Figure 6), the students assign different values to two variables a, b and explore the response of the program that they are asked to build, as described in the 8th step of the worksheet.

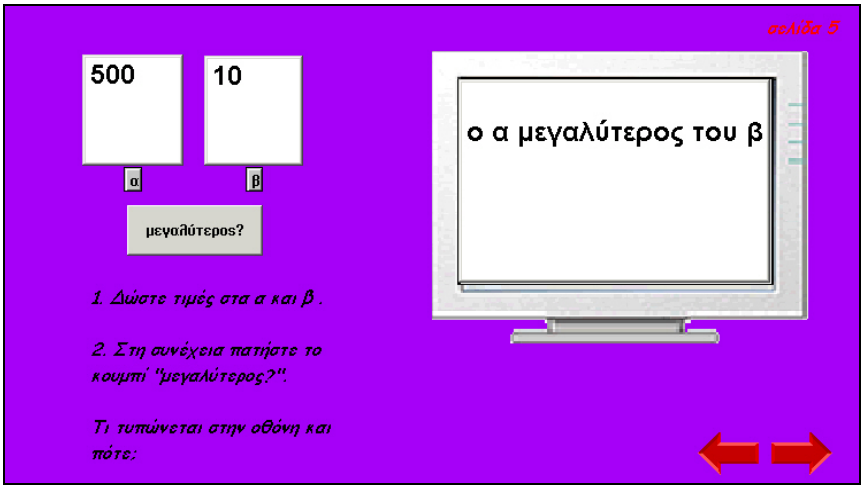

Figure 6: Response of the program "larger?" in case that the entered number a is larger than the entered number b (Page 5 of the microworld)

Lesson Plan

Stage 1 (1st & 2nd step ~5 minutes): Revocation of previous knowledge – Connection with everyday life

In the 1st step, students are asked to tangle with simple problems that include everyday life "dilemmas". On the worksheet, there is a table with statements, which include the concept of the condition, as it is used in our everyday speech, and the students are asked to recognize the condition, to assign to it a value and to predict the result that will occur in relation to this value.

In the 2nd step, the students are given a simple problem, for which they are asked to formulate a solving algorithm in the form of a verbal description. The problem, which the students are asked to solve, is the building of a program that will be able to read a number and, if the number is positive, display the message "the number is positive".

Stage 2 (3rd & 4th step ~10 minutes): Introduction to the single decision structure

In the 3rd step the students are encouraged to open the relevant microworld of MicroWorlds Pro in page 1 and execute the program that visualizes the flow of the logical chart of the single decision structure in its general form.

The students, through interaction with the environment, get acquainted with the general features of the single decision structure and give answers to the relevant questions in the worksheet.

The following are indicative questions of the worksheet: a) What values can be assigned to the condition of the decision structure? b) When are the commands2 executed? c) Are the commands 3…n always carried out despite the condition's value?

In the 4th step students go over to page 2 of the microworld and execute the program of the algorithm and code's flow visualization for the specific example that kept them busy in the previous stage. Afterwards, they answer some questions on the worksheet about the particular example, making clear, at the same time, the features of the decision structure.

Stage 3 (5th, 6th & 7th step ~10 minutes): Introduction to the double decision structure

The 5th step has to do with page 3 of the microworld and the interaction with the "positive?" program, which is given readily executable. The students are asked to give entries of different numbers (positive, negative, zero) in the program and record the program's response, as shown in the relative table of their worksheet. The aim of this step is to introduce to students the concept of the decision structure in its double form (if … then … else… end If) and make them recognize the necessity of its use. After that, we have the 6th step's introductive question, which has to do with the response of the program "positive?". If the inserted number is not positive, the students are faced with the problem of what we would do if we wanted an equivalent message to be displayed in this case.

In the 6th step students execute the program "positive??" that has to do with the solving of a new problem, as it was posed in the previous question, and they are asked to make a deduction of the respective algorithm's verbal description from the result.

In the 7th step the students are asked to execute a specific example of a program using the double decision structure and then answer questions that focus on the structure's features and its difference with the single decision structure form that was presented in a previous phase. The students have to track the common features between the two forms and answer the questions on the worksheet. Some indicative questions are the following: a) How many conditions do we have in this structure (double decision structure)? b) Which values can the condition take here? c) How many junctions do we have in the double decision structure? d) Which are the commands executed in this form and when are they not executed? e) What is the difference between this form and the previous one? When do we use it?

Stage 4 (8th step ~15 minutes): Experimentation - Practice - Feedback

In the 8th step the students are asked to build a program of an analogous degree of difficulty as the previous one, a program which will be able to read two numbers a, b and display their sum if a>b, or else to display their difference.

This problem requires the use of the double decision structure and makes students participate actively in the solving process. Firstly, the train of thought for solving the problem is discussed in the classroom. The algorithm of the problem is given in the form of verbal description on the worksheet. Then, the students, after consulting the microworld, have to translate the algorithm given in a pseudo code and a flowchart and afterwards develop the Logo code, while experimenting in the microworld's environment.

Findings

From the pilot study's data analysis (we present only indicative facts here) we should stress the following:

The students' response to the teaching process was positive. The students remained active throughout the entire course and showed increased interest as far as the interaction with the MicroWorlds Pro environment was concerned, especially when it came to building their own programs. Students that initially didn't show interest in the Informatics course participated actively and showed great concentration in their cooperation with other students. There were five groups that were not motivated, and cooperated only fragmentarily.

In stage 1, the students recognized easily the section of the statements that had to do with the condition, and were easily able to assign a value (false or true), though they seemed to have difficulty in defining the respective result. Figure 7 presents indicative student answers in the table of the worksheet containing statements of everyday life that include a condition in the 1st step.

Statement	Condition	Value *(True/False)*	Result
If it rains, I will take an umbrella.	If it rains	False	I will not take an umbrella.
If it is sunny, I will wear sunglasses.	If it is sunny	True	I will wear sunglasses.
If I study, I will write well in the test.	If I study	False	I will write well in the test.

Figure 7: Indicative student answers on the worksheet in the table of the worksheet containing statements of everyday life that include a condition

The students, while facing the problem that introduces the condition, realized that they can't use the sequence structure and recognized naturally the need to introduce a new structure as a tool to solve problems. The students solve a simple problem, where they naturally and spontaneously introduce the word if, which constitutes the key word of the decision structure.

In stage 2, the students, while interacting with the environment, were particularly motivated as they assigned themselves the value of the condition that they select, determining this way the diagram's flow.

In stage 3, the indicative answers of a student's worksheet as far as the response of the program "positive?" is concerned in step 5 are presented in Figure 8.

	a > 0	a < 0	a = 0
Program's response	It prints the message "the number is positive".	It prints nothing.	It prints nothing.

Figure 8: Indicative answers of a student's worksheet during step 5

In stage 4, the students were enthusiastic after being exposed to the "purely" programming part of the lesson, and admitted having a "live" interaction with the environment. When solving the problem, the students handled with great ease the analysis of the problem, the decision of the appropriate structure, as well as the transfer of the algorithm in its three representations.

The repetitive shift between the code analysis and its reforming/extension, which allows the transfer from simple to complex and the gradual familiarization with the programming language

within the scaffolding process, has proved to be equally effective. During the process of developing Logo code and of debugging, the questions posed by the students were often fixed and imperative. The teacher had to proceed in subtle handlings, in order to accommodate the students with no more support than that they need.

We observed an increased difficulty due to the students' lack of familiarity with the syntactic rules of Logo, often leading to disappointment and urge to quit. The following mistakes were observed on a regular basis: the students left no space between the operators, in the process "name" they left a space between two words (e.g greater 1 instead of greater1), they used the letter "o" instead of 0 (zero), they omitted semicolons before the variable, or forgot to put the word end at the end of the procedure definition.

Although the study was conducted without a control group, the analysis of student answers on the worksheet revealed that (a) students made significant gains in their ability to answer test items covering single and double decicion structure, conditions, condition value and program flow; (b) students in all ability levels showed gains in programming skills.

Follow-up interviews of students revealed that experience with the approach was associated with a reduction in anxiety toward programming, greater willingness to see programming as relevant to everyday life and increased willingness to approach programming challenges with a positive attitude.

Discussion

This pilot implementation in actual classroom circumstances, and, as a matter of fact, in different classes in successive days, offered us important feedback that lead to modifications/interventions to the ergonomics, the appearance and the function of microworlds, as well as in the gradual ameliorative reshaping of the worksheets and of the lesson plan.

The determination of the subject, the content and the functional requirements of the microworld, as well as the creation of the worksheet require deep thought, patience and persistence. The planning of the microworld and of the worksheet defines the setting and affects both the role of the instructor and the students during the teaching-learning process. The application of a shaping-dynamic assessment is necessary and efficient during the development and application circle of both the microworld and the worksheet.

The findings of this trial implementation reaffirm the view that the use of an open software, like MicroWorlds Pro, that allows the development, reuse and adaptation of microworlds, as well as alternative teaching interventions, promotes the growing of creativity and pedagogical freedom of the teacher as well as the active involvement of students.

We need a further analysis in order to generalize the conclusions concerning the features of an effective alternative teaching suggestion, in the level of planning, development and implementation, which responds to the special characteristics of the students of the particular class and of different classes as well.

References

Ackermann, E. (2003). *Hidden Drivers of Pedagogic Transactions: Teachers as Clinicians and Designers*. In Proceedings of Eurologo 2003. Edited by Cnotinfor, Lda. Porto, August.29-37.

Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A., & Miller, P. (1997) *Minilanguages: A Way to Learn Programming Principles*, Education and Information Technologies, 2(1), 65-83.

Dagiene, V. (2003) *A set of Logo problems for learning algorithms*. In Proceedings of Eurologo 2003. Edited by Cnotinfor, Lda. Porto, August.168-177.

diSessa, A. (2000) *Changing minds: Computers, learning, and literacy.* Cambridge, MA: MIT Press.

diSessa, A., Hoyles, C., Noss, R. (1995) *Computers and Exploratory Learning.* Springer - Verlag, Berlin Heidelberg.

Du Boulay, B. (1989) *Some difficulties of learning to program*. In Studying the Novice Programmer, E. Soloway & J. C. Spohrer (eds), Lawrence Erlbaum Associates, Hillsdale, NJ, 283-299.

Ebrahimi, A. (1994) *Novice programmer errors: language constructs and plan composition*, International Journal of Human-Computer Studies, 41, 457-480.

Glezou, K., Stamouli, E., Grigoriadou, M. (2005) *An alternative teaching approach of the decision structure for novice programmers by use of MicroWorlds Pro*. In Proceedings of the 3rd Panhellenic Conference "Teaching Informatics". Edited by Tzimoyiannis, A., Korinthos, October http://www.etpe.gr/uploads1/paper_s33.pdf (In Greek)

Grigoriadou M., Gogoulou A., Gouli E. (2002) *Alternative teaching approaches in introductory programming lessons: teaching suggestions*. In Proceedings of the 3rd Panhellenic (International) Conference "The Information and Communication Technologies in Education", Edited by A. Dimitrakopoulou (Rev.), Rhodes, September, Volume A', 239-248. (In Greek)

Harel, I. and Papert, S. (1991). *Constructionism.* US: Ablex Publishing Corporation.

Komis, B. (2005) *Introduction in teaching informatics*. Kleidarithmos Publications, Athens. (In Greek)

Komis, B. (2001) *A Study of the Basic Programming Concepts in the framework of a Constructive Teaching*, Themes in Education, 2(2-3), 243-270. (In Greek)

Mikropoulos, A. (2004) *Does Logo have a place as a cognitive subject and hollistic model in compulsory education?* In Proceedings of the Two-day International meeting "Teaching Informatics", Volos. 65-72. (In Greek)

Pane, J. and Myers, B. (1996) *Usability Issues in the Design of Novice Programming Systems*, Technical Report (CMU-CS-96-132), School of Computer Science, Carnegie Mellon University.

Pane, J. and Myers, B. (2000) *The Influence of the Psychology of Programming on a Language Design: Project Status Report*. In Proceedings of the 12th Annual Meeting of the Psychology of Programmers Interest Group, Edizioni Memoria, Italy. 193-205.

Papert, S. (1980) *Mindstorms - Children, Computers and Powerful Ideas.* Basic Books, New York.

Rieber, L.P. (2004) *Microworlds*. In Handbook of research for educational communications and technology (2nd ed.), D. Jonassen (Ed.), Mahwah, NJ: Lawrence Erlbaum Associates, 583-603.

Soloway, E. (1986) *Learning to program = Learning to construct mechanisms and explanations,* Communications of the ACM, 29(9), 850-858.

Spohrer, J. and Soloway, E. (1986) *Alternatives to Construct-Based Programming Misconceptions.* In Proceedings of CHI '86, 183-191.

Tzimoyiannis, A. and Georgiou, B. (1999) *Secondary education students' difficulties on the application of decision structure for the development of algorithms. A case study.* In Proceedings of the Panhellenic Conference «Computer science and Education», Computer Instructors Association of Hepirus, A. Tzimoyiannis (Rev.), Ioannina, 183-192. (In Greek)

Vosniadou, S. (2001) *How children learn*, Educational Practices Series, n°7, http://www.ibe.unesco.org/International/Publications/EducationalPractices/prachome.htm