Lifting the hood to see how something works: developing a Logo core as a part of a web-based course for teachers

Evgenia Sendova[1] & Ivailo Ivanov[2]
[1]Bulgarian Academy of Sciences, Bulgaria
E-mail: jenny@math.bas.bg
[2]University of Sofia
Sofia, Bulgaria
E-mail: iivanov@fmi.uni-sofia.bg

Abstract
The paper deals with the educational strategy applied when designing a Logo Core as a part of a course on Logo for teachers, developed in the frames of the European NETLogo Initiative. The idea behind the Logo Core is to enable teachers to lift the hood of the Logo applications to see how they work and hopefully become themselves developers with Logo. Integrating some well known educational principles with the potential of a rich computational environment of an exploratory type in the context of the WWW as the main delivery platform is suggested.

Keywords
Logo, constructionism, learning by exploring, teaching by developing, transparent microworlds

In search of the better path

Many years ago I had joined a group of young mountain climbers who were tackling rocks famous for being a very suitable challenge for beginners. It was possible to reach the top by following a smooth path from the other side. I was on the top in less than half an hour and was waiting for my friends who joined me after several hours of hard climbing. *Didn't you know that there is a more direct path?* I asked them with a smile. But one should have seen their pride and happiness of having overcome the obstacle, to realise how inappropriate my remark was. I still wonder which was the better path...

This story came back to me after reading about the surprise of a famous Logo expert who had given his favourite project, the *simplex lock problem*, to a very fast learner and had expected to see the boy's excitement while struggling with the problem. To the expert's great disappointment, the kid came up with the answer in 5 minutes by just doing a Web search and finding the answer in the **F**requently **A**sked **Q**uestion site...

Some vital problems of education in the context of the information age

"So what is education in the context of the new technologies about – is it just retrieval of information, or publishing information about ourselves? Is the focus on the tools more important than the educational purposes?" And "if the computer use is not driven by a concern for giving students a formal language with which to express, edit and refine their ideas, what does drive its use?" [1]. These are very important questions to consider if we don't just want to change one fashionable key word with another – yesterday *multimedia*, today *World Wide Web*.. After all *life is not about "knowing the right answer...it is about getting things to work!"* [2]. It is not in vain that the representatives of the Logo culture consider the endeavour of "getting it to happen" not just a pedagogic principle, but rather a philosophy of life. In order to feel self-confident citizens of the future our children should acquire skills needed *to participate with understanding in the construction of what is new*, and this aspect of education is what Seymour Papert calls *constructionism*. Ideas in the spirit of constructionism were not alien to the teachers who were masters of the craft of teaching long before the *digital age.* But it is ICT (information and communication technologies) which opened new horizons for working in a constructionist style. Using ICT in such a style is highly recommended in a special document of IFIP and UNESCO [3] based on the experience in countries having used Logo. What is so valuable about Logo is the belief of its creators that *its philosophy was not invented at all, but is the expression of the liberation of learning from the artificial constraints of pre-digital knowledge technologies* [2].

Background

The NETLogo project

This project was designed to a large extent in the spirit of the above considerations – its goal was to promote the use of ICT in the areas of creativity, problem solving and collaborative learning between European schools. Its main focus was to establish and maintain a European on-line presence to support and promote the use of constructionist educational Logo-based tools by teachers in primary school (students aged 7-12 years). The *NETLogo: the European Educational Interactive Site* was designed accordingly [4].

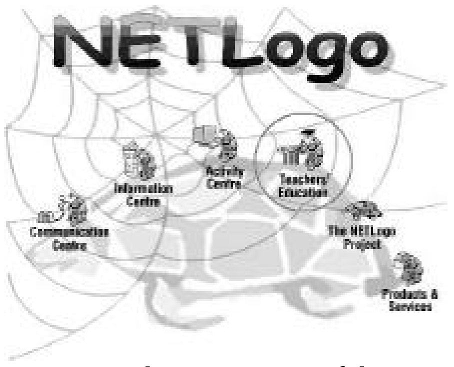

Figure 1: The entry screen of the site www.netlogo.org

As a part of the NETLogo Teacher's Education Centre a training course was developed with the joint efforts of Bulgarian and Hungarian teams together with the NETLogo partners. The main goal of the course was to provide primary school teachers with some Logo applications (microworlds) that could be integrated in various subject areas, on the one hand [5], and with an introductory course on Logo in the context of the Super Logo environment, on the other. Thus the course was designed in two corresponding parts: *Logo applications* and *Logo Core*. Further on, we shall focus on the design and the development of the *Logo Core*, which happened to be an interesting forum for discussions between the authors of this paper as representatives of two generations. (We shall refer to them conditionally as *the old dog* and *the fresh rabbit*).

Through the eyes of *the old dog*

When *thrown the gauntlet* to be in charge of writing the *Logo core*, my first thought was what and how to transfer all the things I found *worked* in my 15-year experience with pupils, university students and teachers. In fact, a similar dilemma existed when writing the first Logo books in Bulgaria [6] – how to transfer the ideas from the pre-digital age which we thought were educationally most valuable in the frames of the integrated subject *Language and Mathematics*. The experimental textbooks were part of a newly developed curriculum (initiated in 1979 by the **R**esearch **G**roup on **E**ducation at the Bulgarian Academy of Sciences) in which the co-study of language, mathematics and Logo-based computer activities played a prominent part. The educational materials developed for the RGE experimental schools included textbooks, teacher guide-books, a bulletin for the teachers and unified computer environments tuned to specific subject domains but still allowing exploratory activities in a broader context. Thus students could pass gradually from constructing controllable models (Lego-Logo), through various problem-oriented microworlds to a fully programmable microworld for explorations in Euclidean geometry – *Geomland* [7]. Such an educational strategy created the feeling of *continuity* and facilitated the integration of every new bit of informatics knowledge with the experience already acquired. The researchers involved in the RGE experiment realised that what was still needed was to involve teachers in an interactive, creative and learner-centred educational process. A Logo Centre providing support for practising teachers was established in 1993 at the University of Sofia. The following year two projects were initiated in the framework of a national program: *Learning by developing with Comenius Logo* [8, 9], designed for the regular and elective informatics classes at middle and lower secondary school level, and VALUE – an interactive on-line WWW resource related to the scope of the first project [10]. Various Logo-based courses were included in pre-service and in-service teacher education at the University of Sofia. The future teachers were given a broader background of Logo programming and involved in project-based activities so as to better understand and experience the difference between *knowing something* and *being able to do something*. In 1997 a new textbook *Informatics in Logo style* [11] appeared at the request of the Ministry of Education. It was designed as a Logo-based informatics book for students from 5th to 10th grade in different forms of training at school: regular, optional, and extra-curricula activities. In this textbook a serious attempt was made to bridge the gap between the two extreme approaches in using computers at school: *programmers versus users*.

Through the eyes of the *fresh rabbit*

The move from earlier versions of Logo to more recent developments (like Super Logo and MicroWorlds) has been a shift from a language emphasis to an immediate-mode visual emphasis. In short: *Programming in Logo is fun, programming in Super Logo is super fun!* – this was my impression when involved in the work on translating *Super Logo* in Bulgarian. It possesses the power of animation, colour, multiple turtles, direct manipulation, and multimedia effects as vehicles for projects. This implementation of the Logo language makes use of all the powerful features of the Windows environment and provides complete support for turtle geometry, Logo data structures, multimedia, multiple turtles, complex and easy graphics handling. It offers a strong and flexible developer's laboratory for exploring, creating and playing. Several new features, new data structures and a balanced combination of a direct manipulation interface have been added within a rich Logo programming environment [12]. In my practice as a teacher in a specialised mathematical school I experienced a kind of a paradoxical situation: 12-year-old students developed projects which looked almost like professional applications thanks to the properties of this environment and managed to impress much older students, already advanced programmers in Pascal. Some of the lessons were attended by pre-service teachers. They were amazed to realise that a lot of things that they had done by means of the language could be carried out so easily by tools of the environment. Only now they were convinced that even advanced Logo features could be accessible for young kids.

The design of the Logo core

Figure 2: The main entry screen of the Logo core part

The initial idea behind the *Logo Core* was to meet the curiosity of those teachers who would like *to lift the hood to see how* the Logo applications *work*. Approaching the design of the Logo core from two different aspects (the emphasis being on the language versus the emphasis on the programming environment) we still had our chance as a team – *the fresh rabbit* was aware of the power of the language, and the *old dog* was not reluctant to *learn the new "environmental" tricks*. Finally we did it in such a way that the users would be able *to voyage* between the interface and the programming language and to do this with a learning goal in mind. The course is structured in six modules. We shall consider them from two aspects: the **informatics topics** (supposed to cover the basics of Logo as a language and an environment), and the **projects** they were introduced to.

Direct manipulation in Super Logo

Informatics topics

- Basics of Super Logo Environment.
- Creating new turtles in a dialogue mode.
- Working with multiple turtles.
- Creating turtle's shapes with the Image Editor.

Projects

- Some basic instructions for controlling the movement of the turtle are presented by buttons, together with a special MiniDemo button, which gives an idea about the potential of the environment in terms of the characteristics of the turtle, text animation and sound.
- Several turtles with the shape of cars are created to participate in a motor race. A further version of the motor race is with cars whose wheels are turning.

Turtle graphics

Informatics topics:

- The basic turtle graphics commands.
- The notion of *cycle*.
- Arithmetic operations. Variables. Metaphors.
- Defining procedures.
- Procedures with inputs. Tail recursion.
- Defining operations.
- Top-down programming.

Projects:

- Drawing stylised letters, carpets, regular polygons, houses, figures consisting of circles and arcs, and more complex structures (including self-similar ones) using the above figures as building blocks;
- Calculating arithmetic expressions directly and step by step to check the priority of operations;
- Designing, drawing and running an electronic chronometer.

The users start working with the language by entering commands to control the relative displacement and turning of the turtle. They draw figures by using various characteristics of the turtle pen. The way arithmetic operations are presented in Logo is shown and their execution is visualised by *function machines*. How variables of text and image type are defined is demonstrated by means of the language and the environment alike. The same applies to the way procedures are defined, edited and debugged.

Procedures with inputs and tail recursion are introduced as a very convenient way to describe fractals. The style of top-down programming is illustrated in the design of an electronic chronometer.

Absolute graphics

Informatics topics

- Controlling the turtle in a Cartesian way.
- Absolute directing of the turtle.
- Creating turtles by means of the language.

Projects

- Drawing stylised pictures (given as a set of points specified by their Cartesian co-ordinates) and editing the procedures for drawing the pictures so as to get their symmetric images are presented.
- Directing the turtle in an absolute direction or towards a specific point is introduced in the context of a project in which the turtle plays the role of a space-pilot travelling from star to star.
- Generating multiple turtles by means of the language is demonstrated through the well-known *four-bug problem*. This problem is rich in variations – various types of behaviour are simulated in terms of human relations, or according to the laws of physics.

Control and data structures

Informatics topics

- Logic expressions. Conditional instructions. Predicates.
- Words, lists and images. Operations on them.

Projects

Simulating the behaviour of various creatures: turtles, bees, etc. depending on their location.

Generating a Character sketch in which the computer tells you what kind of person you are (your name appearing as an acronym) based on a list of words for human qualities – for each letter of the name a word from the list starting with that letter appears.

A more systematic presentation of the logic of Logo and its control structures is presented. The pursuit of making the logic expressions more readable motivates the introduction of user-defined predicate operations. The data types in Super Logo are systematised together with the operations on them. The connection between the words and images as sequences of characters and frames, respectively, is demonstrated in the context of a project generating special graphic effects with the user's name. With the introduction of lists, the project is developed in producing a user's character sketch.

Interactive programming

Informatics topics:

- Programming the communication by keyboard.
- Programming the communication by mouse.

***Projects*:**

- Transmitting messages by using the Morse code of the key pressed on the keyboard.
- The Eater game, in which the player can control a greedy creature by the arrow keys.
- Creating a Graphic editor for drawing in *free hand style* by means of the mouse.

Options for choosing the colour and the width of the pen, the pattern of its trace together with automatic drawing of geometric figures are presented by means of buttons. The project is further developed and enriched by assigning the functions of the buttons to turtles with shapes of buttons.

More advanced multimedia features

Informatics topics

- Working with sound and music.
- Multimedia extensions of Super Logo.

Projects

- Creating a virtual piano, in which each key is a turtle playing a note when pointed by the mouse.
- Creating multimedia cards with animated elements on a music background (by means of a Wave file).

Teaching with Logo

Informatics topics

- Tips for adjusting the environment according to the needs of the teachers.
- Analysis of Logo both as a programming language and as an educational philosophy.

The new role of the teacher equipped with a computer, that of Columbus, daVinci or Prometheus [13] is discussed based on the experience in Bulgaria and other countries. Some lessons learned from successes and failures are conveyed.

The structure of the lessons

Summary – an idea (in a jocular form) about what is to be expected in the unit.

A **table** of the new notions, concepts, primitives to be learned, and activities to be carried out *hands-on*.

Task setting – a lesson presentation in a step-by-step, *learning by doing* style.

The tasks are to be considered in the context of a larger project.

Self-checking – easy problems, aiming at reassuring oneself that the main notions, syntax rules etc. have been acquired.

Activities – a chain of problems with increasing complexity, which are: further developments of the lesson's tasks and suggested practices for mastering the content.

While the tasks are in fact digested so that the teachers could accumulate self-confidence, the activities are the source, which would hopefully give rise to many powerful ideas.

Topics for reflection – relatively short sections on what has been learned in the lesson, how it relates to the rest of the course, how it can be extended, etc.

Educational principles

There is no threshold

When the users enter the course they take part in training games in which they can control one or more turtles by pressing only a few buttons. (The fact that they might lack keyboard skills has been taken into account). From the very first lesson they grasp a general impression of what can be done with Logo (rather than *what Logo can do for them...).*

Introduce instructions only when needed

Or, as others put it, programming should not be a goal, but a tool [14]. This strategy is followed throughout the course with the belief that mastering informatics tools can be creatively linked with working on projects rich in gaining knowledge from various domains. But in order to appreciate a new tool, the user is expected to have experienced some constraints when using the old ones (e.g. the REPEAT command and the notion of a procedure are introduced after the user has experienced the boredom of having to execute the same sequence of commands many times). Propaedeutics of new notions and primitives (e.g. referring to the pen's characteristics) is done in the *Activities section* to accomplish more intriguing designs in an exploratory style.

Programs are never right or wrong

They are rather artefacts that could be developed, refined, enriched, modified. In harmony with this principle many problems are solved in various ways. More elegant and compact solutions are often based on the introduction of more complex informatics tools. In addition, many of the projects are developed in *step-wise refinement and enrichment* style, sometimes in several consecutive lessons.

A proper Logo style is cultivated

Since Logo was designed as a tool for learning, many of its features – interactivity, modularity, extensibility, flexibility of data types follow from this learning goal. We encourage a style of programming in an extended vocabulary of Logo, which is very close to what sounds natural for the modelled microworld. The idea behind such a style is to make a program as *readable* as possible, on the one hand, and to suggest a general structure of the main procedure, which might be completed in various ways, on the other. Our hope is that when working in this style the teachers and their pupils will cultivate a feeling of belonging to a team – the work of one should be readable by others so as to be further developed. *Top-down* and *bottom-up* programming styles are demonstrated in the context of projects that have been experienced previously (e.g. *Chronometer* and *rose-windows*, respectively) [6, 14]. The importance of making the procedures *transparent by state* if they are to be used further as building blocks is demonstrated. Based on numerous examples the readers are encouraged to approach a problem in various ways and to find or build themselves the most appropriate tools. The co-ordinate turtle is introduced relatively late so as to support cultivating a more geometrically consistent programming style in which absolute and relative positioning of the turtle are not mixed [11]

"To become a good carpenter one should participate in carpentry"

Creating complex graphic designs with relatively simple tools is one of the most attractive features of Logo and many tasks and projects make use of it. Drawing beautiful graphic designs is combined with some exploratory activities and logical problems hopefully attractive for children. One of their favourite activities is playing games, but implementing games is much more challenging. The recognition that profound learning comes from designing and building games [16] is reflected in many of the projects (e.g. *Motor race, Eater, Character Sketch, etc.*).

Errors are welcome

In learning, errors are very often more valuable than a single right solution. First of all they help us to become aware of our thinking process. Moreover, it may well be that the approach of *debugging* is one of the most profound educational ideas of the 20th century, since perceiving mistakes as an opportunity to learn might be transferred to other activities inside and outside school. We try to convey the message that teachers should not be afraid of making mistakes themselves. For this purpose, we sometimes allow mistakes to slip in, in order to show how to cope with such situations. Errors are often the source of new ideas and goals. Sometimes debugging goes together with *de-goaling*. The environment is very friendly when it comes to finding and correcting errors. The error messages are very informative and, something very important from the psychological point of view, they make the computer (not us) appear *the stupid one*...

The readers learn by doing and invent by exploring

They are expected not just to read but also to construct and check the solution of the tasks. Most of the lessons are accompanied by additional resources: pre-developed projects and configuration files for the user to run, explore, modify and extend. Exploration activities are introduced even at the beginning of the course. Playing with various values for the parameters helps the users feel like researchers – some of the activities include special procedures for making experiments and finding patterns. Working in exploratory styles is well supported by the environment – the history of any activity is memorised in a special window so that the user can copy all the successful steps in his explorations and name this sequence appropriately as a procedure.

There are 3 types of languages for expressing ideas

Connections between 3 types of languages (a computer one – Logo, mathematics, considered as a formal language, and a natural language – English in this case) have been sought throughout the Logo Core. They are illustrated in most Logo applications, too. The idea behind such integration is that if students are fluent enough in a formal language they could build computational models of phenomena that interest them and learn not only about those phenomena but also experience what is essential in terms of scientific ways of thinking. Behind the clicking there is also a language. Pressing buttons is accompanied by the text-generation of the corresponding instructions. This conveys the message that Logo *is* a language and that it is natural to talk to the computer in a language very close to a natural one.

There are projects for every taste

The extensibility of Logo enables the users to define objects and operations specific for different fields: drawing, calculations, music, animation, languages. One and the same notion is illustrated in various contexts (e.g. the notion of *cycle* is introduced to draw repetitive patterns, to model a metronome, to *teach* the computer to count, to model the sound of a cuckoo). Even in the frames of a single project there is a place for various talents to be harnessed – those of programmers, designers, writers, painters, animators. Based on the informatics tools in the course projects, the users could build their own projects according to their interests. Thus the old cliché 'informatics is mainly for boys' will be overcome once teachers realise that the same informatics tool kit may be used for different scenarios. Furthermore, some cross-cultural connections are promoted in the course – e.g. the turtle draws stylised letters from the Cyrillic alphabet, it creates Greek ornaments, it plays the role of a tourist guide in Europe.

Help in need is help indeed

The help system of Super Logo provides very convenient information about the semantics and the syntax of the primitives of the language together with examples of the way they are used. In addition, this information could appear in the process of developing a project just when needed, thanks to fast key combinations. Another important type of help, especially for novices who find it difficult to remember the names of primitives, is provided by a kind of a *first aid* window – it contains a list of commands and operation used most often. The teacher can update this list so as to tune it to the class activities. It could be used not only when one has forgotten the name of a primitive but also for exploring and discovering the functions of new primitives. There are a lot of additional options for tuning the environment in such a way that the user could concentrate on the project rather than think about syntax and spelling. Among them are the options for colouring the primitives in a procedure, the possibility of working with the so-called *choosers*, providing different graphics and textual tools, hints and menus. The novices who would often forget to provide inputs to a command are not delivered the traditional message, but are offered instead special instruments for measuring distance and angles, a palette of colours, or dialogue boxes to fill in the text inputs. Thanks to the Web platform various links are provided:

- links attaching a glossary item to a Logo primitive or a concept;
- links to previous lessons;
- links to microworlds in which the main idea (or tools) could be seen in a more ambitious, professionally accomplished program thus becoming *transparent* for the user.

In addition, some Logo applications turned out to be quite helpful as a source of tools for the very lessons, e.g. the *Function machines* microworld is used to illustrate the way Logo interprets arithmetic operations, the *Co-ordinate microworld* is used to draw stylised pictures presented as sets of points. Last, but not least, the users are helped in making products that look professional: thanks to the possibility of loading scenery-like pictures and turtle shapes made by professional painters, even the first attempts to produce something with Logo are very appealing.

Editing does not necessarily mean correcting errors

Editing images and procedures brings a very important educational idea, viz. that the process of editing very often consists of modifying an existing object. In the context of animation, the consecutive phases of the movement of an animated object are easily produced by editing the last existing frame to get to the next one. Editing procedures with similar functions are a natural step towards the definition of a more general procedure. In a broader context, the editing of a text, images and musical fragments is very often related to creating variations of a certain theme, which is considered to be *the crux of creativity*.

Believe only half of what the computer says

Developing awareness that the computer's calculations should be interpreted correctly has been sought by means of various examples (e.g. the difference of two numbers which are close enough could be zero or not depending on the precision chosen). Thus the belief that using computers will eliminate the necessity of arithmetic skills should be replaced by an understanding that a shift is needed from what is known as "parrot mathematics" towards a better ability of estimating the results.

The style of presentation

As in the prototypes of the course [6] the style of presentation is that of easy conversation. The authors have tried to create the impression that they are sitting next to the users, carrying out experiments and making discoveries with them. Jocular titles and cartoons are added with the hope of creating a special atmosphere of treating the environment as a nice place where a friendly creature is ready for your commands and ideas. Some educationally powerful informatics metaphors have been visualised and brought into life through an appropriate interface:

Project memory containing the global variables and the procedures of the project;

Turtle's identity card – a dialogue window with the main characteristics of the turtle;

Library of images, which are potential turtle shapes.

The users are expected to get the feeling of becoming partners of the authors on every single task – *smiling* hints provide various approaches to the solution. The idea that *in order to make a good enough software product a profound informatics knowledge is needed* runs like a red thread through the course.

The target audience – the teachers

The course is built on the basis of a language-based exploratory environment that provides a real laboratory for testing and editing ideas, a workshop for building tools appropriate for creative self-expression. One could hardly doubt that such open-ended computational environments enhancing the constructionist style of learning would be a valuable component of the best education systems of the future. Based on our experience that most teachers prefer *creative* to *easy* we believe that teachers and learners would immerse themselves in the learning process by finding enough time to play with the projects in the lessons and with the microworlds. Even more – that they would explore their own ideas for modifications in depth. The modular structure of the course will hopefully meet the requirements for usability at different levels of programming knowledge and skills. Teachers who would like to cultivate in their students a spirit of inventiveness (necessary for the workers of tomorrow) should teach them how to analyse problems that have no predetermined answer and come up with creative solutions. And in order to convey best this inventiveness they themselves should learn by making new things.

Conclusions

Today we are at a stage when the young generation encounters computers out of school and provokes us, the educators: *Can't you do something more interesting for us?* This in turn changes the situation for the teachers, who instead of asking: *Why should we do this?* will hopefully ask: *Why shouldn't we try this*?

Now they have the choice of doing things *on Monday* that are directed towards *one day*. Most of them are mature, knowledgeable and experienced enough to realise that the real teachers are life-long learners, and not only that – they should be brave enough to partly share the learning process with their students. This makes us feel a part of a broader team endeavouring to improve and further develop what is offered by the course. Our great hope is that when reaching the end of the *Logo core* the reader will be convinced that learning Logo is not an end in itself, and *it is the journey that is the main reward*.

References

1. Goldenberg, E.P. (1999) Bringing Back Formal Language: A Use to Counter My Worries about Computers in the Mathematics Classroom. Sofia: *Proceedings of EUROLOGO'99*, 50-61.
2. Papert, S. (1999) What is Logo? Who Needs It? LCSI Inc: *Logo Philosophy and Implementation*.
3. *Informatics for Primary Education* (1997) Recommendations produced by UNESCO, IFIP and INTE, http://school.edu.ru/int/rec_inf.pdf
4. Sampson, D., Patsouras, P. & Terzopoulos, V. (1999). NETLogo: The European Educational Interactive Site, Sofia: *Proceedings of EUROLOGO'99*, 306-316.
5. Turcsányi-Szabó, M. (2000) Subject Oriented Microworld Extendible environment for learning and tailoring educational tools–a scope for teacher training (this volume).
6. Nikolov, R. & Sendova E. (1985) Informatics textbooks for beginners, Sofia: *Proceedings of the Conference Children in the information age*.
7. Sendov, B. & Sendova, E. (1995) East or West – GEOMLAND is best. In diSessa, A., Hoyles, C. & Noss, R. (eds.) *Computers and Exploratory Learning*. Berlin: Springer–Verlag.
8. Blaho, A., Kalas, I. & Tomcsányi, P. (1996) Comenius Logo, Bratislava.
9. Blaho, A. & Kalas, I. (1998) *Learning by Developing*. London: Logotron.
10. Nikolova, I. (1997) Towards VALUE – A Virtual Almanac for Logo Users and Educators. Budapest: *Proceedings of Eurologo'97*, 240-248.
11. Dicheva, D. et al. (1997) School Informatics in Logo Style: a Textbook Facing the New Challenges of the Bulgarian Informatics Curriculum. Budapest: *Proceedings of Eurologo'97*, 243-239.
12. http://www.edi.fmph.uniba.sk/logo/
13. Sendova, E. & Sendov, B. (1993) Columbus, Da Vinci or Prometheus: A New Role for the Mathematics Teacher in a Computer Environment. *Journal of Technology and Teacher Education*, 1, 2
14. Ivan, I. (1997) Programming as a Tool, not as a Goal. Nessebar: *Proceedings of PEG'97*, 398-399.
15. Ivanov, I. (1997) Top-down and Bottom-Up Along The Christmas Tree. Budapest: *Proceedings of Eurologo'97*, 432-434.
16. Ivanov, I. (1994) MATHLAND – an idea for teaching informatics through "transparent" software. Sofia: *Informatics for Secondary Schools–Today and Tomorrow,* 119-123.

Lifting the hood to see how something works: developing a Logo core as a part of a web-based course for teachers. Article (PDF Available) Â· October 2011 with 32 Reads. Cite this publication.Â The paper deals with the educational strategy applied when designing a Logo Core as a part of a course on Logo for teachers, developed in the frames of the European NETLogo Initiative. The idea behind the Logo Core is to enable teachers to lift the hood of the Logo applications to see how they work and hopefully become themselves developers with Logo. Integrating some well known educational principles with the potential of a rich computational environment of an exploratory type in the context of the WWW as the main delivery platform is suggested. The second part of the chapter focuses on a supportive co-curricular programme. Chapter 3 looks at the attribute of reflection and the related concept of learning how to learn. It explores the concept that schools need to have a learning, rather than a performance, orientation with reflection at its heart.Â This needs to be recognised so that students learn to see making mistakes as a learning opportunity and not something to be feared. This is considered in Chapter 5. The learner attribute of engagement is particularly important.Â One reason teachers need to generate their own schemes of work and lessons plans (see the Developing your School with Cambridge guide, Chapter 4: 9 Developing the Cambridge learner attributes. Chapter 1 continued.