Learning of Dynamic Data Structures – Having Fun with Algorithms

Ingrid Nagyova, *ingrid.nagyova@osu.cz*
Dept. of Information and Communication Technology, University of Ostrava

Abstract

One of the basic abilities students specialized in subjects focused on information technologies should cope with, is algorithmic thinking. However, we must admit that algorithms' training seems to be rather difficult for students therefore they may become unconcerned.

This article is an attempt to illustrate that algorithms training may be interesting and inspiring for students. They actively participate in the classwork; they seek inspirational tasks and their possible algorithmic solutions. When teaching dynamic data structures, we apply constructivist teaching methods. At first students primarily learn to create algorithms without using any programming languages.

The scope of dynamic data structures is relatively demanding. It requires a good understanding of computer memory organization. On the other hand, no extensive knowledge in programming languages is required, not even with most demanding projects.

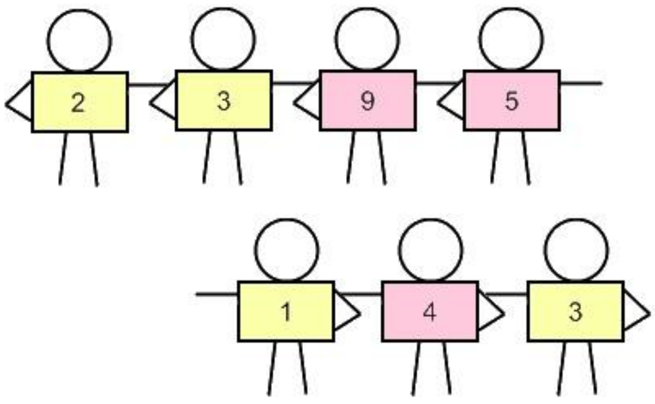

Figure 1. Students Form Dynamic Structures

At the beginning, each student puts one hand on his/her hip (this hand creates a linking loop (linking structure) and the other hand - the empty one - is "pointing" at nil. Getting linked by loops, students form trains. When the rules for work with "student trains" are defined, a lot of potentialities stimulating students to solve various tasks are created, for example a uncoloured train, an organized train, etc.

Keywords

algorithms; pointer; dynamic data structures, stack, queue

Learning Algorithms and Programming

The Pedagogical Faculty of University of Ostrava (Czech Republic) educates future teachers and professionals for all types of educational institutions. The field of study Information Technologies in Education is focused on the preparation methodologists and ICT coordinators on primary and secondary schools. During the course of the bachelor's degree study the students of that field complete four examinations focusing on algorithms and programming:

- basic algorithms and programming
- complex algorithmic structures – including the teaching of dynamic data structures
- Imagine programming
- object-oriented programming.

Teaching algorithms is one of the main curriculum topics of study fields focused on informatics, as well as the field Information Technologies in Education. Information technologies are based on the idea of algorithms. Just a few years ago, every common computer user had to make a practice of programming. Even today, this ability is current for experts working with information technologies.

Strategies and methods of programming and algorithms training are dealt with in quite a few scientific subject publications. In spite of all efforts that are made to apply these recommended methods, algorithms and programming teaching experiences difficulties. Demanding factor and the most abstract level of the subject-matter reflects into students´ lack of interest, and they tend to minimize their efforts to minimum level necessary for scraping through the exam, etc.

This contribution is based on experience gained in our courses on dynamic data structures, where we try to apply the constructivist teaching methods. First of all, students learn to suggest and to create algorithms without using any programming language, just through movement and activities in the classroom. Subsequently, they learn basic orders which are going to be employed in practical problems solution.

The scope of dynamic data structures (Wirth, 1986) is relatively demanding. It requires a good understanding of computer memory organization. However, as far as students cope with the subject matter, they will understand basic principles of computer architecture. Moreover, if the high demand factor and abstraction of the subject-matter those students perceive at the very beginning as an undefeatable barrier is compassed, it incites their enthusiasm. Students become involved in the study and develop their creativity in further research. And it is also included in the title of this paper – Having a Fun with Algorithms (Futschek, 2007).

Learning Dynamic Data Structures

Dynamic data structures' training consists of several coherent items:

- pointers
- linear dynamic structures and linked lists
- non-linear dynamic data structures.

We are going to concentrate on the first two items.

Pointers

The term "pointer" (as the index into the computer memory) corresponds to the Czech expression "signpost". Our university is situated at the foothills of Moravian-Silesian Beskids Mountains, and there are many marked tourist pathways crisscrossing one another in the surrounding area. The project is based upon these pathways, leading a passionate tourist from place to place (see figure 2). Project "Wandering through Moravian-Silesian Beskydy Mountains" has been developed from an original idea of students and thanks to their initiative. They created it in Imagine.

Over the project, students ponder on the term "pointer", and together we try to solve a range of questions: What stands behind the term "pointer?" What information will this signpost carry through? What shall we gain if we follow the message on the signpost?

A pointer is denoted by an arrow. The pointer carries information on the reference (address) of a certain place in computer memory, where a certain value can be stored. Further, teacher can point with his/her right hand (pointer) to individual students (values), and thus call them to come to the blackboard. On this call, each student can point with his/her right hand to another student, or s/he can even get linked to him/her – catching his/her left arm (see figure 1). Left arm represents the linking loop in this game, it is forbidden to use it for any other work.

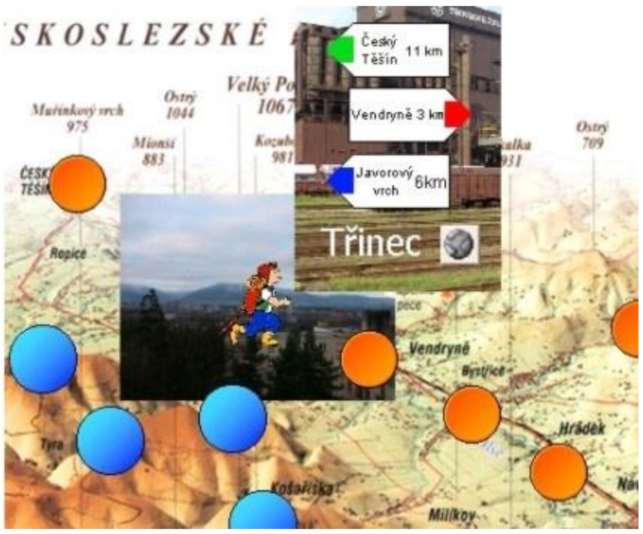

Figure 2. Project "Wandering through Moravian-Silesian Beskydy Mountains"

There are some other rules of the game:

- students, who are not held by anybody, must be given a name,
- each one is allowed to be holding at most one of the follow students,
- it is allowed to unbend from the follow student´s loop only in acute cases,
- if the person who is holding my arm moves, I must move with him/her.

Getting linked by loops, students form one or more trains. Then, "student trains" attempt to solve various practical tasks. They can, for example, couple on a "wagon" – a student from another train - either to the head or to the rear of their train. They can insert a student into the middle of their train. They can even cancel the whole train. More difficult tasks come after, for example reversing the train, whereas the first student becomes the last one - i.e. the one who is not held by anyone – or other tasks can be solved, e.g. reordering the train according to the colours of students´ T-shirts, or reordering it according to their height, etc.

For homework, we often use books or CDs instead of trains (see figure 3). These are to be stacked on one another. The upper CD (like the first student at the head of a train) must always be given a unique name, the name being labelled on it.

Through solving the tasks with CDs, students gradually familiarize with the problem and learn to work in the simulated environment of the defined game. Then, there is just one step to go to the definition of work with linear linked lists.

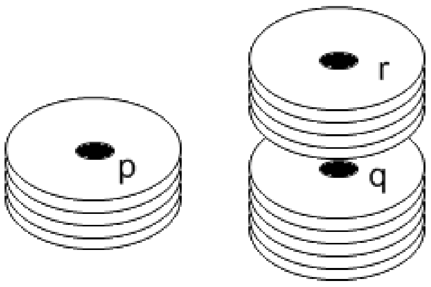

Figure 3. Tasks with CDs - Shifting and Labelling CDs

Linear Linked Lists

The ability to solve the tasks assigned in the environment of "student trains" game can be developed in a software simulator (see figure 4). It has been created again by students in Imagine. Individual "wagons" are represented by rectangles with a load - an integer value. Pointers are denoted by dots with arrows.

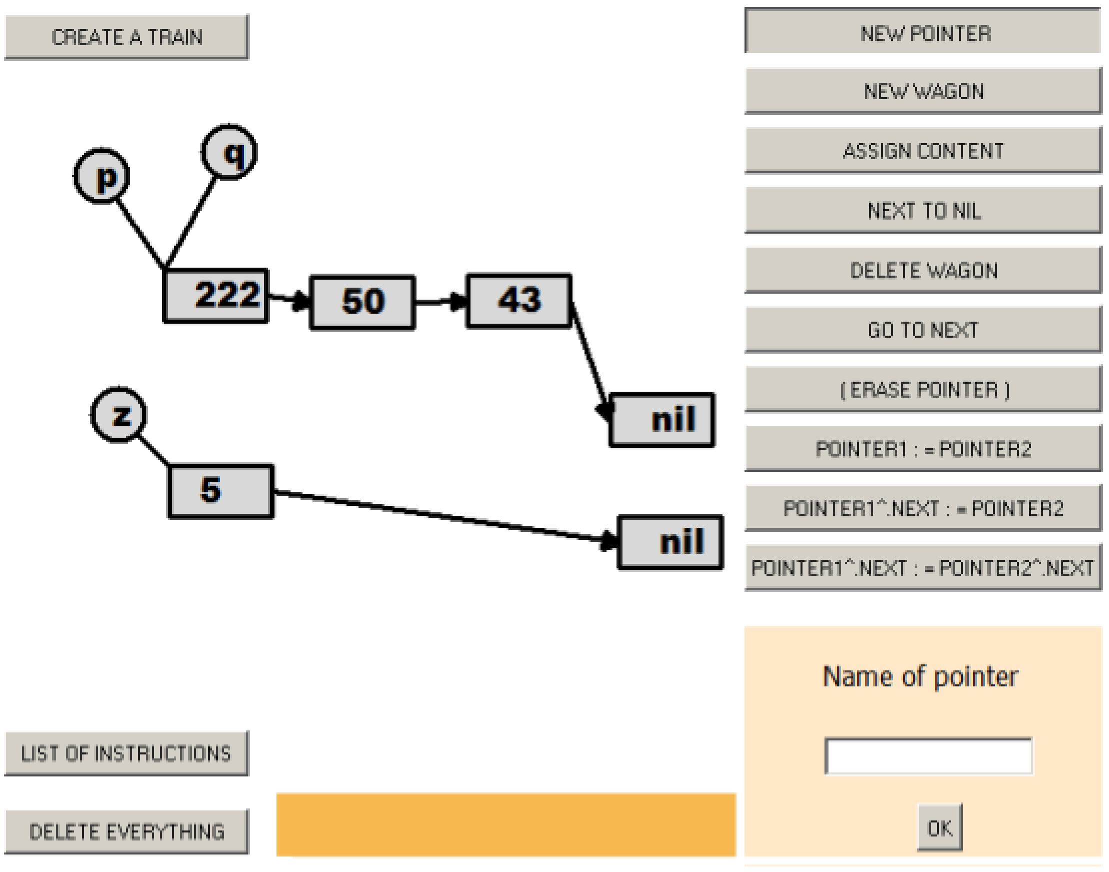

Figure 4. Software Simulator of Linear Linked Lists

Simulator provides:

- creation of new pointers,
- creation of new wagons which are assigned a value,
- deleting wagons, erasing pointers,
- shifting the pointer to the next wagon in the sequence,

- resetting (setting to nil) the value of the wagon pointer,
- mutual assignment of two pointers.

Tackling problems in a simulator is more complicated, since pointers are used as the names of individual wagons. Students are forced, in contrast to the preceding activities, to work with the names of wagons dynamically, to consider their creation and cancelling, to shift the wagons from one to another, etc.

The work of the student in the environment of a simulator is registered and written down step by step in the form of concrete orders of programming language into the computer memory. Whenever the student likes, s/he can look through the record of actions, which were made in Pascal programming language, and if necessary, it can be copied into the development environment of this language.

These way students learn to know linear linked lists and how to work with them. They are getting to know, that it is necessary to set the pointer of the last wagon of the train to nil. They learn that the wagon, which is not pointed at by any pointer, is irretrievably lost. They discover simple and more complicated algorithms for work with linear lists, which are coming up. As soon as these strategies are mastered, we can focus on particular dynamic structures - stack and queue.

Stack and Queue

"Historical architecture reminds us of lives of people and of long past events. Some of the oldest monuments of history are the buildings in Egypt, extant up to the present day. One of those is a well-known temple in Abu Simbel, which had to give way to the construction of Aswan Dam half a century ago. It was resolved that the precious complex of the temple would be moved over a few metres higher." These words are an introduction to the stack and queue theme. The huge complex of the temple had to be cut up into blocks, moved to a new place and put together again. Cars, waiting to be loaded, were standing in a line (queue). First of all, the blocks from the upper part of the temple, which had to be cut up first, were transported. However, it must have been the bottom blocks which had to be placed on the new place primarily. That is why the upper blocks were temporarily stored at a transfer area. The transfer area comes to be a stack.

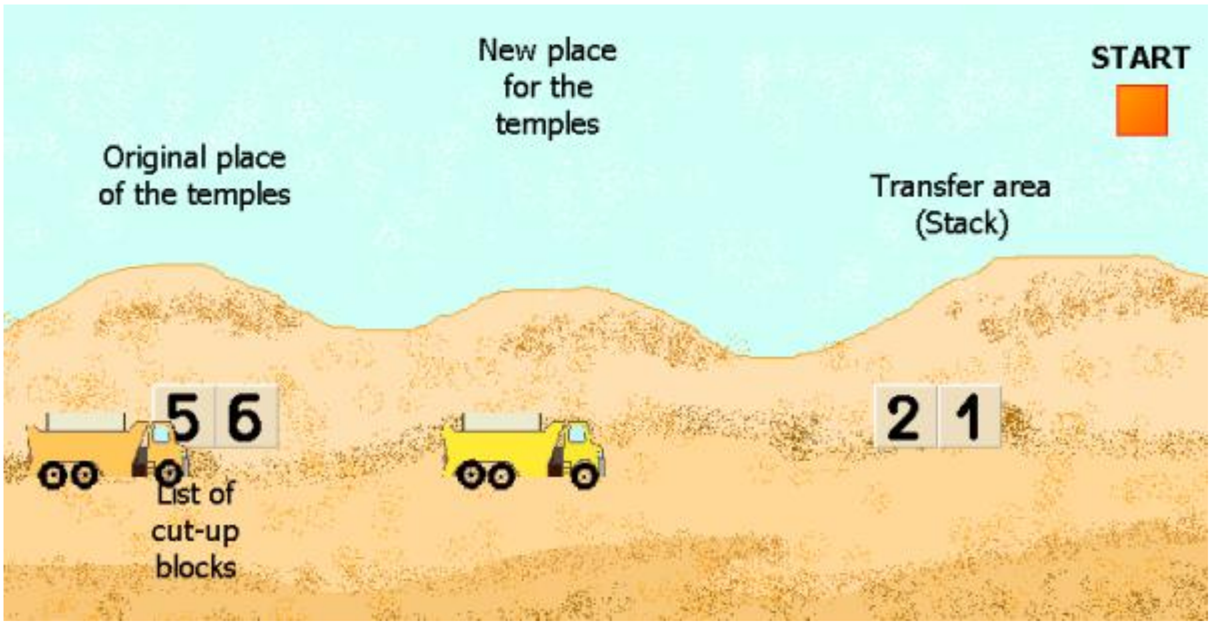

Figure 5. Transfer of the Temple in Abu Simbel

This story demonstrates a model created in Imagine (see figure 5).

Let us come back to the thought of "student trains". Students are lined up in one train and their task is to form a new train, formed by the same "wagons" (elements) in the same sequence as the existing train. We can choose only the first student from the existing train: This student becomes the first wagon in the new train. To solve the problem, we must use a transfer area again, where the train, made of the same wagons, is reversed. There are two steps for handling the task (see figure 6).

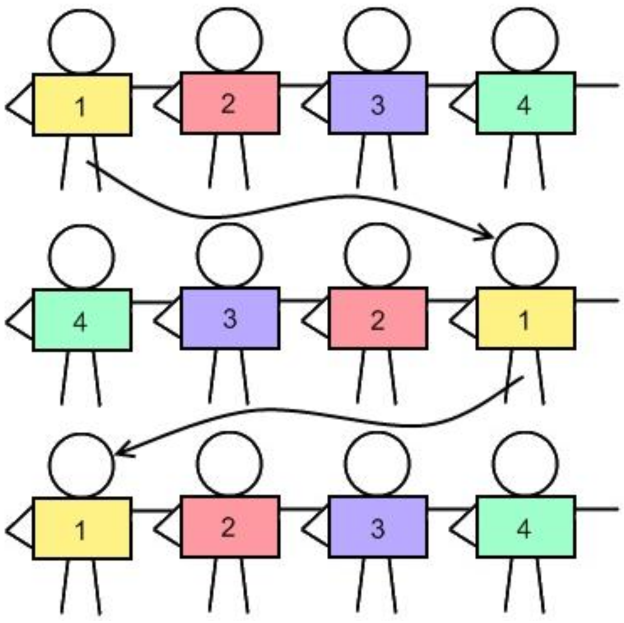

Figure 6. Transfer of Students into a New Train

Students learn the main principles of the stack and the queue operations through solutions of these model situations. Subsequently, they become able to distinguish and to define explicitly these structures. At the same time they come to know the corresponding programming codes by means of the described software simulator.

Conclusion

The scope of dynamic data structures is undoubtedly one of the most demanding. Dealing with algorithms seems to be most difficult to students. Curriculum tends to become tedious and arid for them, in particular on the grounds of the high extent of algorithmic abstraction.

Yet, experience and practice exemplify, that if we find a suitable presentation method of the given subject matter, students can master even complicated abstract themes. An advisable technique seems to be that kind of a teaching process, where students become active participants in class work activities simulating a microcosm with its rules, and students gradually gain required knowledge. The effect of such teaching method is even increased at the moment, when students, having an opportunity to participate in the activities, become a part of the game; they become one of the "wagons" of the "student train."

The field Information Technologies in Education only about 40 students pass out in full-time and combined forms yearly. The results of teaching therefore can be so difficult to evaluate quantitatively. The students' increased interest, discussions, creativity, as well as the focus of the themes of bachelor theses unambiguously confirm the positive evaluation of the described teaching methods, The process of teaching dynamic data structures is based on a yearlong cooperation with students, who, incited through our methods and techniques, brought new ideas, came with the view of enriching the learning process. Many of the above ideas and mentioned procedures are the result of students´ work. I would like to give acknowledgment and thanks to all of them. The educational aids were created in the universal educational environment Imagine (Kalas and Hrusecka, 2004).

The described teaching methods help students understand the structure and running of algorithms in action with dynamic variables. The students learn to design further algorithms and described their running. However, they are unable to use these skills for working with dynamic data structures in a particular programming language, for programme creation. These skills aren't included in the students' curriculum; we plane to find appropriate teaching methods in the future.

References

Futschek, G. (2007*) Logo-like Learning of Basic Concepts of Algorithms - Having Fun with Algorithms*. In Proceedings of EuroLogo 2007. Edited by I. Kalas. Bratislava, August. pp. 51.

Kalas, I. and Hrusecká, A. (2004) *The Great Big Imagine Logo Project Book: projects, worksheets and starting points.* Logotron, Cambridge

Wirth, N. (1986) *Algorithms & Data Structures.* Prentice-Hall, New Jersey