Task Development for Discovery Learning of Informatics Concepts

Philipp Prinzinger, *philipp.prinzinger@tuwien.ac.at*
Institute Information Systems Engineering, TU Wien, Vienna, Austria

Gerald Futschek, *gerald.futschek@tuwien.ac.at*
Institute Information Systems Engineering, TU Wien, Vienna, Austria

Abstract

We describe our own learning process while developing tasks and scaffolding materials for discovery learning of informatics concepts. At our university we perform nearly every day a workshop with school classes of secondary schools. The main aim of the workshops is to involve the school children in interesting informatics activities that make fun and keep in mind for a longer time. We decided to perform mostly activities without use of computer to give more emphasis on thinking about informatics concepts than on information technologies. We quickly found out that a constructive way of learning that allows detection and re-invention of informatics concepts is much better appreciated by the students than presentation and explanation of selected informatics concepts. We present an exemplary development of a task and its scaffolding materials that support the learners in their discovery process.

We used a prototyping approach with very short development cycles. The first year of workshops performance shows not only essential changes in the offered tasks but also a rapid evolution of the general workshop format. All changes aim to better fit the ideal goal to achieve active, discovery learning of all individual students with use of constructive and tangible materials. The evolution of the error detecting activity is described in detail.

Figure 1. Different versions of learning material for the "error detecting" task

The learning materials changed while we learned how to pose the problem in such a way that the kids can better explore situations so that they find themselves solutions to the problem.

Keywords

Task development, discovery learning, informatics concepts, scaffolding, secondary schools

Introduction

In January 2019 we started at our university an outreach program for school classes in the field of informatics. We offer workshops for students from grade 5 to 12 (age 9 to 18). The goal was to involve students in informatics-related activities to show the beauty und fun of solving problems informatically. It is not only the goal to attract students to study informatics but mostly to propagate the idea that informatics is a creative science that allows problem solving that makes fun and can be applied to other topic and disciplines. The learning situation is different to a regular learning unit at school. The students come accompanied by their teacher to a place that is new for them. There an university teacher and some tutors await them and give them some tasks to learn more about informatics. A typical workshop lasts 90 Minutes and there is no follow up planned so far.

Our workshops are structured in 4 parts: Part one is the introduction, in which we welcome the classes and give them a short overview of the topics and the workshop itself. This takes about 10 minutes. In part two we start with our icebreaker activity that last 20 minutes on average. This is a simple common activity guided by the workshop leader, in which all participants are involved. Then we move on to the third part, the main activities in small groups. Here the students can solve different tasks within 45 minutes. In the last 15 minutes of the workshop we summarize the key messages in the fourth and last part.

So this learning intervention is extremely short compared to regular courses. There is also no grading of students involved. Students may explore given situations and discover properties, rules and even algorithms. There are no correct and false solutions. Important is a personal engagement and fun if something useful is detected. Since the time of the workshops is relatively short the situations and activities must be carefully planned and proper supporting material must be prepared. As we focus on informatics concepts learning and due to the short workshop time we provide mainly unplugged activities. So we have not to spend time for technology related issues.

When we started with our workshop programme in January 2019 it was not clear how to properly state the problems to the students to reach our goals. We had at that time just installed the exhibition “Abenteuer Informatik” (adventure informatics) (Gallenbacher 2012) in the newly restaurated aula of the informatics building. So we were tempted to use this exhibition for our goal. The exhibition comprises 44 large boards covering 11 informatics themes: Maximum Flow Problem (see figure 2), Travelling Salesman Problem, Binary Numbers, Ethiopian Multiplication, Codes, Sorting and Searching, Parity Bits, Limit of Computation, Decision Problem, Binary Search, Sorting Networks. The exhibition boards offer interactive tasks, but also solutions and a lot of informative text to the corresponding informatics themes.

Figure 2. Maximum flow problem: A tutor explains a hands-on task on a board of the exhibition

From constructionist learning we know that we should pose interesting problems and provide the students with a proper environment in which they can construct and try their own solutions. Scaffolding is allowed but offering the solution is not.

Related work

Webb and Rosson (2013) describe experiences with an outreach program where computing activities for school girls involve discovery learning.

In the last decade emerged many unplugged activities for learning informatics concepts, most well-known is “CS Unplugged” (Bell, T. et al., 2009) from where we took also the idea of the “error detecting” activity, its further development to discovery learning is explained later in this article. Bell & Lodi (2019) argue that these unplugged activities are constructive and involve in this sense constructionist elements.

In their article Weigend et al. (2019) classify creative unplugged activities in 4 categories: find an algorithm, find an application, find an example and find a visualization. Also Futschek & Moschitz (2010) argue that inventing and playing algorithms can be done with groups of school students where all of them are involved in a creative learning activity. In our workshops we also try to apply creative learning and in especially inventing algorithms.

Workshops for school classes

Since the “Abenteuer Informatik” exhibition offers opportunities to hands-on experiments that deepen an informatics concept for each of the themes, we started to use the exhibition boards for our workshops. We call our first approach the “exhibition workshops”, since we used the exhibition boards to work with small groups of students.

In a short first phase we guided school classes, divided into small groups, through the exhibition. Our plan was to enable the students to work out the presented informatics concepts individually or in groups using the didactically high-quality exhibits. Apart from the above-mentioned challenges, there was also the problem that the exhibits did not allow any alternative variations to the intended concept. If the students did not develop exactly the expected results, they had to be guided to it, which did not correspond to our goal of a more individual and more independent learning.

The exhibition workshops

A typical school class of 24 students was divided in 4 groups á 6 students. Each group should explore 4 themes of the exhibition and was assisted by a tutor who explained the tasks related to the exhibition boards that should be solved using the interaction possibilities explained on the boards. But unfortunately the number of students directly interacting at an exhibition board is due to limited space 1 or 2 students (see Figure 1). So the other students are just observing instead of active interacting. To involve all students the tutors were tempted to explain the informatics concepts in detail instead letting the students discover the main ideas of the concepts themselves. This led to a more teacher-centered learning, where too many technical terms were used by the tutors, which led to a mental overflow of the students and a loss of attention. The different experimentation times of different student groups working on different exhibition themes confronted us also with another problem: additional waiting time when moving from one theme to another.

Figure 3. The “error detecting” problem presented as a magic trick on an exhibition board. A group of students is analysing and testing the concept by arranging and flipping the magnetic cards

To overcome this additional problem, more topics than groups were included in the workshops and the interaction time per participant was limited. Although these modifications were able to reduce the problems described above, it was still not in line with our ideal of constructionist learning. From the very beginning we tried to design the workshops in such a way that as many students as possible could become active and participate. Our vision is to give all students the opportunity to develop their own solutions and concepts. This ambitious goal could not be achieved for all students when working in this way.

Exploratory Workshops

To involve all students of a group in exploring a concept at the same time, we created additional tangible materials which were related to the themes of the exhibition. With these materials the students could work largely independently or with brief explanations of the tutors. The materials became tools and not, like the exhibits, a source of information. In this way we performed a change from guiding groups of students to a phase where we allowed each of the students individual analysis of each concept that led to a deeper understanding of this concept.
But there was still a major issue: the tasks and materials led to only one informatically perfect solution. We identified two problems for achieving our goals: little freedom for creative solutions and a still rather reproductive learning. By aiming at a certain concept, there was little or no room for own solutions and concepts. If the one solution was not found, the tutors had to present it. This led to purely reproductive learning and possible frustration. Although we were able to enable more students to explore the solution and research independently, there were still some limitations. The students may re-construct in their brain the concept but they do not construct a concept themselves.

Constructional Workshops

Therefore we moved to a third phase of our workshop format, where the students could develop the main ideas of informatics concepts themselves. The specific evolution of the workshop formats and their three stages of development are described in detail in the following section using as example the “error detecting” task, that involves the informatics concepts error detection and error correction using parity bits.

Development of scaffolding materials for discovery learning

The development of the error detection and error correction task took its origin in the activity "Parity magic" from CS Unplugged, which is similarly included at one of the boards of the “Abenteuer informatik” exhibition (figure 3). In its original form presented in the exhibition, the activity consists of the demonstration of a magic mind reading trick presented by a tutor who plays the magician. The students are given 25 magnetic cards showing a black cross on a yellow background on one side and a white circle on a blue background on the other side. The students have the task of creating a graphical information, represented by a grid of 5 by 5 cards. Through adding one row and one column of cards by the magician under the cover of making the information more complicated, this grid is extended to 6 by 6 (see figure 3). Now it's the students' turn again: they have the options either to flip any card and thus change the original information, or to do nothing and leave the information untouched. While the students may manipulate the information, the magician turns away with closed eyes and then claims to find out with the power of his thoughts, if and which tile was flipped. Because the magician has added the cards according to a special but secret system, he is always able to determine whether a card has been flipped, and he even can tell exactly which card has been flipped.

After the demonstration the secret was revealed: the 11 additional cards were added in a way, error detection and correction is made in computers. Also the term "parity bits" was introduced. Afterwards it was the students' turn to try out the presented concept for themselves.

We redesigned the task for our workshops several times: In the first variation the students were given a random pattern of the two sided magnetic cards on a 5x5 grid. They were asked to design a concept, how they could place the remaining 11 parts, in order to recognize if a card was flipped and ideally also which card was flipped.

The students didn't see this as a big challenge, as most of them were able to remember the given pattern on the rather small field. This reduced the motivation to consider an appropriate error detection or correction concept. Apart from the low motivation, we had the problem that almost none of the students came up with the “right” concept on their own. So it was always necessary to supervise the students by a tutor who led them to the solution.

In order to increase the motivation of the students and provide them more freedom in exploring, discovering and developing the concept of parity bits, we made another variation. We created “message boards” (as shown in figure 4) with yellow and blue bits on a 5x5 grid. But now one of the bits was not fixed. The loose bit was yellow on one side and blue on the other. The students were asked to put this loose bit with the right colour into the grid. They got the hint: The L-shaped grid can help them to solve the task. This situation should enable the students to analyze the pattern and to recognize that the L-shape creates an even number of blue and yellow parts in all rows and columns. Subsequently, the recognized system could be used to create parity bits for other square patterns of different sizes.

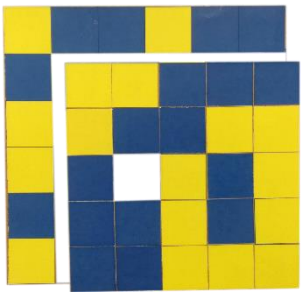


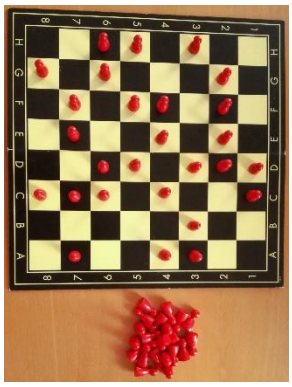
Figure 4. Message boards with added parity bits (L-shape) and one missing bit. The task is to find out the systematic behind the coloring of the bits of the message board. And finally also the color of the missing bit is asked. (The parity bits are such that the number of blue and yellow bits is even in every row and every column. So the missing bit has to be yellow.)

At this stage of development, the students were able to do some research and discovery, but could not discover a system for error detection by themselves. So we had not yet arrived at a fully discovery learning task. The goal was to find materials to engage the students to develop their own solutions and strategies to notice, if anything in the original information (pattern) will be changed. In the sense of Tissenbaum, Sheldon and Abelson (2019) these kind of activities do not have such a long-term impact compared to concrete implementations, but we are convinced that developing, testing and presenting one’s own creative solution has a great impact on motivation and learning behaviour.

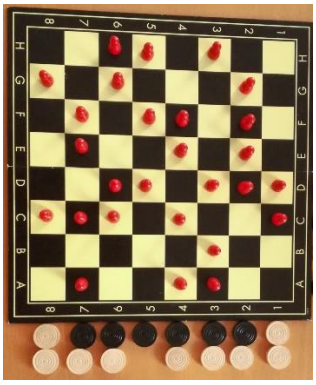
We reached the final step towards discovery learning of the concept with help of a chessboard, some game figures and checkers.

A chessboard as scaffolding material

Subtask 1: students are asked to place some game figures randomly on a chessboard (see figure 5a). Subtask 2: the students should develop a concept for placing the checkers (next to or even on the chessboard) to be able to detect if a single game figure will be added or removed or if nothing has been changed. Subtask 3: they should use the placement of the checkers to determine which game figure has been added or removed. Subtask 4: a placement strategy should be developed using a maximum of 16 checkers.

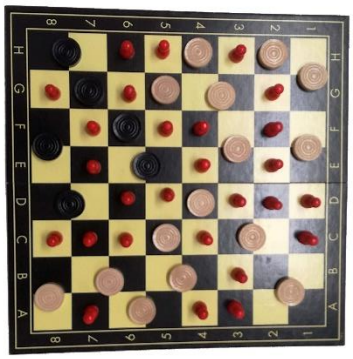


a. Subtask 1

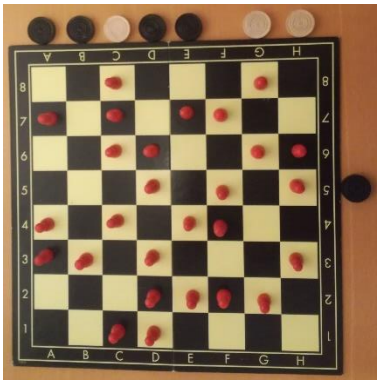


b. Subtask 2

c. Subtask 2 & 3



d. Subtask 2 & 3



e. Subtask 4

Figure 5 a-e. Students' solutions of the subtasks 1-4 (before any changes were made)

The scaffolding: The task described above introduces students to the intended concept of error detection, in particular the subtasks 2, 3, 4 for determining whether something has been changed or not. The first subtask serves to create an initial situation, that could be interpreted as a 2-dimensional code of a message. If a single error occurs during transmission of this message, this error should be detected with help of the checkers. Students' ideas, concepts and suggested solutions are discussed with the tutors after each subtask, e.g. by the tutor pointing out possible advantages and disadvantages. The tutor also explains the connection between the overall task and the computer science concept of error detection and error correction, including the representation of a message by the arrangement of the chessboard. In the second subtask only a single error (adding or removing of a single token) should be detected. It is not yet relevant to localize the error or to use as few checkers as possible.

Figure 5b shows a students' solution to identify whether a game figure will be removed or added before the tutor has made any changes. The checkers are used to count the number of game figures in each column. To reduce the number of checkers needed, the students used the black checkers for three game figures and the white ones for one.

In the third subtask the position of the error should be discovered too, the number of checkers is still limited just by the provided number of checkers.

The solutions in 5c and 5d have a similar approach for error detection and error correction: marking fields. While in 5c all occupied fields are marked with a checkers underneath the game figure, the method in 5d marks all unoccupied fields. The students who designed the solution 5d had too little checkers to mark all fields, so they positioned the checkers on common edges or corners of neighboring fields to mark more fields with just one checker.

Through the markings an error (adding or removing a game figure) can be detected and corrected easily. If a game figure is removed, the error is obvious: either an empty square remains without a marker or only the marker remains. Adding a game figure is also immediately obvious: either it is on an already marked square after being added or the marking below it is missing. A decisive difference between the two methods lies in the informatics feasibility: for the method shown in figure 5c some kind of overhead could be used to store, that this memory unit contains data; although the basic principle of the method in figure 5d is basically feasible, its presented design cannot be implemented informally.

Both methods have in common, that they won't work in reality. Although the method in figure 5d can detect some errors in practice, since all data (game figures and checkers) would be lost in the case of a faulty memory block or message block, error correction becomes impossible in both cases.

In the fourth subtask the number of checkers used for any placement of the game figures should be as low as possible. Students are informed that a maximum of 16 checkers can be used, for any number of figures and for any placement on the chessboard, if only one game figure is removed or added.

The solution given in figure 5e is a representation of the parity bits. Through the placement of the checkers outside the chessboard, one additional line and one additional column is added. In every row and column with an uneven number of game figures a checkers is placed next to the chessboard. In the extended grid every row and every column now has an even number of pieces. If the game figure on the field D5 would be removed, the simulated error can be determined and corrected easily. Row D and column 5 both contain an uneven number of pieces, ergo there must have been a game figure on the square D5 before any changes were made.

Through this kind of scaffolding the students can develop their own concepts, test them,discuss them within the group and with the tutor and refine them step by step if needed.

Discussion of findings

In each phase of the development of our workshops we designed them with close attention to our goals under consideration of the given circumstances. During the performances of the workshops, we often found that the workshop design did not allow us to fully achieve all of our goals. Through constant observation and reflection we were able to identify concrete problems and make adjustments.

To make our findings comprehensible, we first describe our goals and then the actions, observations and adjustments of the respective development stages.

Although the workshops have changed over time, our goals have remained the same throughout: to inspire students for informatics, to show the wide variety of informatics (informatics is more than programming), to spread the joy of informatics, to offer a high degree of interaction, to provide impulses for informatics teaching.

In order to achieve these goals we used at the beginning the interactions of the exhibition boards in the exhibition workshops and a wide range of different topics of the exhibition, which were presented by the tutors with enthusiasm and excitement.

With this tutor-centered approach we could observe that the limitation of the interaction possibilities to one or two persons led to disinterest of the uninvolved persons. Since the tutors presented the contents and transferred knowledge, it was essential that each group was supervised by a tutor. This meant that we had a high personnel effort and hardly any possibilities to shorten the waiting time for the students. Due to the constant attendance and presentation by the tutors, the motivation to find the solution for a given task was rather low. The students often preferred to ask the tutor if they could not solve a problem quickly. When the solution was explained, the secret was revealed and the desire to try it out for themselves was rather low.

In order to address the observed problems of passivity and tutor-centeredness, we developed additional materials based on the exhibition and our own tasks, which can be worked on more independently by the students.

The discovery workshops covered fewer topics, each with several tasks. The information from the exhibition could be used by the pupils to work on the tasks. This meant that the tutors no longer had to do the entire knowledge transfer.

With these changes, we could see that the students dealt with the tasks more intensively. Due to the additional materials, several students could work on one task at the same time. In the small groups, discussions on the topic of the task were held and more joint work on a solution was carried out. For some tasks we had to find out that the pupils only worked on them superficially or not at all. The students said that the tasks were too difficult for them or that they had not yet learned this particular topic.

By varying the tasks we found out that the complexity was one reason, but also aiming at a certain solution increased the difficulty of a task.

After we found out that the additional materials had a positive influence on the students' motivation, we created further tasks with our own materials, which went beyond the scope of the exhibition. With these changes we arrived at the current stage, the constructional workshops.

The new open-ended puzzle-like tasks offered a lot of creative freedom. The students could carry out and try out their own ideas. They spent more time on the tasks and showed a deeper interest in the topic. At these student-centered workshops we observed a lot of fun and instead of a variety of topics we had a variety of solutions.

References

Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, *13*(1), 20-29.

Bell, T., & Lodi, M. (2019). Constructing Computational Thinking Without Using Computers. *Constructivist Foundations*, *14*(3), 342-351.

Futschek, G., & Moschitz, J. (2010). Developing algorithmic thinking by inventing and playing algorithms. *Proceedings of the 2010 Constructionist Approaches to Creative Learning, Thinking and Education: Lessons for the 21st Century (Constructionism 2010)*, 1-10.

Gallenbacher, J. (2012). Abenteuer informatik: hands-on exhibits for learning about computational thinking. In *Proceedings of the 7th Workshop in Primary and Secondary Computing Education* (pp. 149-150). ACM.

Tissenbaum, M., Sheldon, J., & Abelson, H. (2019). From Computational Thinking to Computational Action. *Communications of the ACM,* Vol. 62, No. 3, 34–36.

Webb, H., & Rosson, M. B. (2013, March). Using scaffolded examples to teach computational thinking concepts. In *Proceeding of the 44th ACM technical symposium on Computer science education* (pp. 95-100). ACM.

Weigend, M., Vanicek, J., Pluhar, Z., & Pesek, I. (2019). Computational Thinking Education through Creative Unplugged Activities. *Olympiads in Informatics*, Vol. 13, 171–192.