16/1 (2018), 1–23
DOI: 10.5485/TMCS.2018.0438
tmcs@science.unideb.hu
http://tmcs.math.unideb.hu

Teaching
Mathematics and
Computer Science

CS Unplugged in Higher Education

Kinga Kovácsné Pusztai

Abstract. Nowadays, there is a significant lack of workforce in the IT industry, even though it is one of the most lucrative professions. According to researchers' forecasts, the existing shortage is growing, so the wages offered will be higher, yet it seems that young people are not attracted to the profession. This problem draws attention to the need to change the curriculum so that it can attract students more. One possible solution is to supplement the curriculum with *CS Unplugged* activities, which makes it easier to understand and deepen difficult concepts and make IT lessons more colorful. In my article, besides presenting the already known CS Unplugged activities, I will deal with how this can be applied in Hungarian higher education as well.

Key words and phrases: computer thinking, education, CS Unplugged, search, recursion, binary numbers, information theory.

ZDM Subject Classification: D40.

Introduction

Many countries are struggling with the lack of IT professionals, despite high salaries [4] and good employment opportunities [4]. In Hungary, for example, nearly 22.000 IT personnel are missing [3, 4, 6, and 15]. A New Zealand research [7] investigated the wages of 73.000 jobs and found that the profession was the best paid: four of the top five paid jobs were computer scientists. While labor demand is increasingly growing in the IT sector [2, 3] and more and more professionals require IT knowledge [2], the number of people entering higher education [2] and the number of IT graduates [3] is decreasing. According to researchers, this will result in an informatics workforce shortage of between 750.000 and 1 million in the European Union by 2020 [3, 6]. This is also dangerous for us because the IT workforce shortage of Western European countries appears to have a real extraction effect in Hungary.

Another problem is the disappearance of women from the IT industry, even though it would be in need of them [3]. According to a research [7], from the 1970's onwards, women's participation in IT professions was the highest in the 1980s and is now the lowest. All these problems are not new. According to a research [7], between 2000 and 2004, in many Western countries, the percentage of students attending IT studies decreased by 60 percent. The situation of women is even sharper; in the same period, the number of girls enrolling in university education fell by 80 percent.

The reason behind these contradictions may be the too large and theoretical curriculum [7, 16] and the poorly explaining teacher [7, 16], which causes the students to become unmotivated and lose their interest in the subject [15].

Although the number of recruitment points has increased steadily and significantly in ELTE in recent years, the tendency in European countries is typical of all IT training in Hungary, the number of IT students dropped from 15.000 to 10.000 between 2001 and 2014 [4], and women disappear here too from the field. According to the 2012 situation analysis [1], ”Students in PTI BSc in Hungary do not perform the required studies very hard within the prescribed time. On average, 10% of incoming students can meet the requirements in six semesters. This is due to the fact that even the final semester is a full semester, and completion of the thesis is a serious challenge even for the best students. The building of the subjects on each other is a further deterrent, so the average length of time is between 9 and 10 semesters. ... There is also a high drop-out rate, in case of the BSc degree it is about 40% over the 3 years.”

All these examples show that both in public education and in higher education there is a need for change in education, for which a possible direction may be the use of CS Unplugged in addition to the general description of CS Unplugged, I would like to give you specific ideas on how this method can be used primarily in higher education. I have already taught in elementary school and I have trained informatics teachers, currently I am teaching IT programmers, so I have tried some of my ideas. I am going to introduce these ideas, how I could use them in teaching and how the students responded to them.

My aim is to present a method that helps students understand and deepen difficult concepts, helping them to solve these problems. In my article, I would like to highlight the benefits of the methodology, the simplicity of its application, and its positive effects on students, and so they wish to use it.

CS Unplugged general description

To solve this problem, an unusual approach was developed at Canterbury University, called CS Unplugged. This means activities without the use of a computer designed to help students better understand certain concepts and wake up their interest. Their research has been published on http://csunplugged.org/, with the aim of using, commenting or further developing teachers. On this page, you can download 20 different CS themes for free, which can be used in classroom situations. Of these, 15 are completely computer-free and translated into several different foreign languages. These lessons primarily focus on the age groups of public education. Some of the lesson plans include video, which illustrates its application. For the sake of learning, they also inform and encourage teachers as part of a workshop. Teachers' feedback is very positive, and we believe that this form of work would encourage students' interest and improve the quality of teaching. They also attracted interest from teachers who teach other subjects, who also recommend this method. In the USA, due to the centralized curriculum, there is a great interest in private schools, but in some countries they are used as part of the IT curriculum [17]. In addition, it has been used in many places as part of IT competitions and camps where the participating students (girls) also have a positive opinion about it.

CS Unplugged is effective because the computer is not a subject of learning, it is only a possible tool and knowledge is created when a device or game is used. It is not that Unplugged excludes other activities, but rather it focuses on the aforementioned. During its application, students are more active in the classroom and deepen understanding of certain concepts and processes.

By using CS Unplugged, you can talk about a variety of subjects at a lower age, as it does not require technical experience.

The advantage of this is that there are many activities that students cannot test in the real world. For example, some experiments those are too expensive or too dangerous. These are a so-called second life, that is, the online, virtual world can do it.

CS Unplugged themes, as they do not require a computer, are non-stationary or can be used in the open air. We can work with larger groups and diminish the costs of a lesson. Due to these positive features, developing countries are willing to apply and participate organically in their development.

CS unplugged and programming connection

At Canterbury University, an attempt was made to find out whether programming helped to deepen the knowledge acquired by CS Unplugged. For this, seven interested high school students (of 17 years) were selected to attend the "STAR" course at Canterbury University. All seven learners had knowledge of some Java programming elements (cycles, arrays were known). During the course they only learned programming for 3 months. Their knowledge of Java was at a level of a student who had learned programming for two years in high school.

The experiment took place within the STAR course. The students first met with the topic of Occupation 4, which deals with error detection and correction. They played card flipping, which requires a stack of cards of different color on both sides. In the beginning, select a student who draws the cards into a 5x5 box so that the cards are mixed between upward and downward facing. Then add a line so that each row and column contains an even number of downward facing cards. Then ask a student to flip a card while we close our eyes and we figure out which card has been flipped. (The searched card is the one for which the number of flipped cards is odd.) After playing the game and deepening the concept, the "Pre test" took place in which 6 assignments were given where there was only one error and 6 other tasks with 0, 1, or 2 errors. They had little time to solve the tasks, so speed and precision could be measured. In the third part of the experiment, the programming tasks were performed. Students could choose from 3 levels of difficulty. At each level 9-10 examples were given, from easier to more difficult levels. Finally, the experiment was completed by a "post test" like a pre test, as well as a questionnaire, similarly with short timeframe.

The result of experiment can be seen in Figure 1. During the experiment, each student had an average increase of 2.6 points in their performance. In addition, the number of incorrect answers given to the second test was reduced, which led to the conclusion that their precision increased. Based on the questionnaire, the students liked this approach. There were two interesting things about the experiment. One is that whoever scored the most points in the second test, only selected moderate difficulty levels in the second stage of the experiment. The second interesting thing is that those who chose the difficult level were almost the worst in the first test.

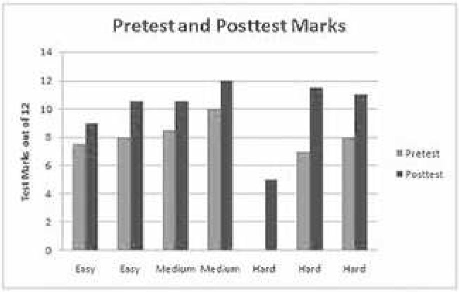


Figure 1. Pre-test and post-test marks for all students [7]

In this experiment, the number of participants was limited, which means that the result was statistically not significant. These students are not representative of high school students in general because they've been selected exclusively among those who were interested in the subject. They rather showed a possible upper limit on the performance.

In summary, we can say that the experiment was successful, and during the programming activities the students understood better the concept of error detection. The questionnaire revealed that students liked this approach, enjoyed programming, and would like to do other similar activities too.

CS Unplugged in Higher Education

I see more opportunities for the CS Unplugged in higher education. It is worth separating some options because there are different reasons, goals, and depths of teaching. These options are as follows:

- In teacher training for IT faculty's students
- In teacher training (grade school- and Kindergarten-teacher Training Faculty)
- In teacher training for non IT faculty's students
- Computer Science students (abbreviated: PTI students)
- Any other higher education training.

In teacher training for IT faculty's students

It is important for IT teachers to be most preoccupied with CS Unplugged teaching, as they will be the ones who can best teach this in public education and those who can promote this topic among their teachers. They would deal with this topic as part of a methodology course.

In grade school- and Kindergarten-teacher training

CS Unplugged teaching is also important in school- and kindergarten- teacher training, as these non-computer activities can be taught in kindergartens or taught for lower-level children. This age is about playing, gaining experience and we can build on later knowledge with unplugged activities playfully. At this time, it is suitable for children as well, it is ideal for day-to-day games in nurseries, and for kindergartens it is suitable for early nursery classes or as part of the afternoon play. The difficulty is that there are relatively few teachers and kindergarten teachers among people who have this kind of knowledge or who have the feeling of trying new things. That is why it is very important that CS Unplugged is used either in their training or in advanced study, primarily by highlighting playful elements.

In teacher training for non IT faculty's students

According to the Digital Education Strategy [12], daily use of digital devices, the new type of computer thinking and approach has to be introduced into classrooms within a few years. It aims to enable all people to acquire basic digital literacy in free training. Vocational training involves the up-to-date digital knowledge of the profession. CS Unplugged helps you learn digital literacy, students learn more about Unplugged activities and deepen their difficult concepts and knowledge. Unfortunately, the IT classes in public education are not present at every grade, and there is also a small number of teaching hours, so it is not sure that students are able to get used to computers in the IT class as much as they can use it in their lifetime. The IT has connections with almost every subject, so I find it very important that non-IT teachers meet at least tangentially with CS Unplugged activities that can be used in their own profession.

PTI students

DES [12] recommends online learning spaces in higher education, so I suggest this type of presentation. At www.csunplugged.org, there are many tasks that discuss university-level problems (e.g. minimal spanning trees, graph coloring tasks). Due to the large amount of curriculum and the requirement of independent learning at the university level, these auxiliary activities cannot be parts of the lesson, but can be utilized as auxiliary material online.

The above-mentioned experiment in my article also demonstrates that CS Unplugged activities and programming together help to understand the concepts and processes even more deeply. That is why it can take part in a curriculum for a programming language or methodology course.

In order to stop the high dropout mentioned in the introduction, it would be a possible idea in the first year to create a special study group that would help students who are far behind or have serious shortcomings to catch up. An integral part of this course can be CS Unplugged, supplemented with programming.

Any other higher education training

In the non-specialized higher education courses, CS Unplugged activities should be dealt with tangentially in the part of computer thinking course.

For the time being, there is no unified view of whether all students in higher education need to learn computer thinking. In the non-specialized higher education courses, the subject of IT lessons is very different, even abroad. Students from some of the disciplines begin to write code immediately and elsewhere they completely exclude programming.

By my opinion, it is very important for everyone to learn computer thinking, because he will need it later in his own vocation.

Application of CS Unplugged in certain topics of IT

Binary numbers

Guide of Canterbury University

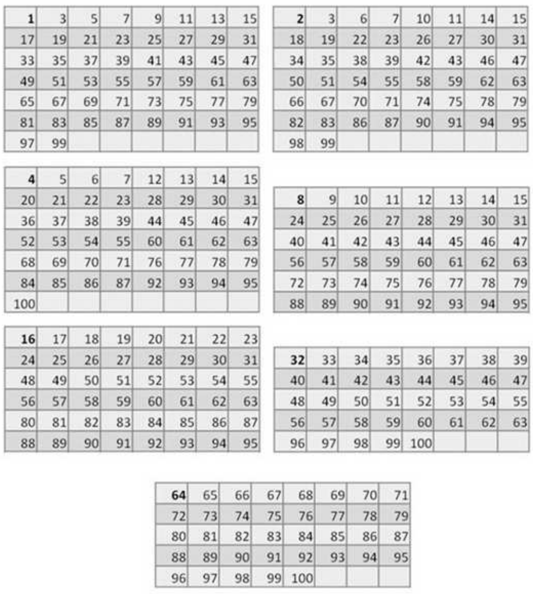
From the activities developed by Canterbury University [17], the first one deals with binary numbers.

- For the first task, the author offers 5 cards on which there is 1, 2, 4, 8, or 16 points. These cards must be used to extract the numbers.

- The next task is to send a secret message. The 26 letters are encoded as five-digit binary numbers and can be sent by any two states device (e.g. Christmas tree lighting).
- The author continues with perhaps the best-known task, which is the counting on the fingers. How long can you count on one hand: until 10 or 31? Children with lower grades are also happy to count on their hands until 31. An interesting question is how much we can count on both hands and feet (up to 1023 or 1048575)?
- Finally, the author explains the purpose of all this.

Guessing game

I would complement this occupation with a number guessing game and its implementation. We need the following seven cards for the game:



1	3	5	7	9	11	13	15
17	19	21	23	25	27	29	31
33	35	37	39	41	43	45	47
49	51	53	55	57	59	61	63
65	67	69	71	73	75	77	79
81	83	85	87	89	91	93	95
97	99						

2	3	6	7	10	11	14	15
18	19	22	23	26	27	30	31
34	35	38	39	42	43	46	47
50	51	54	55	58	59	62	63
66	67	70	71	74	75	78	79
82	83	86	87	90	91	94	95
98	99						

4	5	6	7	12	13	14	15
20	21	22	23	28	29	30	31
36	37	38	39	44	45	46	47
52	53	54	55	60	61	62	63
68	69	70	71	76	77	78	79
84	85	86	87	92	93	94	95
100							

8	9	10	11	12	13	14	15
24	25	26	27	28	29	30	31
40	41	42	43	44	45	46	47
56	57	58	59	60	61	62	63
72	73	74	75	76	77	78	79
88	89	90	91	92	93	94	95

16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95

32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
96	97	98	99	100			

64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100			

Figure 2. Number cards (own editing)

Think of a number between 1 and 100, tell me which cards it is on, and I will tell you which number you thought of. The guesser has an easy job, because he has to add only the numbers in the top left corner of the cards.

It is worthwhile to deal with this game with math teachers and school-teachers, but also in the training of non-IT and non-teacher students.

Another interesting task might be to create these cards with a program. Of course, then the guessing game does not go to 100, but to n. To create the program, it is worth noticing a pattern from the numbers on the cards. However, I only suggest the processing of the game in this way for IT and IT teacher students.

Educational experience

Based on my own educational experience, primary school students enjoyed the game and were curious about the solution. The cards were then copied and they showed them to others during the break or at home.

The IT teacher's students liked the game, with them we already created the cards that made them challenging.

Nim game

A recent option for this topic is the well-known Nim game. The essence of the game is that there are some lines, and in each line there are some matches. (In Figure 3, 4 rows and 1, 3, 5, 7 match in rows.) The next player can take from one line as many matches as they like. The winner is the one who gets the last match.

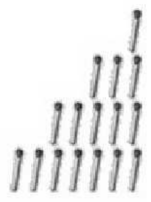


Figure 3. Nin game [12]

The winning strategy lies in transforming the rows into binary numbers and in the application of the XOR operation.

The presentation of the game, the definition of winning strategies for certain states, could be part of the curriculum of non-IT teachers, school-teachers and non-IT students. The generalization and perhaps the writing of a program could be part of the training of IT teachers and program design IT professionals.

Information Theory

Guess who

We introduce information theory in any training with a classical game called "Guess who".

The game can be played in several ways. If you play in a group, choose a topic, then send a member of the class. The others agree on who / what they will personally imagine. When the student comes in, they will talk to them as if they were the fictitional person/device and they should recognise who/what they are imperonating. The person whose question made them recognise what/who they are will have to go out to be the next riddle.

In another game you play in pairs, each person in the pair has a person / device on the head that is impersonated and needs to be discovered. So they talk to each other in the way mentioned above. The winner is who can find out who/what they are impersonating.

Twenty guesses

On the aforementioned page [17], activity 5 deals with information theory, this presents a method for measuring information content. The presented method is an adaptation of the game of 20 questions. Children should ask questions to a partner who can answer yes or no until they come up with the answer. During the game you will discuss how much information you have in a message or present a decision tree. Finally, an explanation is given for the purpose of the game.

The game shown is already playable at a very small age, but it is more interesting to write a program that can "play well", meaning that it does not really think of a number, but always follows the avoiding strategy, always thinking of the bigger interval between the intervals split by the number which was asked.

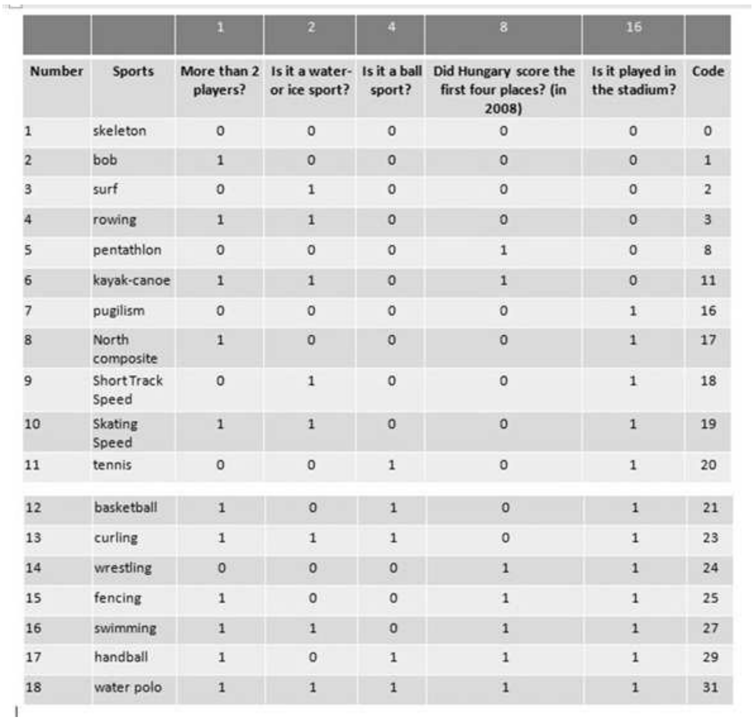
While I can imagine the presentation of the game primarily as a teacher training, I suggest to add the programming to the training of IT teachers and PTI specialists.

Development of Twenty guesses

For this work I developed the twenty questions game as follows. We should choose a theme (e.g. sport). One student thinks of a special sport – from the sports chosen by the teacher – then the teacher asks simultaneously five yes or no questions, based on which they will guess the answer.

When processing this task, you can present the decision tree of the game. It is even more interesting if we are trying to create a new data table with a student in a topic they choose, which includes the events, the five questions, and the code of the answers to the questions. Such a data table can be seen in Figure 4.

The game easily connects to any non-informatics college or science field, so it is equally suitable for IT and non-IT teachers as well as a general higher education training course in Computer Thought.



		1	2	4	8	16	
Number	Sports	More than 2 players?	Is it a water- or ice sport?	Is it a ball sport?	Did Hungary score the first four places? (in 2008)	Is it played in the stadium?	Code
1	skeleton	0	0	0	0	0	0
2	bob	1	0	0	0	0	1
3	surf	0	1	0	0	0	2
4	rowing	1	1	0	0	0	3
5	pentathlon	0	0	0	1	0	8
6	kayak-canoe	1	1	0	1	0	11
7	pugilism	0	0	0	0	1	16
8	North composite	1	0	0	0	1	17
9	Short Track Speed	0	1	0	0	1	18
10	Skating Speed	1	1	0	0	1	19
11	tennis	0	0	1	0	1	20
12	basketball	1	0	1	0	1	21
13	curling	1	1	1	0	1	23
14	wrestling	0	0	0	1	1	24
15	fencing	1	0	0	1	1	25
16	swimming	1	1	0	1	1	27
17	handball	1	0	1	1	1	29
18	water polo	1	1	1	1	1	31

Figure 4. Data table which includes the five questions, and the code of the answers to the questions (own editing)

Balance game

Next I propose the "balance game" of information theory. One of the 12 coins in the game is fake. False coin weights are different from others, but we do not know whether they are heavier or lighter. The task is then, that without measured weights, with a two-armed balance, what is the number of fewest measurements needed to determine which the counterfeit coin is and whether it is lighter or heavier than the others.

When playing this game, the students must reach the winning strategy by generalizing the actual numbers. IT students are expected to write a program that plays ”well”.

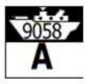
Searching Algorithm

Guide of Canterbury University

Six of the activities developed by Canterbury University [17] are concerned with searches. Perhaps this is the topic that most age groups can deal with.

The university's tutor introduces three search methods, linear search, binary search, and hashing strategy. Recommended for 9 years of age or older.

- In the introductory games, select about 15 kids and set them up against the class. Each child gets a card with a number (in random order). Before the remaining class, these numbers remain hidden. Then give some other kids a container with four or five candy. Their job is to find a particular number. They need ”to pay” to see a particular card. If they find the right number before using all their sugar, they can keep the residue.
- In the next introductory game, the only difference is that the children are sorted in ascending order.
- In the next session, students are playing in pairs. One member of the pair gets the 1A card and the other one 1B. There are a number of battleships (Figure 5) in the upper part of the cards, with a letter below it. In the lower part there are the same ships, but the number is missing (Figure 6). Each student should circle a battleship at the top of the game board and tell the number to their partner. They then alternately try to figure out where the partner's ship is. (The letter of the ship is to be said and your partner tells the number of the boat that the letter indicated.) How many shots did you need to find your partner's boat?

 *Figure 5.* Ship with number [17]	*Figure 6.* Ship without number [17]

- The game is followed by a discussion where, among other things, you discuss the lower and upper limits of the possible scores.
- They will play the same game as the previous one, but the numbers on the boats here are in ascending order.
- This game is also followed by a discussion where the definition of a good strategy and the term "binary search" are included.
- The third game presents the hashing strategy. In this game, the ships are stacked in columns and you can calculate which pillar the boat is in (0 to 9). You only need to add up the vessel's digits. The last digit of the sum is the number of the column in which the vessel is located. (For example, for ship 2345 2 + 3 + 4 + 5 = 14, so it is in column 4.)(It can be seen in Figure 7.)

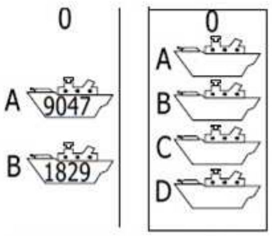


Figure 7. Battleships in columns [17]

- Discuss which ships are easier and harder to find.
- Finally, it is very important to compare the three search methods. Which is faster, which is the worst case and what is its computational cost?

  Educational experience

In a research [15] this guide was used to develop algorithmic thinking for secondary school students. Based on the results of the research, the students liked the games, and in their opinion they played a decisive role in understanding the algorithms.

Based on this, it can be said that the educational guide is very useful primarily in public education, but I'm sure students of higher education will be happy to play with it.

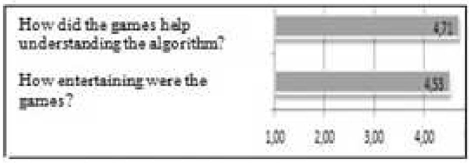
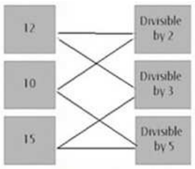


Figure 8. Questionnaire (ownd editing based on [15])

Backtrack

I would add to this activity a game that - albeit not so much - uses backtracking. In fact, I do not recommend a specific game, but just an opportunity to develop any known pair play games (e.g. memory, dominoes) to get to know this technique. The essence of this method is to not play with unequivocal pairs, with all the data having several possible pairs. In this case, there may be two pieces of data left in the end that are not pairs of each other. You then have to put back a pair that can have as pairs elements of the remaining data. It may not be enough to put just one pair back.

For the sake of easier understanding, I will illustrate with two examples. The first was taken from the topic of mathematics. An element of pairs is a number (12, 10, 16), the other is a property that tells you how to divide (by 2, 3, and 5). In Figure 9 we can see that each number has two (or more than two) pairs. We play a memory card with a few cards of this type (more than six). The last two remaining cards (Figure 10) may then be "10" and "divisible by 3". These are not pairs of one another, so you have to put back a pair that can be divided by three, and its attribute is that it is a divisor of 10. For example, "15" and "divisible by 5", that can be seen is Figure 11. This way we can finish the game.

 *Figure 9.* Initial state (own editing)	10 Divisible by 3 *Figure 10.* At the end of the game, we may not end up with a pair (own editing)

This game can be used on any teaching lesson and on any university training, enhancing your knowledge. I would most strongly emphasize it in the teacher trainings, but it can be tangentially covered in any forms of higher education.

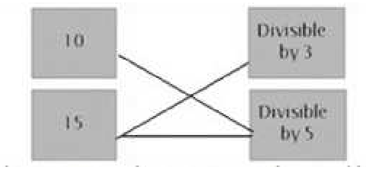
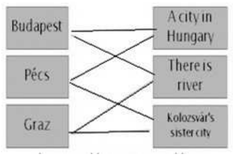


Figure 11. After putting back a proper pair (own editing)

The next example in Figure 12-14 is from geography and couples are cities and characteristics of cities.

 Budapest Pécs Graz A city in Hungary There is river Kolozsvár's sister city *Figure 12.* Initial state (own editing)	Pécs There is river *Figure 13.* At the end of the game, we may not end up with a pair (own editing)

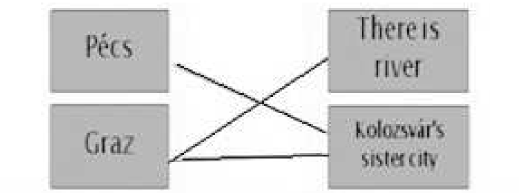


Figure 14. After putting back a proper pair (own editing)

Educational experience

I implemented the above-mentioned idea in a program called Memino. The program included a number of spreadsheets on several topics, but it would also be very easy to create new datasheets. At a methodological conference, I impressed the teachers and I received a request for the program and the idea, and a public education apprenticeship company bought the program for distribution. It can be said, though, that although I have no concrete teaching experience, my teacher colleagues have found the program useful and applied it during their training.

Recursion

Recursion is a very complicated method at first hearing, but whoever understands its essence and gets acquainted with it can easily apply it. During my primary school and my university teaching experience I experienced that a senior high school student can more easily write a recursive procedure in a logo programming language than an IT university student at an algorithms course.

For this topic, Canterbury University has not yet issued any guide, but there are a number of well-known recursive tasks that can be used to understand recursion, maybe not as usual.

First, let's look at a definition [1]: "Recursion is a technique in which a concept is defined, directly or indirectly, in terms of itself."

To learn the concept I suggest the following ideas:

Recursion in the world

We also encounter self-similar motives and fractals in nature. (It can be seen in Figure 15.) This may be the formation of clouds, crystal growth, but there are several examples of animals or plants in the world (e.g. cauliflower).

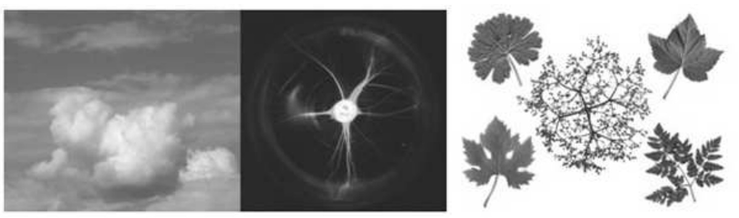


Figure 15. Recursion in the nature [15]

Recursion is also present in human works, and even with two mirrors (or mirror and phone), we can easily illustrate it by contrasting the two mirrors simultaneously.

These examples are worth presenting in all the training courses where the recursion concept emerges. (Figure 16.)

Hanoi towers

The task - that the Figure 17 shows - may be familiar to everyone The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

Figure 16. Recursion in human works [15]

(1) Only one disk can be moved at a time.

(2) Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack.

(3) No disk may be placed on top of a smaller disk.

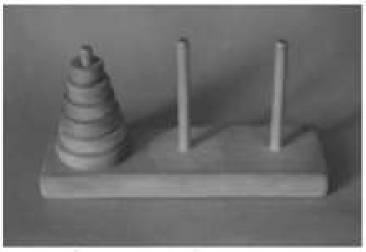


Figure 17. Hanoi towers [14]

The game was invented by French mathematician Édouard Lucas in 1883. The idea came from a legend that, when the world was created, a 64-wheeled Hanoi tower began to "play" Brahma's monks. The rules were the same as those of the Hanoi Tower today. According to the legend, when the monks are going to make the disks over to the third bar, the monastery crashes and our world ceases to exist. [21]

Solving the problem (*Hanoi (n, i, j)*, where n: number of disks, i: from where, j: where) consists of the following steps:

- If there is one disc, move the disc from the$i$-th rod to the j-th rod.
- Otherwise:

- Pass the tower of top $n-1$ disks from the first rod to the third rod (Hanoi $(n-1, i, 6-i-j)$
- Place the largest disc from the first column to the second. (Transfer (i, j))
- Pass the tower of the top $n-1$ disks from the third rod to the second rod. (Hanoi $(n-1, 6-i-j, j)$

It is worth playing this game on paper, but only with IT teacher training and PTI training students. It is worthwhile to play with high school children with real rods on real discs. (In case of lack of equipment, you can use a table instead of the rod, and books instead of the disks.)

A non-recursive game that nevertheless helps to understand the essence of recursion

The very essence of many simple known logic games is that a state of the game can be traced back to a simpler position, that is to say, for determining the winning strategy it is enough to determine the last position and a rule that can be traced back. Such games are for example the number ladder or the above-mentioned NIM game.

The number ladder rules: On an 11 degree ladder you can take 1 or 2 steps at a time. Two opponents take turns and the winner is the one who takes the last step. (Variations: instead of a ladder, you can face up to one another on a number line, or play with matches.)

To determine the winning strategy, a backward stepping vision is required. The starting player can win the game by first taking a 2 step and then always taking a number of steps which rounds his companion's steps to 3.

It is easy to write a program for this game, and it is best to use general N, M parameters instead of 11 and 2.

With such simple games, it is worth focusing on teacher training.

Comparison of recursive and iterative algorithms

Perhaps one of the most prominent examples of comparing the recursive and the iterative solution is the merge sort, which is well known in its iterative and recursive versions. To play algorithms, you will need some glasses to fill with different difficulty but with the same volume of materials. (For example, you can use soil, sand, rice, sugar, flour ...) The task is to set the glasses in ascending order by weight.

For an Iterative variant, we first combine the one-length sequences with pairs, then combine the ordered two, four, ..., 2^k-series sequences. The last series is usually incomplete.

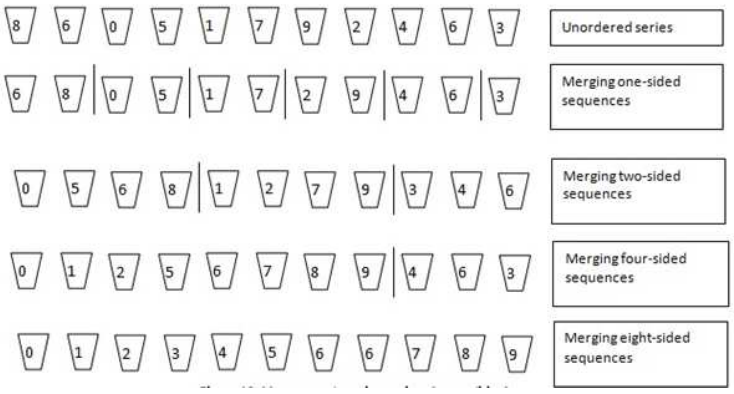


Figure 18. Merge sort, iterative variant (own editing)

The recursive variant is János Neumann's 1945 result, which is based on the principle of "Divide at Impera" rule. This is an algorithmic design strategy that focuses on subtracting the problem into smaller parts that can be recursively solved and then merged with the solutions.

Based on this, (Figure 19) we first halve (in case of an odd series, almost halve) the sequence, call the merge sort for both sequences, then combine the ordered sequences. At the demonstration, I don't use glasses to illustrate but I recommend it to the students.

I suggest playing algorithms at the training of IT teachers, and I consider it sufficient for students of PTI training as auxiliary material in online student space.

Pointer

In The algorithms and their applications course one of the most important, and one of the most difficult concepts to understand is the pointer, which, like recursion, hasn?t benefited from auxiliary material so far. That is why I think it is important to figure out a CS Unplugged activity for this topic as well. I recommend the following game for high school students (i.e. in IT teacher training) and PTI specialists as online guide.

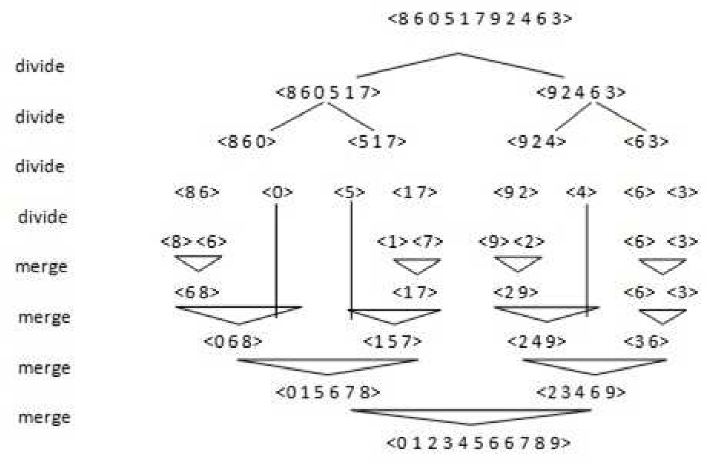


Figure 19. Merge sort, recursive variant

From the classroom students, select some who will be members of a list. Have a number on their neck (or shirt), which will be the key to the pointer. Have the children with their right hand on the next element, that is, the student who follows them in the list. If you want to show a bidirectional list, direct point them to the previous item with their left hand. If the pointer is NULL, do not show them with any hand. For a head element list, the teacher should be the head, who is different from the pupil by not having a key (that is, not having a number on their neck or T-shirt). Creating a new pointer: making a student stands up, and deleting a pointer should be illustrated by making the student sit down. That way we can illustrate the three basic operations: search, insert, and delete.

Finally, it is important to link pointers and recursion, which is best achieved by presenting the binary tree data structure.

I used the above-mentioned idea in the algorithms and data structures course, not as part of the lesson, but as auxiliary material. I asked students to fill out a questionnaire. I only got ten answers so my survey cannot be considered representative but it still shows a picture of the usability of the idea. In the questionnaire I asked 4 questions and gave students the opportunity to write a personal opinion. The result is shown in Figure 20. Based on the evaluation of the students, it can be said that the used help made it easier to understand the pointer.

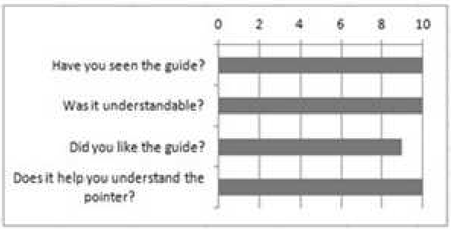


Figure 20. The results of the questionnaire (own editing)

Summary

While in the IT industry, the market is huge and profitable; in a number of countries the recruitment limit is reduced. The disappearance of women from the market is further noticeable, and the high dropout rate from the university is typical. One of the reasons for these problems may be the lack of understanding of IT skills.

In my article, I was wondering how to supplement the curriculum with CS Unplugged activities to help address problems.

In my article, after the general characterization, I presented a foreign study that concluded that CS Unplugged helped to deepen programming knowledge.

After that, based on Canterbury University's research, I supplemented free online download sessions with my own assignments and educational experiences.

Since not all topics have been elaborated, I have also been looking for exercises for topics that are part of special training courses.

Based on the surveys in my article, it can be said that adding CS Unplugged to courses helps students to understand and deepen the curriculum and to sharpen the lesson. The students clearly expressed their interest in each survey, and became more interested and more motivated.

Based on the above, CS Unplugged is primarily used in higher education, which will hopefully have an impact on public education as well.

References

[1] Al Aho & Jeff Ullman, *Foundations of Computer Science*, 1992 `http://infolab.stanford.edu/~ullman/focs/ch02.pdf`. Last download: 05.09.2017.

[2] Az Eötvös Loránd Tudományegyetem Intézményfejlesztési terve Helyzetelemzés, 2012.
https://www.elte.hu/file/ELTE_IFT_helyzet.pdf. Last download: 05.09.2017.

[3] B. Bakó, Miért lett óriási az informatikushiány?, in: *Mandiner, 2016.10.26*, http://mandiner.hu/cikk/20161026_miert_
lett_oriasi_az_informatikus_hiany. Last download: 05.09.2017.

[4] Bellresearch, A hazai informatikus- és IT-mérnökképzés helyzetének, problémáinak, gátló tényezőinek vizsgálata, in: *Education, 2016. 03. 10.*, Hungary.

[5] http://24.hu/belfold/2016/09/05/oriasi
-informatikushiany-van-magyarorszagon/ Last download: 05.09.2017.

[6] http://info.berzsenyi.hu/logo/elmelet/rekurziv-eljarasok Last download: 05.09.2017.

[7] Informatikust mindenhová
https://www.it-services.hu/hirek/informatikust-mindenhova, Last download: 05.09.2017.

[8] J. Voigt, T. Bell, Copetition-style programming problems for Computer Science Unplugged activities, in: *A new learning paradigm: competition supported by technology*, (E. Verdu, R. Lorenzo, M. Revilla & L. Regueras,, eds.), CEDETEL, 47151, Boecillo, Spain, 2009, 207–234.

[9] Kijöttek a PISA-eredmények: rosszabb, mint valaha
http://index.hu/tudomany/2016/12/06/pisa_felmeres_eredmenyek/
Last download: 05.09.2017.

[10] K. Kovácsné Pusztai & T. Török, Keretprogramok az oktatásban, in: *INFO ÉRA*, Füzesgyarmat, Hungary, 2007.

[11] K. Kovácsné Pusztai & T. Török, Számítógépes oktatóprogramok használata bármely tanítási órán, in: *Info Savaria*, Szombathely, Hungary, 2002.

[12] K. Kovácsné Pusztai, Meminó, in: *Suli-Soft*, Budapest, Hungary, 2003.

[13] K. Kovácsné Pusztai, Számítógépes gondolkodás a felsőoktatásban, in: *INFO DIDACT*, (ISBN: 978-615-80608-0-6) Zamárdi, Hungary, 2016.

[14] Magyarország Digitális Oktatási Stratégiája, Bp., 2016. 06. 30. availablefromhttp://ivsz.hu/wp-content/uploads/2016/10/magyarorszag
-digitalis-oktatasi-strategiaja.pdf. Last download: 30.10.2016.

[15] V. Mahler-Lakó, A. Mahler, Algoritmikus gondolkodás fejlesztése keresési és rendezési algoritmusokon keresztül, in: *INFO DIDACT*, (ISBN: 978-615-80608-0-6) Zamárdi, Hungary, 2016.

[16] http://ordoglakat.blog.hu/2011/03/20/hanoi_tornyai. Last download: 05.09.2017.

[17] Sulinet: Informatika 9.-12. évfolyam
http://tudasbazis.sulinet.hu/hu/informatika/informatika/informatika-
9-12-evfolyam Last download: 28.04.2017.

[18] T. Bell, J. Alexander, I. Freeman & M. Grimley, Computer Science Unplugged: school students doing real computing without computers, *New Zealand Journal of Applied Computing and Information Technology* **13**, no. 1 (2009), 20–29.

[19] T. Bell, Ian H. Witten & M. Fellows, Computer Science Unplugged, 2015 `http://csunplugged.org/` Last download: 05.09.2017.

[20] V. Vígh, Rekurzív logikai játékok, SZTE Bolyai institut Szeged, Hungary, 2014.

[21] Yvon Feaster, Luke segars, Sally K. Wahba, Jason O. Hallstrom, Teaching CS Unplugged in the High School (with Limited Success), in: *Proceeding of the 16th annual joint conference on Innovation and technology in computer science education*, Darmstadt, Germany, 2011, 248–252.

[22] Wikipedia `https://hu.wikipedia.org/wiki/Hanoi_tornyai` Last download: 15.01.2018.

KINGA KOVÁCSNÉ PUSZTAI
ELTE FACULTY OF INFORMATICS
DEPARTMENT OF ALGORITHMS AND APPLICATIONS
H-1117 BUDAPEST, PÁZMÁNY PÉTER SÉTÁNY 1/C

E-mail: `kinga@inf.elte.hu`

(Received September, 2017)