18/4 (2020), 295-305
DOI: 10.5485/TMCS.2020.0486
tmcs@science.unideb.hu
http://tmcs.math.unideb.hu

Teaching
Mathematics and
Computer Science

Better understanding mathematics by algorithmic thinking and computer programming

KATALIN FRIED, ISTVÁN FEKETE AND PÉTER PRINCZ

Abstract. Tamás Varga's mathematics education experiment covered not just mathematics, but also other related topics. In many of his works he clearly stated that computer science can support the understanding of mathematics as much as mathematics supports informatics. On the other hand, not much later than the introduction of the new curriculum in 1978, personal computers started to spread, making it possible to teach informatics in classes and in extracurricular activities. Varga's guided discovery approach has a didactic value for other age groups as well, not only in primary school. Its long-lasting effect can be observed even in present times. Having reviewed several educational results in the spirit of Tamás Varga, we have decided to present an extracurricular course. It is an open study group for age 12-18. Students solve problems by developing Python programs and, according to our experiences, this results in a deeper understanding of mathematical concepts.

Key words and phrases: mathematics education, informatics education, computer programming, Python language, experimental problem-solving, algorithms and data structures.

MSC Subject Classification: 97B10, 97B20, 97D50, 97N80, 97P20, 97P30, 97P40, 97P50, 97U70.

Tamás Varga's reform movement and its legacy

Tamás Varga (1919-1987) was an outstanding figure in Hungarian mathematics education, one of the leaders of the reform movement. A comprehensive review of Varga's role in renewing mathematics education in Hungary can be read in (Pálfalvi, 2019) and (Gosztonyi, 2020).

During the complex mathematics teaching reform experiment he supervised between 1963 and 1978, the curriculum was expanded with new topics – such as sets, logics (see in (Varga T., 1969)), series, functions, combinatorics, probability and thinking methods –, which were intertwined and were built on one another, also through exercises solved in class. Teachers, following the principle of guided discovery, improved students' motivation, allowed them to work and play with tools, and gave them the freedom to make mistakes.

Figure 1. Tamás Varga (right) having a discussion with mathematicians János Surányi (left) and Ervin Fried (middle, father of the first author) in front of the Hungarian Academy of Sciences building. (Courtesy of Mária Halmos and Katalin Ács)

In this article, we discuss two characteristic effects of Varga's work in more detail. One of them is the appearance of the elements of computing in the curriculum, in the frame of the early Hungarian era of computers. The other is the validity of his approach to different age groups of students. Both effects can be seen throughout time, from the beginning to the present day.

The elements of algorithms and computing also appeared in the two-volume book on new mathematical topics (Varga T., 1972-1973). The use of calculators – under controlled conditions – was supported in the curriculum; see in (Varga T., 1980). When

quoting details of the 1978 curriculum, Pálfalvi mentions: “The other aim of using calculators in schools is developing algorithmic thinking” (Pálfalvi, 2019).

There were several events that influenced Tamás Varga in suggesting the importance of teaching the subject Informatics, which covers basic algorithms, some computational techniques, usually the use of a ‘simple’ programming language, some elementary computer programming.

- In Hungary, there were experiments in robotics already in 1955. Dániel Muszka built the Ladybird robot at the Szeged University (Wikipedia, Ladybird of Szeged, n.d.).
- Varga himself used a programmable calculator and learnt how to solve elementary algorithmic mathematical exercises with it. Later, he bought an early PC (1980).
- Universities launched ‘Computer programming’ lines (Szeged, Debrecen, Budapest).
- A ‘School computer’ program was launched in 1981, when all schools were given PC’s; enough to teach informatics in at least one class at the time. Informatics became a compulsory subject in secondary schools.
- The appearance of personal computers was accompanied by the publishing of a gap-filling book by (Fried, Kepes, Sztrókay, & Török, 1984).
- Informatics was introduced as obligatory subject for prospective teachers at middle school level (1983).
- A textbook for the subject of Informatics for secondary schools was published (Simonovits, 1985).
- LOGO programming language was introduced in some kindergartens.

The emphasis in informatics education has changed over time. Programming skills were replaced by ‘application abilities’, programming by learning the use of already existing computer programs. The drawbacks of this paradigm are evident.

“From kindergarten to university, Tamás Varga transformed the Hungarian culture of mathematics learning, mathematics education and pedagogical work at all levels,” – as science historian István Gazda said (Gazda, 2015) at the ceremony in 2015 where the Hungarian Heritage Award was posthumously presented to Tamás Varga (together with his brothers, Balázs Vargha, a literary historian, and writer Domokos Varga).

Before writing this article, the authors inquired around in their environment and found the following cases of teaching informatics, which cover the all age spectrums:

(1) The playful introduction of LOGO in kindergarten of ELTE. Discussion with the then director of the program.
(2) Discussion about present-day informatics education with a mathematics teacher who, as well as her school, participated in the experiment of Varga.
(3) Discussion with a secondary school teacher who, already in the years of the experiment held extracurricular classes in informatics and was in good terms with Varga.
(4) The academic results of an outstanding secondary school and extracurricular class in informatics, whose teacher also does scientific research on the didactic issues of informatics education.
(5) Discussion with the leader of the playhouse "Kuckó" about the activities, where interested children from age 12 to 18 are welcome.
(6) An open extracurricular class for 12-18-year-olds led by P. Princz, one of the authors of this article.
(7) The class material of a former teacher training college covering algorithms from several topics of mathematics (Fried & Simonovits, 2004).
(8) A university course "Algorithms in Python", with the goal of programming with a mathematical and algorithmic background. (The lecture given by I. Fekete, one of the seminars held by P. Princz; both authors of this article.)

The materials of the conversations are still being processed. Instead of reviewing the full spectrum, the authors finally decided to present the material of one of them, the one in point (6), in detail.

A computer study group for age 12-18

In the next sections we provide an overview of a computer study group for 12-18-year-old students announced by one of the authors. The course is held near Budapest, open and free for everybody. It takes 3 hours one day at the weekend for 12 weeks (in municipal organization). Its goal is to offer creative usage of computer science and to convey practical knowledge pointing towards problem solving and programming. A

further goal for higher-grade students is to be able to graduate from high school in the subject of Informatics on advanced level.

Experience has shown that students are strong in media-centric Internet use and word processing applications. The course would like to supplement this knowledge with analytical thinking and program development, which in this case connects computer science with mathematics and other subjects.

Today, Python is considered one of the most suitable languages for this purpose worldwide. It is a characteristic approach of the course to start using Python almost intuitively when solving problems. A more comprehensive language review is always done after the applications. Many exercises are solved in cooperation, using group thinking, but mostly individual programming.

Some exercises solved in the study group

In this section we list some of the typical problems discussed, without being exhaustive. It is followed by the curriculum of the study group, supplemented by several methodological observations in the following chapter. Playing with each of the exercises presented below was a joyful and successful activity for the students.

Battleship game

This is a well-known game from our childhood and can be played on checkered paper. The two players try to sink the opponent's "battleship" in 5 × 5 squares with "torpedo shots". Each player has five shots. The winner is the one who first hits the other player's battleship. Here are some of the didactic values of this exercise, even it is a simple console game with ASCII art, without turtle graphics:

- Good representation of board games (chess, go) as well.
- The battlefield is a non-trivial data structure: a list of lists.
- The concepts of embedding and recursion appear.
- We also learnt typical game loops and game control structures.

Programming this game offers the opportunity to discuss Pythagorean theorem, coordinate geometry, and the concept of Euclidean distance, perpendicular, and projection, all in a playful manner.

```
. . . . .
. . . . .
. X . . .
. . . X .
. . . . .
You have 3 more shots
Where do you shoot? (row): 2
Where do you shoot? (column): 4
2 4
Almost, but the distance is:  1.4142135623730951
. . . . .
. . . X .
. X . . .
. . . X .
. . . . .
You have 2 more shots
Where do you shoot? (row): 1
Where do you shoot? (column): 3
1 3
Hit! My battleship is destroyed...
Game over!
```

Figure 2. Screenshot from a part of a battleship game

The print screen of the console input-output shows (Figure 2) the last two steps of a battleship game, so that the locations of the first two shots can also be seen.

Drawing polygons and other shapes

Students draw relatively simple geometric shapes, which however can be described using formulas. Some examples are regular polygons, circles, ellipses, and egg shapes. The turtle graphics associated with Seymour Papert has been successfully integrated into Python. In this way, LOGO can be used in a modern language.

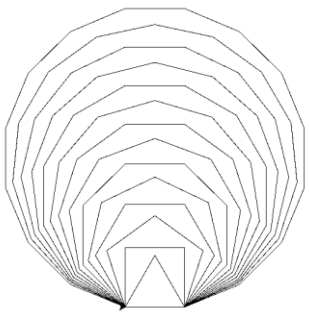

Figure 3. Sequence of regular polygons

Figure 3 illustrates a series of regular polygons of fixed side lengths. Apparently, the sequence of these polygons is converging to a circle, towards infinity.

The solution to drawing shapes accomplishes the following didactic values:

- Children enjoy watching the trajectory being plotted, the function stepping through the desired interval.
- Actually, the slower, the better!
- A few examples follow. Unfortunately, only the end result is presented as a screenshot.

Programming polygon plots like those shown above offers the opportunity to discuss the concepts of geometric convergence and a glimpse into the analysis.

A competition problem about cryptography

"KöMaL" (Mathematical and Physical Journal for Secondary Schools) regularly publishes problems in a score content in mathematics, physics, and informatics. The problem I. 421. was to create a program for encrypting – decrypting an arbitrary text with the help of the ADFGVX ciphering (based on a true story from World War I: invented by the Germans, cracked by the French).

The following can be learnt from this complex competition problem:

- It is not visual at all: no plotting, not even console printing, it is just reading the input and writing the output.
- Ciphering – deciphering (mapping and inverse mapping between alphabets) needs algorithmic thinking.
- Children need to understand both algorithms, cipher a short message, and decipher it to check they could recover the ciphered message.

Solving algorithmic problems like the example shown is an excellent opportunity for the teacher to touch the concepts of functions and inverse function, mapping between character sets.

Plotting the trajectory of projectile motion

According to the National Core Curriculum of Hungary (NAT 2012) projectile motion is introduced in the 9th Grade Physics. One of the aims of the course is to form a bridge not only to mathematics but also to other subjects, especially physics, from the

aspect of solving programming problems. This class of problems is the theoretical background for the popular Moon landing games (see Figure 4).

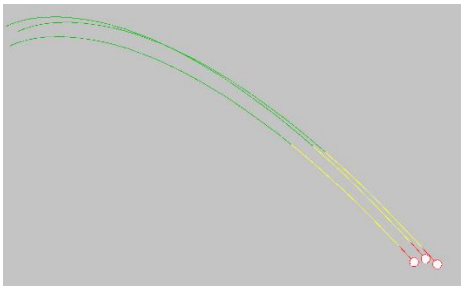

Figure 4. Trajectory plot of simulated lunar landings

This problem has been solved several times on Researchers' Night events organized by the Skool Foundation and Ericsson, as well as in the Programming for Girls courses (Princz, Inclined throw, 2018a).

The outcome of this thought-provoking problem can be summarized as follows:

- Teaching programming from the start for 15-18-year-old children.
- National Core Curriculum (NAT 2012), 9th Grade, Physics. Hence the target age group.
- A 4-hour (net, 4 × 60 minutes) workshop, best if booked for a whole day, from 9 am to 4 pm.
- There is a free access to it at a downloadable GitLab project: (Princz, Computer study group, 2018b).
- The sample code of the interim steps for every approx. 30-minute progress is also accessible on the GitLab link given above (see the code directory).

Programming function trajectories like the example shown above is an opportunity to discuss physics and mathematics in unity, to view physics as an exciting, real-world application of mathematics, like quadratic equation and parabola.

The curriculum of the programming study group

Independently from solving exercises, teaching programing has its own didactics, which can be characterized as follows:

- Tackle a relatively broad subset of the imperative programming.
- Best if it can be completed in 4-6 hours. (Fits into a one-day workshop.)
- Best if it is shorter than 70 lines of code, i.e. fits on one A4 page printed.
- Smaller, incremental steps: a few minutes' theory explanation, followed by writing 5-10 lines of code as practice, refined later.
- Graphical and slow execution for visual feedback. Progress of the algorithm, plotting the trajectory on the interval can be followed in real time by humans (as in Logo).

Python has many different directions and levels, so it is necessary to devise an educational concept for what and how to teach. The curriculum of the course, divided into 12 weeks, including 3 hours of work, together with the exercises, is as follows:

1 The Python language. Installation. Environment. The Hello, world! program. Print statement. Sequence of statements.

2 Numeric representations. Variables and types. Converting. Assignment. Operations. Python console. The input() function. Logo, turtle module.

3 Branching: if, if..else, the elif keyword. Logical expressions. Comparisons.

4 Looping: while and for keywords, break, continue and else, nested loops.

5 Defining, calling, and composing functions. Parameters. Lists in Python.

6 Visibility and lifetime of global and local variables. Code refactoring.

7 Introduction to data structures: string, list, tuple, dictionary, set.

8 Data structures continued in-depth: tuple, dictionary, set.

9 Exception handling, runtime errors. First draft of the battleship game.

10 Finishing the battleship game. File handling, I/O. Type conversion.

11 The standard library. Useful modules. The object-oriented paradigm.

12 Summary of the semester. Discussing a competition problem from KöMaL.

Books that teach a programming language typically set simple exercises for each language idiom, but these exercises rarely form a series of increasing difficulty and are not suitable for deepening algorithmic thinking. Mathematics textbooks, on the other

hand, naturally follow a path from easier to more difficult problems, although not with the intent to be programmed.

The novelty of the approach of this study group is that it is trying to bring together the best of both worlds: that is, to set exercises in increasing difficulty of the mathematical background, yet offer the possibility to view them as programming problems of similar difficulty. This can bring a sense of achievement for the students both in programming and mathematics. Overall, the authors think this approach is a step towards the mindset exhibited in (Fried & Simonovits, 2004).

Acknowledgement

The research has been supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.2-16-2017-00013, Thematic Fundamental Research Collaborations Grounding Innovation in Informatics and Infocommunications).

References

Fried, K., & Simonovits, M. (2004). *A problémamegoldás számítógépes iskolája (How to solve it on computers)*. Budapest: Typotex.

Fried, K., Kepes, J., Sztrókay, K., & Török, T. (1984). *Etűdök személyi számítógépekre (Etudes for personal computers).* Budapest: Gondolat.

Gazda, I. (2015). *Magyar Örökség Nagydíj (Hungarian Heritage Award)*. Retrieved February 5, 2020, from https://www.vtamk.hu/magyar-orokseg-dij/

Gosztonyi, K. (2020). Tamás Varga's reform movement and the Hungarian "Guided Discovery" approach. *Teaching Mathematics and Computer Science*, *18*(3), 11-28.

Pálfalvi, J. (2019). *Varga Tamás élete és a komplex kísérlet (The life of Tamás Varga and the Complex Mathematics Educational Experiment)*. Budapest: TypoTeX.

Princz, P. (2018a). *Inclined throw.* Retrieved February 5, 2020, from https://gitlab.com/princzp/4hr_python_workshop

Princz, P. (2018b). *Computer study group.* Retrieved February 5, 2020, from https://gitlab.com/princzp/prog_szakkor

Simonovits, M. (1985). *Számítástechnika tankönyv (Textbook on computer science).* Budapest: Tankönyvkiadó.

Varga, T. (1969). *Matematikai logika kezdőknek I-II. (Mathematical logic for beginners).* Budapest: Tankönyvkiadó.

Varga, T. (1972-1973). *Játsszunk matematikát (Let's play mathematics) 1-2*. Budapest: Móra Ferenc Ifjúsági Könyvkiadó.

Varga, T. (1980). Az új matek – számolás fejben, írásban, géppel (The new mathematics – mental, written, computational calculation.). *Élet és Tudomány*. https://adtplus.arcanum.hu/hu/view/EletEsTudomany_1980_1/?

Wikipedia, Ladybird of Szeged. (n.d.). Retrieved November 1, 2019, from https://en.wikipedia.org/wiki/Ladybird_of_Szeged

KATALIN FRIED
ELTE, FACULTY OF SCIENCE, INSTITUTE OF MATHEMATICS

E-mail: friedkati@caesar.elte.hu, kfried@cs.elte.hu

ISTVÁN FEKETE
ELTE, FACULTY OF INFORMATICS, 3IN RESEARCH GROUP, MARTONVÁSÁR

E-mail: fekete.istvan@inf.elte.hu

PÉTER PRINCZ
ELTE, FACULTY OF SCIENCE, PHD STUDENT
ERICSON HUNGARY R&D

E-mail: princzp@caesar.elte.hu, peter.princz@ericsson.com