Informatics in Education, 2024, Vol. 23, No. 4, 869–882

DOI: https://doi.org/10.15388/infedu.2024.31

Pascal as the Driving Force of Informatics Education in Schools in Poland

Maciej M. SYSŁO

Warsaw School of Computer Science, Warsaw, Poland
e-mail: syslo@ii.uni.wroc.pl

Dedicated to the memory
of Niklaus Wirth, whose Pascal
shaped informatics education in Poland
for over 20 years

Abstract. The first books in Polish about the Pascal programming language appeared in the late 1970s, and were soon followed by a Polish translation of Niklaus Wirth's book *Algorithms + Data Structures = Programs*. At that time, many efforts were made to prepare teachers to teach informatics in schools, and Pascal was one of the topics taught, often with no access to computers.

The situation improved significantly after the first informatics curriculum was approved for schools by the Ministry of National Education (MEN) in 1985, in response to the increasing number of microcomputers in schools and students' interest in programming. As a confirmation of the conclusions drawn from the discussion in the ABACUS magazine (1984) on "Which is the best programming language for a first course in computer science," Pascal very quickly became the basic programming language in Polish schools as a tool for practicing algorithmics. The first informatics textbook was published in 1988 and in it, after a short Logo course, Pascal took the main place as a language for developing solutions to algorithmic problems in the form of computer programs.

In the following years, many teaching aids were developed for programming education in schools, as well as other aids, e.g., an authoring system, programmed in Pascal. Pascal was the primary programming language in informatics education until the end of the first decade of the 21st century, subsequently replaced by Python.

It is interesting that Polish winners of national and international Olympiads in Informatics for many years used Pascal, demonstrating the enormous professional values of this language.

In this article we present in detail, among others things: the role and influence of Pascal on the development and level of informatics education in schools in Poland, as well as other scientific activities and initiatives supported by this language. Many students educated in this technical and mental environment found their place in professional development teams in the best computer and technology corporations, and evidently most of them joined the national working force.

Key words: Pascal, programming, informatics, education.

1. A Bit of History Before Pascal

In this section we briefly review activities of educational nature which preceded the decision described in Section 3, considered as the formal beginning of informatics education in Poland. All the projects and activities mentioned here have directly or indirectly influenced the role that Pascal played in informatics education in Poland until recently.

1.1. *First Informatics Classes*

Two initiatives related to teaching informatics[1] in schools, which appeared in 1960–1970 in two academic centers in Wrocław and Warsaw, are considered as the beginning of informatics education in Poland. They were different in nature, since they were born in different academic environments. At the University of Wrocław in 1962, the Numerical Methods Division (KMN) was established within the Mathematical Institute and the mainframe computer (Elliott 803, made in the UK) was installed as the first commercially made computer in universities in Poland. In 1964/1965, academic teachers from the KMN offered some high schools informatics classes with the possibility to write programs and run them on a real computer. On the other hand, in Warsaw the informatics lessons started in 1970 and were run in two university mathematics classes, and they were devoted mainly to some theoretical foundations of computations and informatics (Sysło, 2014).

The first informatics classes in Poland were established in two high schools (I and III) in Wrocław in the school year 1964/1965, when computers were still called "mathematical machines," and it was only at the end of the 1960s that the term "informatics" was introduced to describe computer science. The school subject was called *Programming and Using a Computer* and its syllabus was proposed by the KMN (Zuber, 1966). As in those days computers were mainly used for numerical calculations, students in these first informatics classes learned some basic numerical methods for solving mathematical[2] problems and use some programming languages for Elliott 803 (Assembler, Autocode Mark III, and Algol 60) to run their programs on a real computer.

Informatics was taught in those schools by academic teachers. For a given problem, students proceeded through all the stages of solving a problem with a computer, which today is called algorithmic thinking (or computational thinking): describe a problem, choose an algorithm and data structure, implement an algorithm as a computer program, run the program, debug and analyze outputs obtained and verify the correctness of your solution, and discuss possible applications of the solution. Sysło (2023) illustrates how close this approach was to today's ideas of computational thinking with appropriate mental tools; Sysło (2014) discusses the role of programming and computers in problem solving.

[1] In this paper, we use the term 'informatics' for computer science, which is a common practice in texts and discussions in Polish.

[2] For this purpose, the syllabus of mathematics was extended by adding some sections and topics related to numerical methods and computer calculations such as error analysis, mathematical tables and interpolation, polynomials, solving algebraic equations and systems of linear equations, and numerical integration.

1.2. *Algol 60 as a Predecessor of Pascal*

In the first years of informatics classes, there were no textbooks for school lessons, teachers used their own hand-outs. Then some texts were published on programming Elliott 803, Algol 60 (Paszkowski, 1965), and numerical methods and programming with programs in Algol 60 (Zuber, 1972).

Although Niklaus Wirth was not among the authors of Algol 60, his sympathy for this language led him to Algol W, and then to Pascal.

Academics at the KMN department of the University of Wrocław were staunch supporters of Algol 60 as not only a programming language, but primarily as a tool for designing algorithmic solutions to problems in various fields, and experimenting with them on a computer happily standing next to them. This is how the idea of a series of books devoted to various branches of informatics and mathematics was born in the KMN, of which Algol 60 programs were an integral part. One of these books (Kucharczyk and Sysło, 1975) was later expanded and the Algol 60 programs were replaced by programs in Pascal (Sysło *et al.*, 1983). Today you can find similar books on many different topics, supplemented with software in many different languages and environments (Sysło, 1997, 2016), often available online.

1.3. *Teacher's Preparation*

In the 1970s, a syllabus for informatics lessons in mathematics classes in high schools and an accompanying syllabus for in-service training of mathematics teachers to prepare them for teaching informatics classes were developed in the Institute of Teacher Education (IKN, Warsaw), which also organized in-service courses for mathematics teachers on teaching informatics in high schools. In contrast to the Wrocław syllabus for informatics, the IKN syllabus assumed that informatics may be taught with no access to computers. It was only advised to visit a computing center equipped with a computer to run programs. However, it is estimated that by 1980, more than 1 000 teachers attended in-service courses for informatics teachers, and informatics was taught in more than 1 000 high schools in Poland, although most of the classes had no access to any computer.

In the early 1980s, many universities and computing centers offered teachers in-service courses on various aspects of microcomputers and their use in schools for informatics classes and on how to support learning and teaching in other subjects. Learning to program in Pascal was one of the most popular courses offered both to teachers and to students in schools and as part of extracurricular activities. The leading lecturer of these courses was Jan Madey from the University of Warsaw; he also wrote many educational materials on programming and co-authored a textbook on Pascal (Iglewski *et al.*, 1978), which was the main resource on the language.

1.4. *Government Decisions and Actions*

In the 1970s and 1980s, the Polish government recognized computer technology as an important component of the national economy and made a number of decisions that were

intended to contribute to the development of informatics education on K-12 and university level. Many government programs and projects were established. The greatest importance was attached to (1) the education and development of teachers in the field of informatics and (2) the creation and development of educational software. In the latter area, the creation of packages, libraries, and systems in the Pascal language environment had gained the greatest attention and popularity.

In the mid-1980s, a special department was established in the Ministry of National Education (MEN) to manage and monitor activities ran by government agencies and public institutions with respect to the computerization of schools. One of the tasks of this department was to conduct projects related to supplying schools with computers and peripheral devices. A project was implemented to build a Polish microcomputer for schools, *Elwro Junior*, which inherited many features from its *ZX Spectrum* prototype, including the Pascal language compiler (see Section 4.1). The next initiatives through many years that followed concerned the purchase of IBM PC computers for schools, equipped with a licensed Turbo Pascal 7 environment.

In the second half of the 1980s, several governmental projects were established to accomplish the goals formulated in the state programs. Two such projects, RRI.16 and RRI.14, were the most important with regard to their achievements for general informatics education.

The goal of Project RRI.16: "Informatics For Schools – Development of Informatics Education in High School and Preparation Of Education Software and Its Implementation In Schools" was to (1) prepare educational materials for the subject *Elements of Informatics* (EI) in high schools (see Section 3); (2) develop methods for teaching and learning with computers; and (3) produce sample packages of educational software. The most important results obtained in this project were the first textbook for informatics in Poland (see Section 3) and several systems and packages of educational software for different school subjects (see Section 4). All these products were made available for students and for schools – they were either bought by the MEN and then sent to schools (software) or they were made available for purchase (textbooks).

The Project RRI.14: "Informatics And Computer Support of Education And Research in Higher Education Institutions" was addressed to tertiary education and its goal was to (1) equip higher education institutions with modern computers and informatics tools; (2) produce sample software packages for supporting educational processes; and (3) prepare and train academic and school teachers for incorporating modern computer technology into their teaching and research. Hundreds of software systems and packages were designed and produced by the participating schools and universities, and then made freely available to all other institutions participating in the project. The software was developed in Pascal and it was intended to support students and teachers in learning and using Pascal in solving problems from a variety of areas.

Today we should acknowledge that the two projects from the second half of the 1980s laid a solid foundation for future decisions and activities in the area of informatics education in all types of schools on all levels of education. Schools had gotten modern computer systems equipped with application and educational software to support education and

research (in universities). The software was documented in a uniform way in three volumes, and although the production and distribution of the project's results (books and software) were not the project's responsibilities, the participating and other educational institutions could freely copy and use the results for non-commercial use in education and research.

2. On Teaching and Learning Programming

In teaching programming in the first school classes in the mid-1960s, in preparing teachers in the 1970s to teach informatics, and later introducing Pascal into teaching and learning informatics, we followed the approach set forth by Niklaus Wirth in his early paper (Wirth, 1971a) which he then developed in his seminal work – the book *Algorithms + Data Structures = Programs* (Wirth, 1976, 1980).[3]

Wirth distinguished programming from coding, which is usually taught through examples demonstrating specific techniques. Unfortunately, nowadays we can see the growing popularity of coding, especially addressed to the youngest students and novices in programming. Teaching through examples has two main drawbacks: examples are quite often complete programs and they are chosen to show what computers can do. Active and creative programming should however consist of designing new programs, rather than deconstructing given programs. Quite often students get the impression that learning programming depends mainly on mastering a programming language.

The proper choice of examples – problems to be solved – is a crucial point in introducing and understanding programming techniques, which are gradually developed – according to Wirth, in a sequence of top-down refinement steps –, and which students then implement in the form of programs solving the problems considered.

In 1984, the ABACUS magazine vol. 1, no. 4 (published by Springer but discontinued after a few years) organized an expert discussion on "Languages For First Courses in Computer Science." The following five languages were included (the names of the experts are in brackets): APL (K.W. Smillie), Basic (S.J. Garland), Fortran (L.P. Meissner), Pascal (O. Lecarme (1984)), and PL/I (R. Conway).

In the epilogue of the discussion, Edwin D. Relly, taking into account the languages and their "dialects," voted for Pascal (with the UCSD dialect). For many years in the 1990s and later, Pascal was at the top of the list of programming languages used in education in schools and universities.

There were many opinions in the discussion that generally apply to learning programming, regardless of the language used. Garland (Basic) noted that "programming is a science as well as a skill and scientific methodology leads to good programs. In this broader view, learning to program means learning computer science, and this involves studying programming methodology, algorithms, and data structures"; "a programming

[3] Interestingly, in the book, Wirth used Pascal's notation, while in the article he used "a slightly augmented Algol 60 notation," although in the same year, 1971, he published an article introducing Pascal (Wirth, 1971b). This may indicate that Wirth "improved and corrected" some features of Algol 60 in Pascal, mainly for the needs of learning programming.

language should be the vehicle, not the object of instruction"; "simple tasks should have simple solutions"; "a good language reflects important concepts … minimizes inessential details"; "a good design breaks a problem and its solution down into logical components"; "a language should support a course, not dominate it." Lecarme (Pascal) stated that "The main philosophical idea is that computers are made for running our programs, and not the opposite … the notions of storage, CPU, peripherals are technical details … while the important things in programming are the notions of objects, which represent information, and actions, which manipulate these objects."; "students should understand the distinction between a course on programming and a course on a programming language."

In the next issue of the ABACUS magazine, five authors responded to the allegations of the remaining authors (pages 46–50), and in another article (Weiss, 1984) the attention was focused on the question of what a programming language is and what its role should be, especially in education. The answers to these questions are very important for teachers: *What role should a programming language play in teaching informatics?* And for students: *What role should a programming language play in learning informatics?*

In the discussion about the role of programming languages a question appeared: Is it a means for telling a computer what to do or a means for solving a problem? In other words: Is the purpose of programs to instruct a computer, rather than recognizing that the purpose of computers is to execute programs? Already in the mid-1970s, E.W. Dijkstra (1976) noticed that "the fact that programming languages could be used as a vehicle for instructing existing automatic computers, has for a long time been regarded as their most important property. The efficiency with which existing automatic computers could execute programs written in a certain language became the major quality criterion for that language!" Then he presented his own position: "I view a programming language primarily as a vehicle for the description of (potentially highly sophisticated) abstract mechanisms." Unfortunately, practitioners usually tell a computer what to do in order to solve a problem. Today's definition of a programming language goes beyond "a way to solve a problem," beyond "something in which you can write a program" and includes mainly a procedure used to direct a computer.

Our approach to use a programming language, in particular Pascal, in learning programming can be considered as a stage of algorithmic thinking, and in general problem solving with the help of computers, which today is theoretically and practically developed and extended to many areas with the support of mental tools and skills of computational thinking. In this sense, programming in the process of problem solving supported by algorithmic or computational thinking has a long history in informatics education in Poland (Sysło, 2023).

3. Pascal as a Programming Language in Informatics Education

The official beginning of informatics education in Poland is considered to be the approval by the MEN of the first informatics curriculum for the school subject *Elements of Informatics* for high schools in 1985, developed by the team of the Polish Information Processing

Society (PTI). The first textbook for Elements of Informatics appeared in 1988 and a package Elements of Informatics of 10 applications has been developed and delivered to high schools in 1992. Pascal played a central role in these activities as an object of teaching and learning, as well as a tool for creating various educational applications (see Section 4). In this section we describe these achievements with an emphasis on Pascal's role in them.

It is interesting to note that informatics as an autonomous school subject introduced officially in 1985 has remained in the national core curriculum for all these years until today.

3.1. *The Curriculum*

As mentioned in the preceding sections, the first curricula for teaching informatics were designed in the 1960s for schools in Wrocław and for future teachers of informatics who attended in-service training in Warsaw in the 1970s. The first curriculum for all high schools in Poland was approved by the MEN in 1985. The subject was called *Elements of Informatics* (EI) and it was assumed that at most 2 students will sit at a microcomputer in the classrooms. The curriculum was designed for 75 lessons (45 minutes each) and covered the following topics (numbers in parenthesis denote the number of lessons):

1. *The use of a microcomputer* (2) – first steps in using a computer and its software.
2. *Practical applications of a microcomputer* (6) – how to use applications for text editing, creating graphics and sounds, building tables and simple databases, and creating simulations.
3. *Drawing pictures on the screen* (4) – creation of simple elements of drawings, such as points, lines, curves, and use them to obtain more complex drawings.
4. *Procedures* (12) – elements of programming using procedures; recursion.
5. *Elements of programming style* (12) – elements of structural programming, procedures and structural relations between them; top-down programming.
6. *Non-elementary methods in creating graphics* (12) – the use of graphics procedures with parameters.
7. *Operations on texts (12)* – operations on texts as lists of characters.
8. *Individual problem solving (15)* – the use of acquired knowledge and skills in solving more advanced problems.

Like in any official government document, no specific software or application was named in the curriculum. However, in the most popular implementations, the Logo environment was used to implement Sections 4 and 7, and the Pascal environment to implement Sections 5 and 8. Moreover, various applications available in Polish were suggested to implement Sections 1 to 3 and 6.

3.2. *The Textbook*

The first textbook for the subject Elements of Informatics (under the same title) was written by a team supervised by the author as a product of the project RRI.16 sponsored by the MEN (Gurbiel *et al.*, 1989). It appeared in 1989 and then its unchanged edition was printed

every year until 1999. More than 100 000 copies have been sold. The book owed its long life mainly due to the approach adopted – computers and software tools were not described in full details but only with respect to the main theme (problem) of the presentation and discussion. The content of the book was (and still is) universal, and we are not surprised to see our textbook used by students and teachers even today:

1. What is Informatics. Elements of the history of computers and informatics.
2. How Computers are Designed and How They Work (operating systems).
3. Playing and Learning (Turtle graphics – Logo).
4. From Problem to Program (elements of programming in Turbo Pascal).
5. Designing One's Own Directory (database in Turbo Pascal).
6. Calculations in Mathematics (elements of numerical methods).
7. Computing Faster – efficiency of algorithms (elements of algorithmic complexity).
8. Writing With No Paper and Pencil – text editing.
9. Easy and Effective Managing Of a Small Business (spreadsheets).

The textbook was accompanied by a book with solutions to all the problems from the textbook and a guide book for teachers, which can be considered as a textbook on didactics of informatics. The package EI described in the next section was used in all three volumes as supporting software for students and teachers.

Regarding programming, Logo appeared in Chapter 3 in the process of designing and developing programs for Turtle graphics, and Turbo Pascal was utilized in Chapters 4 to 7 when solving various algorithmic problems in the way described in the title of Chapter 4 "From Problem to Program." Solutions to problems were gradually developed in a sequence of top-down refinement steps and finally programs were ready for running on a computer. All programs developed in each of the volumes accompanied the books on floppy discs.

3.3. *The Package of Educational Software*

The EI software package was designed and developed for the IBM PC under the MS DOS operating system from 1987–1992 by a team of almost 50 academic teachers from 4 universities under the supervision of the author in project RRI.14, sponsored by the MEN for supporting teachers of informatics and other subjects (Sysło, 1994). The software has been comprehensively tested and was accompanied by a documentation and educational materials (over 1 000 pages). 1 300 copies of the package were produced and sent to secondary schools across the country, as well as to over 50 universities where teachers were trained. The package consisted of 10 applications and educational systems designed and created in the Turbo Pascal language environment, either entirely (such as MET-NUM and MAT-STAT) or additionally with inline code:

1. SB – an application for constructing and executing flow-charts of algorithms.
2. DYSKIETKA – an application for presenting basic operations on a computer file system and demonstrating their effects.
3. TP-TOOL – a set of tools for supporting learning programming in Turbo Pascal.
4. ESO – an application for demonstrating OS commands and learning elements of concurrent programming.

5. DISC-MATH – a system consisting of 6 programs for supporting learning algorithms and data structures: operations on list and tree data structures, sorting algorithms, operations and algorithms on graphs, backtracking algorithms, a model of a universal computer (Turing machine).
6. EKO-SYM – an application for the simulation of eco-processes.
7. MAT-STAT – an application for supporting lessons on probability and statistics and also for estimating parameters based on experimental data.
8. PS-STAT – an application for analyzing experimental data addressed at non-specialists in other areas.
9. ASD – an application for demonstrating and analyzing algorithms and data structures.
10. MET-NUM – a system for supporting learning numerical methods (computer realization of mathematical calculations) and for performing numerical calculations.

As can be seen from the titles of applications, the package was designed for supporting learning and teaching not only of informatics (applications 1–10), but also mathematics (applications 5, 7, 9, and 10), statistics (applications 6, 7, and 8) and biology (application 6). Two screenshots are shown in Figures 1 and 2.

There was a general didactical idea/rule behind the design and development of the package. As most educational software, the package together with its very rich written educational materials, was used by teachers during the introduction of concepts while describing their properties. Then the students could follow with the package demonstrations of algorithms, their Pascal implementations, and the results of their execution. In this way students became gradually involved in using the package to learn how to solve and program some basic problems, thus implementing the vision of Seymour Papert (1980): the student programs the computer instead of the computer being used to program the student. The use of the EI package in informatics education, by teachers and students, can be shortly described in four stages:

1. a tool for demonstrating concepts, their properties, and algorithms;
2. a tool to performing experiments, mainly by students, with different parameter values, in learning algorithms and testing programs in Pascal;
3. an environment for students to plan and design their own exercises supported by the package; and
4. an environment for students to write their own programs in Pascal using some software modules available in the package.

From today's point of view, the EI package appears ahead of its time – teachers were not prepared and trained to use it in algorithmics and programming. After several years, some of the applications from the package have been redesigned and implemented for the Windows operating system and now are available as open educational resources.

3.4. *Pascal in Other Educational Activities*

The popularity and uniqueness of Pascal as the language of algorithmics and broadly understood problem solving resulted in its popularity as the language chosen by students

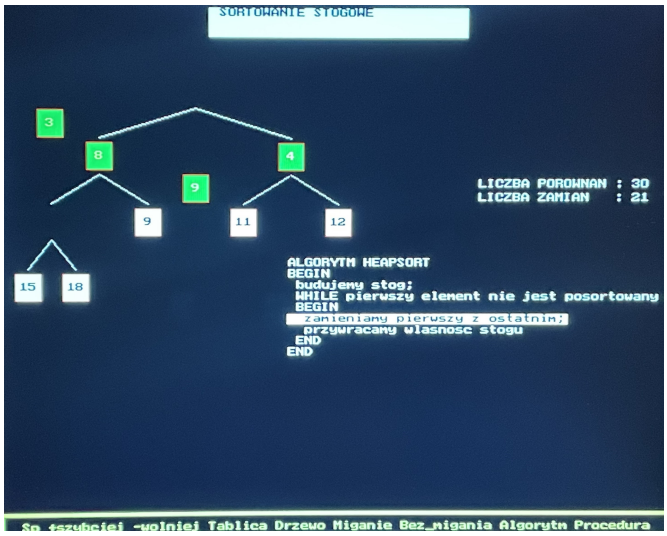

Fig. 1. A module SORTING in DISC-MATH – A demonstration of sorting by HeapSort on a tree

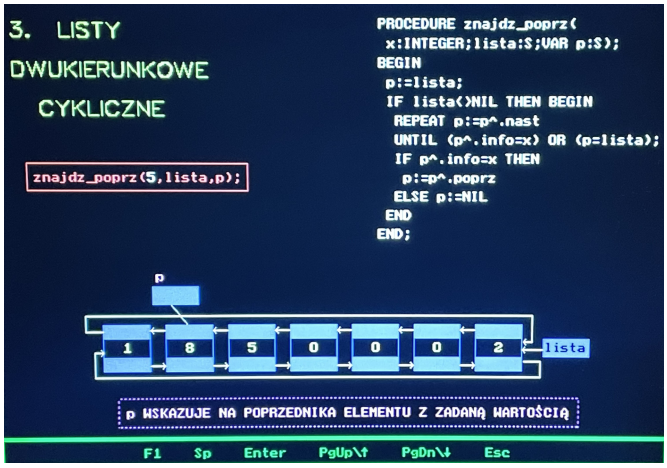

Fig. 2. A module LISTDATA in DISC-MATH – A demonstration of searching an element in a cyclic two-way list

at the high school final examination (“Matura” in Polish) in informatics and the language used by students in the *Olympiad in Informatics* (OI).

Pascal was one of the languages in which students could program solutions in the first Matura exam in 2005 (when a new form of Matura exams was introduced in Poland) until 2023. From 2024 on, however, Pascal officially disappears from the Matura exam. At the beginning it was the Turbo Pascal 7 edition, and in recent years *Free Pascal*. On the one hand, it has survived for so long, probably thanks to informatics teachers who once learned this language during training and continued classes in this language with their students. On

the other hand, the enormous popularity of Python resulted in the abandonment of Pascal, which stopped developing.

For 24 years, Pascal was also one of the programming languages of the national OI and was withdrawn only before the 25th edition in the school year 2017/2018, In the international OI in 2018, solutions to tasks written in Pascal were accepted, similarly in the Baltic OI and in the Central-European OI (until 2017). Over the years, Pascal was increasingly replaced by C++ in informatics competitions as a language more suitable for algorithmic programming and generating faster implementations.

Unfortunately, for several years now, Pascal has no longer appeared on the lists of the most popular programming languages among programmers, or as a possible language for teaching programming.

3.5. *Pascal in the Hands and Opinion of a Brilliant Programmer*

At the end of this section, let us share Andrzej Gąsienica-Samek's opinions about programming languages, in particular about Pascal. In the second half of the 1990s, Andrzej was one of the most talented and most successful programmers of the young generation in Poland. At the age of 7, he was still playing with the Basic language on Atari, but in the early 1990s he started programming in Pascal after his father acquired an IBM PC. He achieved his greatest successes in national and international competitions and OIs in 1996-1999, when he was a high school student. In the national OI he was Polish champion three times (in 1996, 1998, 1999) and once runner-up (1997). At the international OI he won three gold medals (1997 in South Africa, 1998 in Portugal, and 1999 in Turkey) and one silver medal (1996 in Hungary). He was also the *Champion of the Computing Olympiad* (Winter Championship) in the US twice, in 1997 and 1998. Additionally, he won the highest trophies in the Baltic OI and the Central-European OI. Andrzej was also successful with the University of Warsaw team in the ICPC competition, using C++, which currently dominates in OI and in other informatics competitions.

In Andrzej's opinion, there was practically no need to use pointers in Pascal. The array range control was very useful in particular for beginners, and this is something that does not exist in C++ so far. He believes that there is no better language than Pascal for presenting the basics of structured programming, because in C++ we have to get down to low-level concepts such as pointers too quickly. In practice, as long as we do not introduce dynamic memory management, arrays are sufficient, as in Pascal, for simple implementation of many algorithms.

He stated that he thinks that Pascal was perfect for OI problems, which were in some sense tailored to the language. Since C++ is at the forefront, balanced trees appeared in the solutions of OI tasks, and Pascal without an update (which is not available) became insufficient. Moreover, Pascal was not available for Linux, which was becoming the standard, as MS DOS was for Turbo Pascal. Pascal was no longer developed, and libraries began to determine the greater usefulness of C++. In practice, in C++ it was possible to use data structures that used dynamically allocated memory without worrying about memory allocation. In Pascal, data structures remained at the record and array levels.

Currently, in Andrzej's professional work as a programmer, the choice of programming language and working environment for solving a problem depends on functional and platform requirements. Today's popular web projects use at least two languages simultaneously for backend and frontend purposes, not to mention SQL, which is still the only sensible language for database queries.

4. Pascal as a Tool

4.1. *A School Computer – the Elwro 800 Junior and Its Software*

In the mid-1980s schools and universities were buying microcomputers to build their first computer labs and to offer students regular classes on informatics and on other subjects. Since the most popular was the ZX Spectrum, the MEN decided to organize a competition for a Polish school microcomputer which was to meet the following two requirements at least:

- to work in one of its modes as the ZX Spectrum, so that the rich software available in Poland for the ZX Spectrum could be used;
- to work under the control of a disc operating system, so that the software working under CP/M could be used.

The winner was the *Elwro 800 Junior* ("Junior") designed by the team from the Poznań Polytechnic, supervised by Wojciech Cellary and Paweł Krysztofiak (Cellary and Krysztofiak, 1988). The most important feature of Junior was its operating system CP/J compatible with CP/M, and moreover CP/J could control a local network *Junet* consisting of up to 15 microcomputers of type Junior or ZX Spectrum. Students' Juniors didn't have disc drives, but a teacher's Junior had an external disc drive as a server and a printer. It was a very economic and effective design – all students working in Junet had access to the discs and printers.

The ability of the Junior to work in ZX Spectrum mode and under the supervision of the CP/M (CP/J) system enabled the usage of almost all programs and environments developed for 8-bit microcomputers, including office applications, games, and programming languages such as Logo and Turbo Pascal (Bielecki, 1988). Moreover, the popularity of the ZX Spectrum and the CP/M operating system also contributed to the individual activities of non-professional programmers, creating many small and large programs and applications, including those intended for individual and school education.

Let us mention here JU-LEK, one of the most advanced systems built for Junior. It was implemented in Turbo Pascal 3.02A with the additional graphics library GSJ 0.0 (designed by K. Pielesiak from the Poznań Polytechnic). JU-LEK was an authoring tool, designed and implemented by Witold Rudolf (1990) in his master thesis, supervised by Ewa Gurbiel (Institute of Computer Science, University of Wrocław) in 1989. This tool was supposed to be used for preparing and executing lessons of presentation type (PT CAI – Presentation Type Computer Assisted Instruction). Lessons in JU-LEK reminded of today's PowerPoint

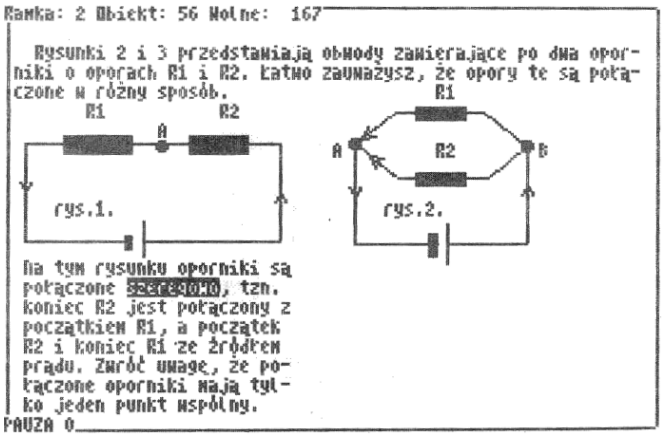

Fig. 3. A sample frame from the authoring system JU-LEK

presentations. It was used to prepare lessons in various subjects – see a frame from lesson in physics in Figure 3.

Lessons made in JU-LEK consisted of at most 99 frames (or screens) connected (ordered) according to a lesson plan (graph). Each frame could contain up to 150 objects such as texts, graphics, and commands (delete, break, repeat, pointer, answer, and run an external program in Turbo Pascal). Graphical objects could have the attributes type of a line, color, and intensity of a color.

JU-LEK consisted of two sets of modules: one for a teacher (an author) and one for students. The author's set contained two main modules, GEN – for building a lesson, and EXE – for executing (trying) a lesson created by GEN. Students were interactively executing lessons built in JU-LEK.

JU-LEK and its author were awarded the III Prize in the 1990 *National Competition for Master Degree Thesis in Informatics*. The system was purchased by the OFEK Foundation, which published its guide book and then disseminated the tool to schools equipped with Juniors.

In the beginning of the 1990s, when the era of 8-bit microcomputers was coming to an end, schools were looking for IBM PCs and the MEN stopped to buy Juniors and sending them to schools. Almost 10 000 Juniors were installed in almost 1 000 schools. In 1993, 60 % of computers in schools were still 8-bit computers and more than half of them were Juniors.

4.2. *Educational Software*

The software for Junior (SO CP/J, Logo, Turbo Pascal, JU-LEK) formed the basic computer environment for informatics lessons conducted according to Elements of Informatics (see Section 3.2). We should admit today that introducing the Elwro 800 Junior microcomputer to schools in the second half of the 1980s created opportunities for thousands of students to make the first steps in learning how computers work and how to use them in a number

of ways, in particular in writing their first programs in Basic, Logo, or Pascal. The lessons learned, by both the authors and the users, were a big step towards the future in the environment of new and rapidly changing technology.

At that time, in many areas of education, teachers themselves created many educational applications and systems, mainly for their classes and for their students, for example, chemists (Burewicz *et al.*, 1992) to simulate experiments, and mathematicians (Sepko-Guzicka and Guzicki, 1987; Jakubas, 1994–1998) to better explain and visualize mathematical concepts and perform some sample calculations to demonstrate behavior of mathematical formulas and problem solutions. This was due to the simplicity of the environments in which they created, such as Basic, Logo, or Turbo Pascal. Today it would be difficult to repeat those experiences.

References

Bielecki, J. (1988). *Turbo Pascal. Oprogramowanie Elwro 800 Junior*. WNT, Warszawa.

Burewicz, A., Gulińska, H., Miranowicz, N., Szmidt, H. (1992). *Edukacyjne Programy Komputerowe w Nauczaniu Chemii*. OFEK, Jelenia Góra.

Cellary, W., Krysztofiak, P. (1988). *Elwro 800 Junior*. WNT, Warszawa.

Dijkstra, E.W. (1976). *A Discipline of Programming*. Prentice-Hall, Englewood Cliffs, NJ.

Gurbiel, E., Kołczyk, E., Krupicka, H., Łukojć, K., Płoski, Z., Sysło, M.M., Witkowski, J., Zuber, R. (1989). *Elementy Informatyki, Vol. 1–3*. Wydawnictwo Uniwersytetu Wrocławskiego, Wrocław.

Iglewski, M., Madey, J., Matwin, S. (1978). *Pascal. Język wzorcowy i Pascal 6000*. WNT, Warszawa.

Jakubas, E. (1994–1998). Matematyka z Komputerem, Zamość.

Kucharczyk, J., Sysło, M.M. (1975). *Algorytmy Optymalizacji w języku Algol 60*. PWN, Warszawa.

Lecarme, O. (1984). Pascal. Languages for First Courses in Computer Science. *ABACUS*, 1(4), 58–66. Epilogue 1(4):75–79, Rebuttals, 2(1):46–50.

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, New York.

Paszkowski, S. (1965). *Algol 60*. PWN, Warszawa.

Rudolf, W. (1990). System Autorski JU-LEK. *Informatyka*, (8), 18–21.

Sepko-Guzicka, E., Guzicki, W. (1987). *Zastosowanie Komputerów w Nauczaniu Matematyki*. IKN, Warszawa.

Sysło, M.M. (1994). Pakiet 'Elementy Informatyki'. *Komputer w Edukacji*, (1), 15–25.

Sysło, M.M. (1997). *Algorytmy* (1st ed.). WSiP, Warszawa.

Sysło, M.M. (2014). The First 25 Years of Computers in Education in Poland: 1965–1990. In: Tatnall, A., Davey, B. (Eds.), *History of Computers in Education*. IFIP Advances in Information and Communication Technology*: Vol.* 424. Springer, Heidelberg, pp. 266–290.

Sysło, M.M. (2016). *Algorytmy* (2nd ed.). Helion, Gliwice.

Sysło, M.M. (2023). Computer Science Education with a Computer in the Background. In: *Local Proceedings of the 16th International Conference on Informatics in Schools (ISSEP 2023)*, pp. 89–101.

Sysło, M.M., Deo, N., Kowalik, J.S. (1983). *Discrete Optimization Algorithms with Pascal Programs*. Prentice Hall, Englewood Cliffs, NJ.

Weiss, E.A. (1984). Programming Language Survey: Two Approaches. *ABACUS*, 2(1), 51–57.

Wirth, N. (1971a). Program Development by Stepwise Refinement. *Communications of the ACM*, 14(4), 221–227.

Wirth, N. (1971b). The Programming Language Pascal. *Acta Informatica*, 1(1), 35–63.

Wirth, N. (1976). *Algorithms + Data Structures = Programs*. Prentice-Hall.

Wirth, N. (1980). *Algorytmy + Struktury Danych = Programy*. WNT, Warszawa. Translated from Wirth, N. (1976).

Zuber, R. (1966). O Realizacji Przedmiotu 'Programowanie i obsługa maszyn cyfrowych'. *Matematyka*, 19(2), 76–83.

Zuber, R. (1972). *Metody Numeryczne i Programowanie*. PZWS, Warszawa.