MODESTUM
OPEN ACCESS

EURASIA Journal of Mathematics, Science and Technology Education, 2018, 14(5), 1705-1718
ISSN:1305-8223 (online) 1305-8215 (print)
Research Paper https://doi.org/10.29333/ejmste/85109

A Multidisciplinary Approach to Teaching Mathematics and Information and Communication Technology

Vladimír Jehlička [1*], Ondřej Rejsek [1]

[1] University of Pardubice, Faculty of Transport Engineering, CZECH REPUBLIC

Received 6 October 2017 ▪ Revised 2 January 2018 ▪ Accepted 6 January 2018

ABSTRACT

The submitted article supports the multidisciplinary approach to STEM (Science, Technology, Engineer, Mathematics) teaching. Pupils in many countries are showing a reduced interest in STEM studies. Should we wish to increase their interest in STEM studies, then we must increase their interest in the study of mathematics. This article concentrates on providing an illustrative demonstration of the interdisciplinary relations between mathematics and information and communication technology (ICT) that can readily be used in teaching these two subjects at primary and secondary schools. The interdisciplinary teaching approach described promotes not only a better understanding of the study of mathematics but it also develops pupils' logical reasoning and critical thinking. The need to interconnect mathematics and ICT in teaching is documented through specific examples. Attention is focused on number translation between individual numerical systems and entering numbers in the memories of computers and calculators.

Keywords: ICT, mathematics, multidisciplinary teaching, STEM

STUDENTS' INTEREST IN SCHOOL SCIENCE AND TECHNOLOGY

We live in a world in which many technical products surround us. Young people, in particular, cannot imagine living their lives without having the latest technical innovations at hand on a daily basis. Yet, at the same time, pupils are showing a reduced interest in technical subjects and this trend is clearly visible worldwide.

For example, Potvin and Hasni (2014) discuss the decline in interest towards school science and technology in Canada. Decreasing interest in studying STEM at secondary schools are dealt with by Meng, Idris and Eu, (2014) from Universiti Sains Malaysia. Sellami, El.Kassem, Al-Qassass and Al-Rakeb (2017) from Qatar University, Doha, Qatar, are devoting research to students' interest in STEM. Issues of STEM are addressed by Martin, Im and Song (2016) from Seoul National University, Seoul, Republic of Korea. Oliveira and Barufaldi (2009) from The Center for Science and Mathematics Education at The University of Texas at Austin USA have set themselves the objective to improve the scientific literacy of students. They emphasized the essentiality of adopting an interdisciplinary approach to studies, in their astrobiology course. Rosca and Todasca (2014), from the University of Bucharest, are of the opinion that the reason why Romanian students are less and less interested in studying chemistry lies in the weak correspondence between teaching chemistry and other subjects, such as physics, biology and mathematics. Education reforms in the STEM field are dealt with by Eilks and Markic (2014) from the University of Bremen, Germany.

Students' interest in school science and technology can be boosted in several ways. Brunton and Coll (2005), from New Zealand, prefer to create close links between schools and the industry. They use a specific case to draw attention to a serious issue. The development of cooperation between schools and the biotechnological industry is hindered by teachers who have an insufficient knowledge of biotechnology. Ardies, de Mayer, Gijbels and van Keulen (2015), from Belgium and Holland, draw attention to the influence teachers, parents and technical toys in homes have on pupils' interest in the study of technical subjects. Various education games can be used to motivate pupils. For example, Klisch, Miller, Wang and Epstein (2012), from the University of Houston, USA, describe a game that simulates environmental pollution caused by toxic chemicals and it simulates the environmental impact

✉ Vladimir.Jehlicka@upce.cz **(*Correspondence)** ✉ Ondrej.Rejsek@upce.cz

Contribution of this paper to the literature

- This article contributes to increasing pupils' motivation to study STEM and is aimed at demonstrating a multidisciplinary approach to teaching mathematics, computer science and programming.
- The article provides instructions for linking mathematics teaching (entering numbers in different numerical systems) with teaching of algorithmization and programming (creating small sample programs with simple mathematical computations).
- The article can be used by mathematics and computer science teachers to create additional motivational tasks for the individual experimentation of pupils for counting with calculators, smartphones, tablets, laptops and personal computers.

less common scents from perfumes have when disseminated. Lorenzo (2005) describes science teaching in Albacete, Spain, based on problem-solving.

Another teaching method is based on computer modeling and simulation processes. A global warming computer model, which is used by teachers in Holland, is described by van Borkulo, van Joolingen, Savelsbergh and de Jong (2012). Matsumoto (2014) describes simulation calculations in chemistry courses at a high school in San Francisco, USA, with the use of Mathematica, a computer algebra system.

When it comes to the teaching of school science and technology, it is essential that emphasis be placed on a multidisciplinary approach to teaching and on the integration of individual branches of science. The multidisciplinary teaching of 6th grade primary school pupils in Soul, Korea, is described by Chai and Chun (2015). Here teaching concentrates on developing creative thinking in combination with art studies and STEM (Science, Technology, Engineer, Mathematics) education. Multidisciplinary teaching integrating the knowledge of physics, mathematics, electrical engineering and computer technology in order to create an idea about the anatomy and the functioning of the human body is applied at the University of Miami, Florida, USA (Debebe, Bhatt & McGoron 2013). The multidisciplinary approach to teaching information systems at South African higher education institutions (Turpin, Matthee & Kruger 2015) places emphasis on the development of student creativity in the creation of information systems. An interdisciplinary interconnection between mathematics and physics in the teaching of mechanical engineering at universities in Argentina (Tinnirello, Gago & Dadamo 2013) is applied in the creation of simulation models and virtual laboratories. Němec, Berková and Hubálovský (2016) describe a modeling and simulation multidisciplinary approach to teaching at the University of Hradec Králové in the Czech Republic. Rey, Dios, Sandia & Martinez-Azua (2014) endeavor to affect the multidisciplinary teaching of mathematics, statistics, macro-economy and sociology in Spain. Mathematics constitutes an important part of teaching bioengineers in Argentina (Merino, Anino & Ravera 2013). Matlab programming courses at the University of Valencia, Spain, are taught on the basis of applications that focus on physics, mathematics, engineering, etc. (Soria-Olivas, Martín-Guerrero, Martínez-Sober & Avuso-Moya 2010). Carreras-Marin et al. (2013), authors from Barcelona, present the multidisciplinary links between mathematics, world economy and the history of economy. Authors from the University of Pardubice, Czech Republic (Juránek, Machalík & Zemčík 2011), have demonstrated a link between mathematics, ICT and mechanical engineering on the basis of an analysis of the wear and tear of equipment, such as combustion engines, transmissions, and the like. The multidisciplinary inclusion of mathematics and algorithm development in teaching mechanical engineering leading to the development of mathematical models, simulation and visualization is described by Tinnirello et al. (2012), authors from Buenos Aires, Argentina. Kuna and Turčáni (2012), authors from Nitra, Slovakia, focus on the interdisciplinary link between mathematics and ICT and they research the impact of the knowledge of mathematics on the teaching of ICT. Jehlička (2013, 2010), an author from the University of Hradec Králové, Czech Republic, pursues an interdisciplinary approach to the teaching of algorithm development and programming. The integration of the teaching of mathematics in other disciplines in the context of training engineers in Bossier City, USA, is discussed by Coriell and Corbett (2013). The integration of mathematics into teaching other science disciplines is described by Alghamdi (2017) from the University of Damman, Saudi Arabia. Improvements in the level of mathematical education of future engineers are dealt with by Konysheva and Ibragimova (2017) from Vyatka State University, Russia.

It is obvious that in order to succeed in the study of school science and technology, a sound knowledge of mathematics is an absolute necessity. Yet, the fundamental problem remains pupils' motivation to study mathematics. The education systems in East Asia, Australia, England, the Netherlands and USA are discussed by Zhu and Leung (2011) from the University of Hong Kong, specifically in regards to improving pupils' interest in math's studies. In order to succeed in their architecture studies, it is essential that students have a sound understanding of the basics of mathematics, as supported in their paper by Moratalla de la Hoz and Sanz Garcia (2010) from the University of Madrid, Spain. Studying mathematics is, however, a difficult and unpopular task for many pupils and it is of little interest to them. Doorman, Drijvers, Gravemeijer, Boon and Reed (2012), from the

Utrecht University, the Netherlands, draw attention to the difficulties associated with explaining functions in secondary school math's lessons. If pupils are to find their math's lessons interesting, the math's teacher needs to show them how mathematics is interconnected with various day-to-day situations. In his paper, Foster (2015) from The University of Nottingham, England, notes that it is essential for math's teachers to improvise during math's lessons and for them to be able to react to all sorts of their pupils' contributions. How to determine the level of knowledge reached when it comes to the teaching of Numerical Mathematics at the Jan Perner Transport Faculty, University of Pardubice, Czech Republic is discussed by Kulička and Berková (2015) and Kulička (2014). Czech primary school pupils have shown a tremendously sharp drop in their knowledge of mathematics over the past years under PISA (Programme for International Student Assessment) (OECD, 2016). Pupils usually find this subject to be difficult, dull and of little use in practical life. This may be due to an insufficient number of primary school math's lessons, and the fact that mathematics is linked to only a few other subjects. For example, it is clear that teaching physics is impossible without a basic knowledge of mathematics. Nor should the teaching of ICT, for that matter, go without a close link to the teaching of mathematics. However, primary school pupils and secondary school students have a very low awareness of the interconnection between mathematics and ICT.

THE MULTIDISCIPLINARY LINKS BETWEEN THE TEACHING OF MATHEMATICS AND ICT

A basic knowledge of mathematics for the work of future IT experts is underestimated when it comes to the teaching of the basics of ICT, and in many cases also when it comes to training future ICT teachers. We often come across the opinion that it suffices if pupils are able to work with commercially available programs, if they learn how to make use of all of the programme options, if they learn how to create a website, process visual and audio information, etc. Put simply, mathematics is allegedly not all that important for IT experts. The following text should contribute to rebutting this myth.

Defining the Research Questions

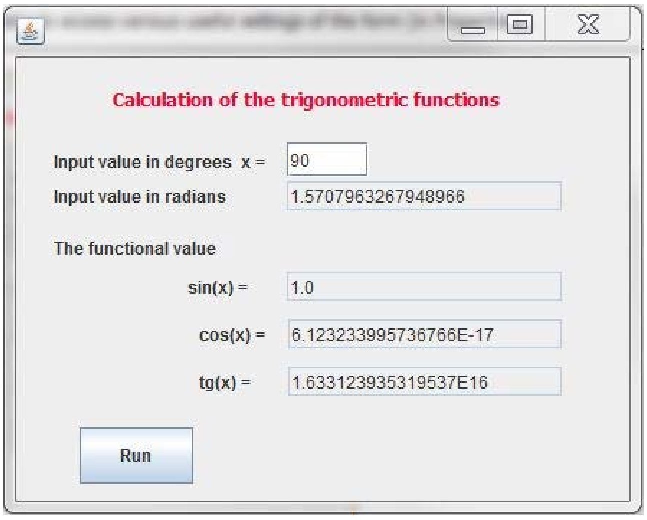
ICT should be understood as being a very extensive and complicated system of searching for, classifying, follow-up processing and evaluating various types of information. Given how widespread computer technology is today, the majority of relating operations are carried out on computers. Information is stored in computer memories and it is then processed in the form of suitable data. In order to do this, standard mathematical and logical operations are applied. Primary school pupils and secondary school students usually quite automatically assume that all calculations that can be made manually using pen and paper can also be made on digital computers – and that much faster and more precisely. In many cases, however, this does not apply. It is the very teachers of ICT that should be aware of the various difficulties and restrictions computer technology poses so that they can draw their pupil's attention to these issues. They should also be able to explain why such issues arise based on the knowledge of mathematics necessary and they should clearly explain to their pupils that without mathematics, there is no ICT.

The basic research question: Would it be possible to identify simple illustrative examples demonstrating the link between mathematics and ICT that would motivate primary school pupils and secondary school students to want to study mathematics more?

Methodology of Research

First of all, simple calculation methods need to be identified that while using computer technology do not provide the results expected by pupils based on their preceding study of mathematics. Then, the cause of the problems encountered needs to be defined. This can be followed by an interpretation of simple, basic mathematics-related information that can be used to justify the foregoing questionable calculations. Finally, pupils need to be shown that without a sound knowledge of mathematics, it is not possible to mindlessly accept just any sort of calculation generated by computer technology.

Specific examples of simple calculations will then be used to demonstrate the necessity of adopting a multidisciplinary approach to teaching ICT, which would not be possible without a basic knowledge of mathematics. Thus, both subjects create an integral and inseparable interdisciplinary system. In the chapters to follow, emphasis is primarily placed on the need of mastering number translation between the decimal and binary numeral system. This is then followed by the method involved in storing the numbers into the memories of computers and rounding them off. Specific examples are then used to demonstrate questionable calculations the cause of which lies in the very storing of the numbers in the computer's memory. Without having the necessary knowledge of mathematics, a future teacher of ICT would not be able to explain these seemingly illogical results of calculations made using calculators or computers to his pupils.



Calculation of the trigonometric functions

Input value in degrees x = 90

Input value in radians 1.5707963267948966

The functional value

sin(x) = 1.0

cos(x) = 6.123233995736766E-17

tg(x) = 1.633123935319537E16

Figure 1. Calculation of the functional values sin 90°, cos 90° and tg 90° using a program created in JAVA

PROBLEM CALCULATIONS

Primary school pupils and secondary school students frequently use pocket calculators and mobile phones to make routine calculations. Higher graders then use personal computers, notebooks and tablets with standard software, which includes Microsoft Excel, for example. Based on the mathematics taught at second stage primary schools, pupils know, for example, that

1. $\sin 90° = 1$,
2. $\cos\ 90° = 0$,
3. tg 90° is not defined.

It is clear that they can verify this information with the help of computer technology. For illustrative purposes, we will show the results given by CASIO *fx-991MS,* a pocket calculator. The expected numeric values appeared on the display in the first two cases, but in the third case *Math ERROR* was displayed. Similar calculations can also be made on other types of calculators. For example, when using the Citizen SR-59 calculator, the first two calculations were correct and the error in the third case was reported as an *E*.

Figure 1 shows the result of the calculation of the discussed functional values of the trigonometric functions; a program that was created in JAVA, a computer programming language, was used for this purpose. Furthermore, all of the hereinafter referred to programs were also created using JAVA (Herout, 2010). Note that the first value corresponds to theoretical assumptions, but the second and the third do not.

What is the problem? Where did the error occur? In order to understand this phenomenon, we first need to focus on issues surrounding the storing of numbers into the memory of a computer and therewith associated number translations between the decimal and binary numeral system.

THE DISPLAY OF NUMBERS IN THE MEMORIES OF POCKET CALCULATORS AND COMPUTERS

In their math's lessons, primary school pupils learn about number translations between individual numerical systems. They are taught what the basis of a numeral system is, and how a numeric value can be expressed by the multiplication of the suitable powers of the base and the addition of the obtained intermediate results. At the same time, these pupils learn in their ICT classes that all information is stored in computers in the form of zeros and ones. That means that numbers are stored in a binary system.

Number Notation in Various Numerical Systems

As an example, the decimal number 456 can be expressed in a decimal, binary, octal and hexadecimal numeral system. Let us first write this number down using suitable powers of the bases of the individual numerical systems.

$$456 = 4.10^2 + 5.10^1 + 6.10^0 = 400 + 50 + 6$$

$$456 = 1.2^8 + 1.2^7 + 1.2^6 + 0.2^5 + 0.2^4 + 1.2^3 + 0.2^2 + 0.2^1 + 0.2^0 = 256 + 128 + 64 + 8$$

Table 1. Number notation in various numerical systems

Numeral system				Roman numerals
Decimal	Binary	Octal	Hexadecimal	
0	0	0	0	
1	1	1	1	I
2	10	2	2	II
3	11	3	3	III
4	100	4	4	IV
5	101	5	5	V
6	110	6	6	VI
7	111	7	7	VII
8	1000	10	8	VIII
9	1001	11	9	IX
10	1010	12	A	X
11	1011	13	B	XI
12	1100	14	C	XII
13	1101	15	D	XIII
14	1110	16	E	XIV
15	1111	17	F	XV
16	10000	20	10	XVI
17	10001	21	11	XVII
...	...	...	...	...

$$456 = 7.8^2 + 1.8^1 + 0.8^0 = 448 + 8$$

$$456 = 1.16^2 + 12.16^1 + 8.16^0 = 256 + 192 + 8$$

Before we proceed to write down the number in the above mentioned numerical systems, we need to realize that we have ten numerical symbols (0, 1, 2, ..., 9) that can be used to express numbers in the decimal system, two numerical symbols (0, 1) in the binary system, eight numerical symbols (0, 1, 2, ..., 7) in the octal system and sixteen numerical symbols (0, 1, 2, ..., 9, A, B, C, D, E, F) in the hexadecimal numeral system. As soon as we exhaust the applicable symbols, in the given order, the notation must be made in a higher order. Refer to **Table 1**. In the last column, the numbers are expressed in Roman numerals, only for a matter of interest. It is clear that the notation is absolutely unsuitable when it comes to mathematics.

Now it is simple to write the given number in the individual numerical systems, where the index states the value of the basis of the given numeral system.

$$(456)_{10} = (111001000)_2 = (710)_8 = (1C8)_{16}$$

Pupils learn that other numerical systems can also be created using the above described principle, at a random base. Thus, they can write the assigned number in a quinary or senary numeral system, for example. This part of the interpretation of mathematics very much develops the abstraction and generalization skills of primary school pupils. But it usually comes across to them as an autotelic game with numbers that is in no way related to daily life. But the opposite is true.

Storing Numbers in Computer Memories

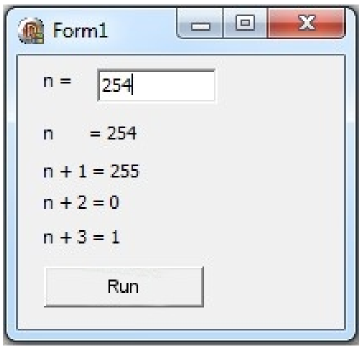
Digital computer technology operates with memories the smallest unit of which is one bit, into which we can store information only of the Boolean type (denoted True and False), thus represented as either a 1 or 0. It follows from the aforesaid that numbers can be stored into a computer memory only in a binary system. Pupils then quite logically ask why then should they concentrate on octal or even hexadecimal numerical systems? The answer is obvious. A computer does in fact operate in a binary system, but the notation is very unclear. If we divide a number entered in a binary system into groups of three or four numbers, then we can quite logically also enter the given number in an octal or hexadecimal system.

$$(456)_{10} = (111 \quad 001 \quad 000)_2 = (710)_8$$

$$(456)_{10} = (1 \quad 1100 \quad 1000)_2 = (1C8)_{16}$$

The intention to simplify the notation and make clearer the reading of numbers written in a binary system is the reason why we in some cases use a hexadecimal system (Jehlička, 2010). For example, if we use a memory with a 4 byte capacity to enter an integer, then we store a, for us, very poorly arranged number that contains 32 zeros or ones. Much more transparent is its notation in the hexadecimal system

$$(1111 \quad 0000 \quad 0111 \quad 0101 \quad 1001 \quad 1100 \quad 0110 \quad 0000)_2 \;=\; (F0 \;\, 75 \;\, 9C \;\, 60)_{16}$$



n = 254

n = 254

n + 1 = 255

n + 2 = 0

n + 3 = 1

Figure 2. An example of an overflow when working with a Byte type variable in Visual Basic

Storing positive integers in computer memories

If we want to store a positive integer in a computer memory, we need to make a space in the memory that contains as much bits as is necessary for the given number to be expressed in a binary system.

Let's start with a simple example, where we delimit only 8 bits, which is 1 byte, in the computer memory. A number ranging from $(0000\ 0000)_2$ to $(1111\ 1111)_2$ can be stored in this space using the binary system. Numbers, in a decimal system, ranging from $(0)_{10}$ to $(255)_{10}$ correspond to this.

If we were to create programs in Visual Basic, for example, then we will declare the respective variable as a variable of the Byte type. It is, however, necessary to realize the rigors of such a declaration. A greater numerical value really cannot be stored in the variable. If we were to, for example, gradually add the value 1 to the respective variable, this would lead to a maximum value of $(1111\ 1111)_2 = (255)_{10}$ and by adding the number one $(0000\ 0001)_2 = (1)_{10}$ we would cause a so-called overflow, in which case zeros will be stored in all of the delimited bits and the respective number one should be entered in a higher order which, however, cannot be displayed because it is beyond the scope of the delimited memory for storage of the Byte type variable. It follows that a zero, the resulting value, will be stored in the given variable: $(1111\ \ 1111)_2 + (0000\ \ 0001)_2 = (0000\ \ 0000)_2$.

A simple program can be created in Visual Basic to substantiate this fact, upon declaring a Byte type variable. The variable will be in the range of $\langle 0;\ \ 255\rangle$. An example of an output of the program is provided in **Figure 2**.

Storing negative integers in computer memories

Negative numbers can be expressed in various ways in computer memories. One of the methods is to reserve the highest order of a binary number for the identification of the sign of a positive or negative number and the number is stored in its absolute value in further bits. A disadvantage of this type of viewing is the fact that there is not only a *positive* zero $(0000\ \ 0000)_2$, but also a *negative* zero $(1000\ \ 0000)_2$.

The second option would be to express negative numbers with a so-called shifted zero. In which case the decimal number -128 may be allocated to the value $(0000\ \ 0000)_2$. A decimal zero corresponds to the value $(1000\ \ 0000)_2$ and the decimal number 127 corresponds to the value $(1111\ \ 1111)_2$.

Nonetheless, negative numbers are most often stored in computer memories in two's complement form, which is created by taking the absolute value of a negative number, expressed in a binary system, creating its negation and then adding a one to it. For illustrative purposes, let's try to enter the decimal number -5 in two's complement form.

$(0000\ \ 0101)_2$ notation of the absolute value of the number, that being 5

$(1111\ \ 1010)_2$ negation of the notation – zeros are replaced by ones and vice versa

$(1111\ \ 1011)_2$ number one is added = resulting notation in two's complement form.

By analogy, we can enter a random negative integer, provided we have delimited a sufficient size memory capacity. This negative number notation method removes difficulties associated with the ambiguous notation of zeros as well as difficulties associated with a shifted zero.

Should we be interested in a back transfer, meaning determining the absolute value of a negative number, then we first subtract a one from the number expressed in two's complement form and then we negate it. Thus, let us determine the absolute value of the number $(1111\ \ 0111)_2$, for example.

$(1111\ \ 0111)_2$ negative number in two's complement form

Table 2. Notation of numbers in two's complement form in a memory with the size of 4 bits

Decimal number	Two's compliment	Decimal number	Two's compliment
-8	1000	0	0000
-7	1001	1	0001
-6	1010	2	0010
-5	1011	3	0011
-4	1100	4	0100
-3	1101	5	0101
-2	1110	6	0110
-1	1111	7	0111

Table 3. An example of adding up and subtracting numbers in a binary system

5 + 2 = 7		5 – 2 = 3		– 5 – 2 = – 7	
5	0000 0101	5	0000 0101	– 5	1111 1011
2	0000 0010	– 2	1111 1110	– 2	1111 1110
5 + 2 = 7	0000 0111	5 – 2 = 3	0000 0011	– 5 – 2 = – 7	1111 1001

Table 4. Range of integers that can be entered in computer memories varying in size

Memory range	Number range (decimal)	Type of variable	
		JAVA	Visual Basic
1 byte = 8 bit	$\langle 0; 255\rangle$		Byte
	$\langle -128; 127\rangle$	Byte	
2 byte = 16 bit	$\langle -32\,768; 32\,767\rangle$	Short	Short
4 byte = 32 bit	$\langle -2\,147\,483\,648; 2\,147\,483\,647\rangle$	Int	Integer
8 byte = 64 bit	$\langle -2^{63}; 2^{63}-1\rangle$, i.e. approximately $\langle -9{,}223.10^{18}; -9{,}223.10^{18}\rangle$	Long	Long
16 byte = 128 bit	$\langle -7{,}9228.10^{28}; -7{,}9228.10^{28}\rangle$		Decimal

$(1111\ \ 0110)_2$ result after subtracting the number one

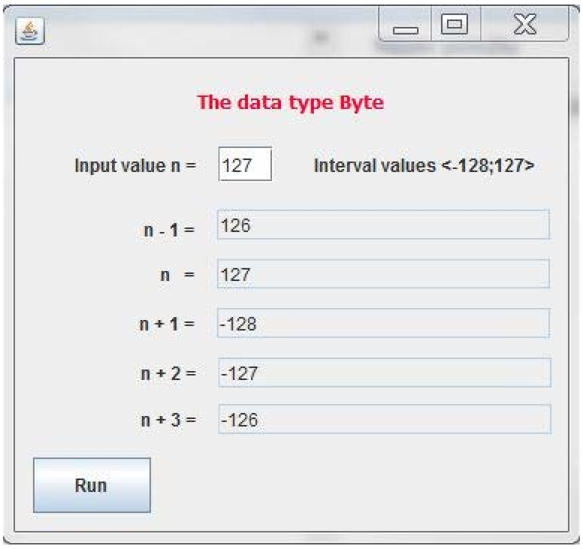
$(0000\ \ 1001)_2$ negation = resulting absolute value = $(9)_{10}$.

For illustrative purposes, we can draw up a **Table 2** including all positive and negative numbers that can be written into a memory with the size of 4 bits.

It is very easy to add up or subtract numbers that are written in two's complement form. As a matter of fact, if we are to subtract a positive integer, then we add the corresponding negative number, as shown in examples in **Table 3**. We will leave out the possible transfer of the number one to a higher order, which in our case exceeds the delimited 8 bits, because the given value cannot be entered in the computer memory of the size given.

Thus, we can see that the number one for negative numbers and a zero for positive numbers is always stored in the highest bit (first from the left). The range of numbers that can be entered is then given by the number of *N*-bits that we have at our disposable for this notation in the computer memory. Generally, the given range can be expressed from -2^{n-1} to 2^{n-1}-1. **Table 4** provides the usable ranges of decimal positive and negative integers subject to the size of the computer memory that we delimit for the given notation. At the same time, the types of variables used by JAVA or Visual Basic are also provided.

We will now enter the above mentioned program using JAVA, while again declaring a variable of the Byte type. But in this case, the variable will be somewhere in the range of $\langle -128;\ 127\rangle$. An example of an output of the program is provided in **Figures 3** to **5** declaring variable types of Byte, Short and Integer, including a corresponding numerical range of entered values.

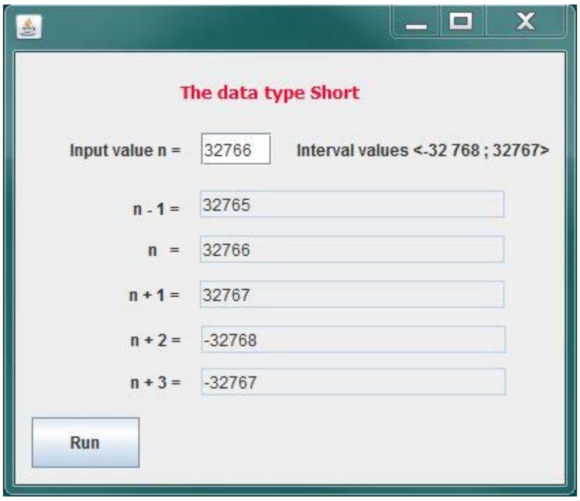


The data type Byte

Input value n = 127 Interval values <-128;127>

```
n - 1 = 126
n   =   127
n + 1 = -128
n + 2 = -127
n + 3 = -126
```

Figure 3. An example of an overflow when working with a Byte type variable

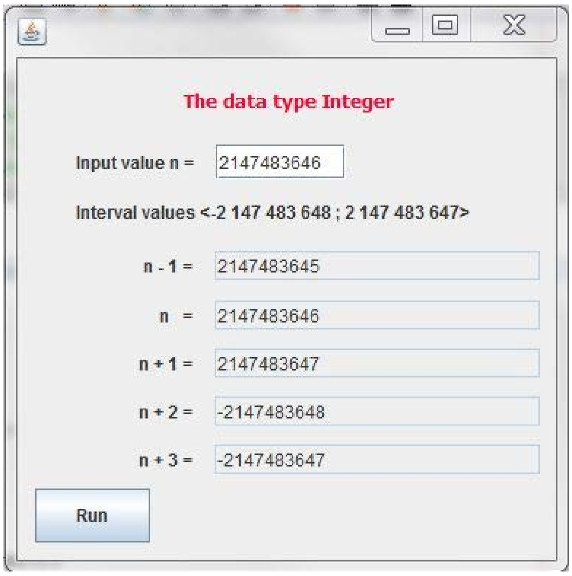


The data type Short

Input value n = 32766 Interval values <-32 768 ; 32767>

```
n - 1 = 32765
n   =   32766
n + 1 = 32767
n + 2 = -32768
n + 3 = -32767
```

Figure 4. An example of a memory overflow when working with a Short type variable



The data type Integer

Input value n = 2147483646

Interval values <-2 147 483 648 ; 2 147 483 647>

```
n - 1 = 2147483645
n   =   2147483646
n + 1 = 2147483647
n + 2 = -2147483648
n + 3 = -2147483647
```

Figure 5. An example of a memory overflow when working with an Integer type variable

Table 5. Storing real numbers that can be entered in reserved computer memories varying in size

Memory range	Number range	Number of valid digits	Type of variable	
			JAVA	Visual Basic
4 byte = 32 bit	$\langle \pm 1{,}4.10^{-45}; \pm 3{,}4.10^{38} \rangle$	7 – 8	Float	Single
8 byte = 64 bit	$\langle \pm 4{,}9.10^{-324}; \pm 1{,}7.10^{308} \rangle$	15 – 16	Double	Double
16 byte = 128 bit	$\langle \pm 7{,}9228.10^{-28}; \pm 7{,}9228.10^{28} \rangle$	28		Decimal

If the computer programmer is unaware of this, he may create a program using JAVA that will not produce the desired result. The program may also crash while running or it may end up in a so-called infinite loop. For example, if it creates a loop in which the control variable will be declared a Byte type, whereby it should reach a maximum value of, for example, 255 in the loop. Should this be the case, the loop will run until it reaches 127 and then it will again continue from -128 to 127, but it will never reach the end value of 255. And so the loop will never end; refer to the following source code that pertains to a simple method applied under JAVA:

```
public void overflow()
{
byte i;                       // declaration
for(i=1;i<=255;i++)           // infinite loop
System.out.println("i= "+i);  // loop body
}
```

By analogy, issues of the same type also occur in other programming languages. Similar problems arise in connection with whatever declaration of integer variables. In such cases, translators do not report any syntactic errors, but the program cannot be used and detecting the errors can be particularly difficult.

Storing real numbers in computer memories

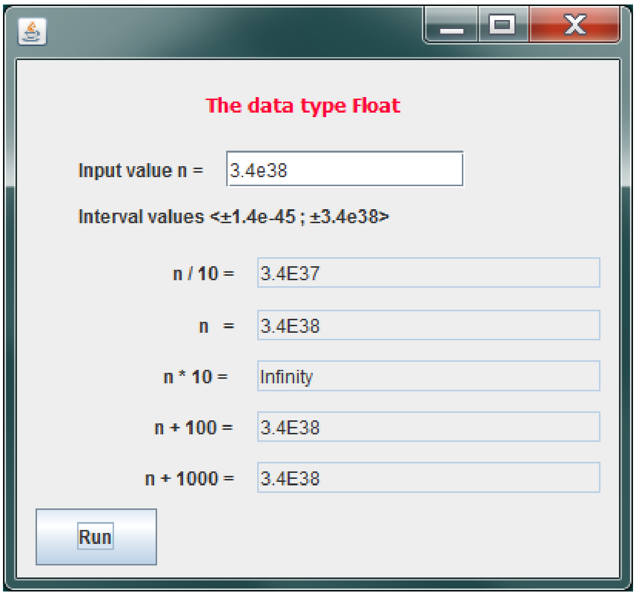
When making calculations using calculators or a computer, it is not possible to restrict oneself only to integers. Real numbers with a decimal part which are stored in exponential form also need to be entered in the computer memory. For example, the decimal number 42 531.86 can, in exponential form, be expressed as $4.253186.10^4$, where 4.253186 is the mantissa and 4 is the exponent of the written number. Thus, the matter concerns two separate numbers that are viewed as being positive or negative integers. Therefore, the notation of real numbers with a decimal part will then entirely depend on the range of the delimited computer memory. The size of the memory delimited for the notation of the mantissa will determine the accuracy of the representation of the real number, and thereby the number of valid digits. The size of the memory reserved for the notation of the exponent will determine the order range of the written numbers.

Table 5 provides memory ranges reserved for the storage of numbers, and the corresponding ranges of decimal numbers that can be entered into the reserved memory parts. Also provided are the corresponding types of variables that can be used in creating programs.

In order to verify the data that are provided in **Table 5**, a simple application can be created in JAVA that will document possible problems in programs that may arise when working with real numbers.

Figure 6 shows the test result when inputting the limit values of the Float variable type in JAVA. The error message *Infinity* appeared when multiplying by ten. When trying to add significantly lower-order values, the resulting value of the variable did not change.

When trying to input an above limit value of the variable, the error message *Infinity* appeared in all cases. Refer to **Figure 7**.



The data type Float

Input value n = 3.4e38

Interval values <±1.4e-45 ; ±3.4e38>

n / 10 = 3.4E37

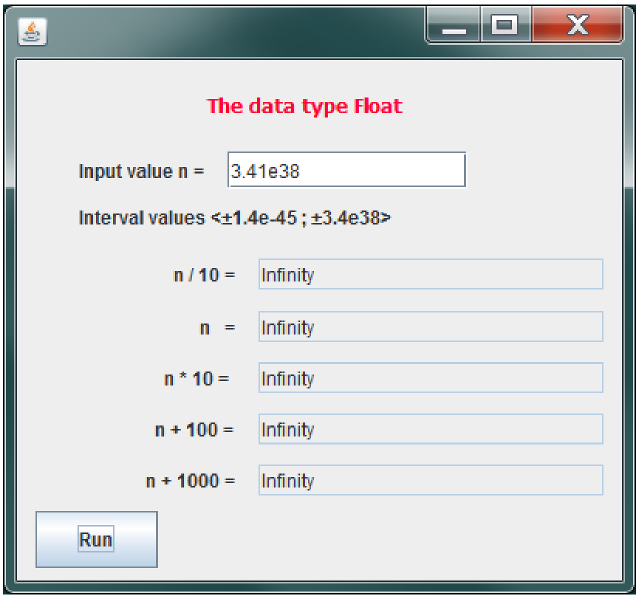
n = 3.4E38

n * 10 = Infinity

n + 100 = 3.4E38

n + 1000 = 3.4E38

Figure 6. Inputting the largest number of the Float type



The data type Float

Input value n = 3.41e38

Interval values <±1.4e-45 ; ±3.4e38>

n / 10 = Infinity

n = Infinity

n * 10 = Infinity

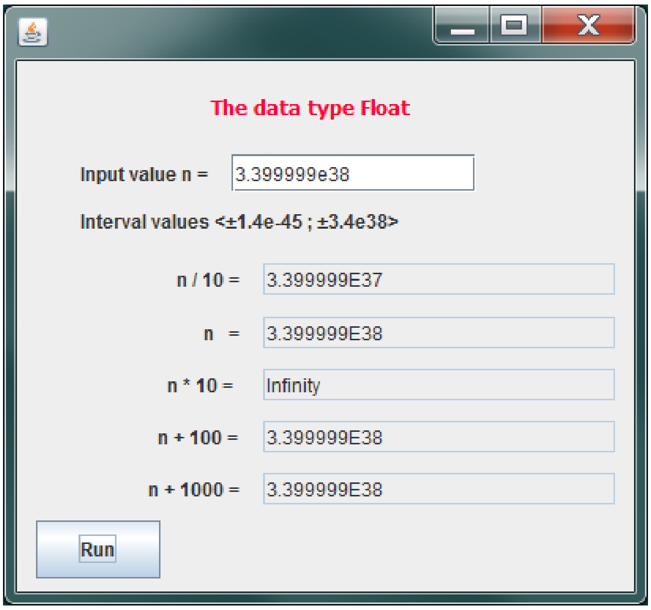
n + 100 = Infinity

n + 1000 = Infinity

Figure 7. Inputting an above limit number of the Float type

The Float data type (refer to Table 5) uses 32 bits, where a sign takes up 1 bit (1 represents negative numbers, 0 represents positive numbers), an exponent 8 bits and a mantissa 23 bits. The Double type uses 64 bits, of which 1 bit for the sign, 11 bits for the exponent and 52 bits for the mantissa. In both cases, the number format follows the IEEE 754 standard. Neither the Float nor the Double type are precise because they contain a fixed number of spaces for the mantissa and the exponent. The input value is entered from the keyboard in the form of a text string that is converted into the Float data type using the standard method. Thus, on conversion, a numerical value of the highest possible accuracy is stored. Refer to Figure 8 and Figure 9.

Where real numbers are concerned, not respecting the number of valid digits may be the source of errors. In what concerns the Float variable type, only seven valid digits can be used. If we enter an input chain of signs with a greater number of mantissa digits, the mantissa will be automatically reduced in accordance with the IEEE 754 standard. Refer to Figure 9. This needs to be kept in mind when, for example, adding up a larger number of lower-order numbers than is the final sum. It is obvious looking at Figures 8 and 9 that the result of the sum of the entered number n and values of a different order, specifically 100 or 1000, is not correct. The above described problems apply by analogy to the Double variable type. The number of valid digits is, however, double and the values of the exponent are of a higher order.



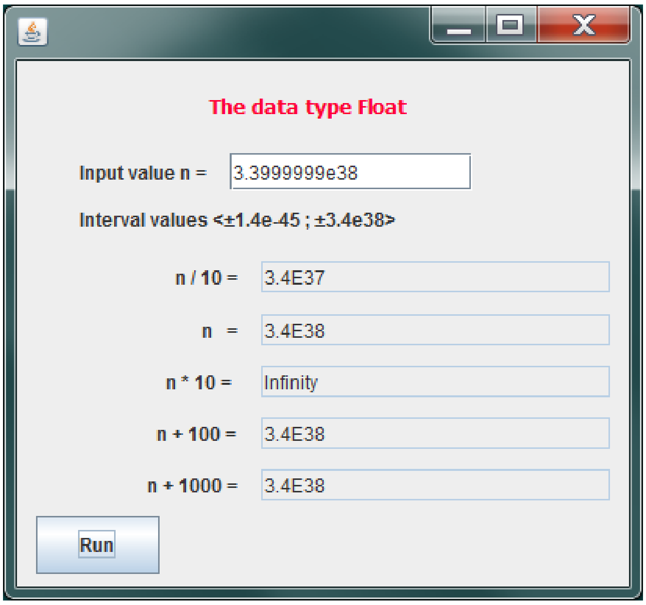
The data type Float

Input value n = 3.399999e38

Interval values <±1.4e-45 ; ±3.4e38>

```
     n / 10 =  3.399999E37
     n   =     3.399999E38
     n * 10 =  Infinity
    n + 100 =  3.399999E38
   n + 1000 =  3.399999E38
```

Figure 8. Inputting a number with seven valid digits (mantissa) for the Float data type



The data type Float

Input value n = 3.3999999e38

Interval values <±1.4e-45 ; ±3.4e38>

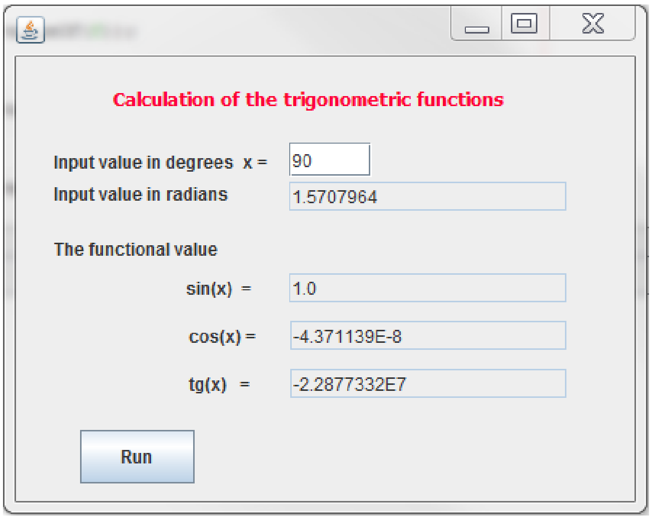
```
     n / 10 =  3.4E37
     n   =     3.4E38
     n * 10 =  Infinity
    n + 100 =  3.4E38
   n + 1000 =  3.4E38
```

Figure 9. Inputting a number with eight valid digits (mantissa) for the Float data type

THE INFLUENCE OF THE SIZE OF A RESERVED COMPUTER MEMORY ON THE ACCURACY OF A CALCULATION IN CREATING PROGRAMS

Let's now go back to the problems associated with calculating the values of trigonometric functions that were referred to at the beginning of this paper. In the given program, the variable *x* was declared as being a Double variable type and the memory was reserved to 8 bytes. Refer to **Figure 1**. Should we use a Float type variable, meaning that we will reserve only 4 bytes in the computer memory for storing a numerical value, the accuracy of the calculation will only worsen. Refer to **Figure 10**.

Apart from declaring variables, the accuracy of the notation of the number π, which is used for the conversion of angular degrees to radians, for example, also significantly influences the accuracy of the calculation. **Figure 11** shows the result of the same calculation as provided under **Figure 1**, but the number π is entered with a precision of only 5 decimal places. The Double type variable was used in both cases. It is obvious that we would make even bigger mistakes if we were to enter the value π = 3.14.



Calculation of the trigonometric functions

Input value in degrees x = 90

Input value in radians 1.5707964

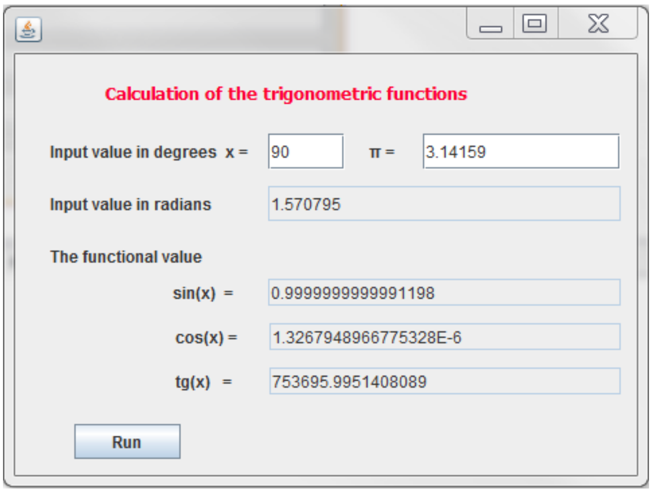
The functional value

sin(x) = 1.0

cos(x) = -4.371139E-8

tg(x) = -2.2877332E7

Figure 10. Calculation of the value of trigonometric functions in declaring a Float type variable



Calculation of the trigonometric functions

Input value in degrees x = 90 π = 3.14159

Input value in radians 1.570795

The functional value

sin(x) = 0.9999999999991198

cos(x) = 1.3267948966775328E-6

tg(x) = 753695.9951408089

Figure 11. Calculation of the value of trigonometric functions when entering the number π with five decimal places

CONCLUSION

As part of the research conducted, we collected several simple calculations and processed them using computer technology and the results proved to be incorrect. At the same time we also demonstrated that these errors can easily be explained if we have a basic knowledge of mathematics. It is a case of a natural and illustrative combination of teaching mathematics and ICT at primary and secondary schools.

If pupils come to understand how numbers are stored in a computer memory, they will be aware of the fact that not only did they learn about the binary system in their math's class because they had to, but also because it is a very important thing to know with a practical impact on basic and simple calculations made using calculators or personal computers, in MS Excel, for example. Pupils presented with specific arithmetical problems that they will come across when working on computers will realize the necessity of understanding the basics of mathematics. They will also become aware of the fact that they cannot impetuously believe all the results provided to them by computer technology. Every result needs to be thought through and correctly interpreted. Thus, an interdisciplinary approach to the teaching of ICT in correlation with mathematics should contribute to primary school pupils and secondary school students improving their command of mathematics.

REFERENCES

Alghamdi, A. K. H. (2017). The Effects of an Integrated Curriculum on Student Achievement in Saudi Arabia. *EURASIA Journal of Mathematics, Science and Technology Education, 13*(9), 6079-6100. https://doi.org/10.12973/eurasia.2017.01051a

Ardies, J., de Mayer, S., Gijbels, D., & van Keulen, H. (2015). Students attitudes towards technology. *International Journal of Technology and Design Education*, *25*(1), 43-65. https://doi.org/10.1007/s10798-014-9268-x

Brunton, M., & Coll, R. K. (2005). Enhancing Technology Education by Forming Links with Industry: A New Zealand Case Study. *International Journal of Science and Mathematics Education*, *3*(1), 141-166. https://doi.org/10.1007/s10763-004-1516-1

Carreras-Marin, A., Blasco Martel, Y., Badia-Miro, M., Bosch-Princep, M., Morillo-Lopez, I., Cairo-I-Cespedes, G., & Casares Vidal, M. D. (2013). The promotion and assessment of generic skills from interdisciplinary teaching teams. *Edulearn13: 5th International Conference on Education and new Learning Technologies* (pp. 201-207). IATED-INT Assoc Technology Education & Development, Lauri Volpi 6, Valencia, Burjassot 46100, Spain.

Chai, S., & Chun, S. (2015). The Effects of STEM-based Programming Education with Robot on Creativity and Character of Elementary School Students. *Journal of the Korean Association of Information Education*, *19*(2), 159-166. https://doi.org/10.14352/jkaie.2015.19.2.159

Coriell, J., & Corbett, K. (2013). An Innovative Approach to Secondary Mathematics for Engineering and Science. *2013 IEEE Frontiers in Education Cconference*. IEEE, 345 E 47th St, NY 10017 USA. https://doi.org/10.1109/FIE.2013.6684961

Debebe, S. A., Bhatt, R., & McGoron, A. J. (2013). Web Based Interactive Medical Imaging Applications for Teaching Nuclear Medicine. *29th Southern Biomedical Engineering Conference (SBEC 2013)* (pp. 133-134). IEEE, 345 E 47th St, NY 10017 USA. https://doi.org/10.1109/SBEC.2013.75

Doorman, M., Drijvers, P., Gravemeijer, K., Boon, P., & Reed, H. (2012). Tool use and the development of the function concept: from repeated calculations to functional thinking. *International Journal of Science and Mathematics Education*, *10*(6), 1243-1267. https://doi.org/10.1007/s10763-012-9329-0

Eilks, I. & Markic, S., (2014). Editorial: Traditions and Trends in Mathematics, Science and Technology Education in Germany. *EURASIA Journal of Mathematics, Science and Technology Education*, *10*(4), 229-230. https://doi.org/10.12973/eurasia.2014.1093a

Foster, C. (2015). Exploiting unexpected situations in the mathematics classroom. *International Journal of Science and Mathematics Education*, *13*(5), 1065-1088. https://doi.org/10.1007/s10763-014-9515-3

Herout, P. (2010). *Učebnice jazyka Java*. [Java textbook]. České Budějovice, Czech Republic: Kopp.

Jehlička, V. (2010). Interdisciplinary relations in teaching of programming (Conference Paper). In: *Applied Computing Conference - Proceedings*. Timisoara, Romania, 2010, pp 33-38.

Jehlička, V. (2013). System approach to the teaching of algorithm development. In: *Proceedings of the 10th International Conference Efficiency and Responsibility in Education 2013*, Prag, Czech Republic: Powerprint, s. r. o., pp 248-255.

Juránek, R., Machalík, S., & Zemčík, P. (2011). Research on image features for classification of wear debris. *Machine Graphics and Vision*, *20*(4), 479-493.

Klisch, Y., Miller, L. M., Wang, S., & Epstein, J. (2012). The Impact of a Science Education Game on Students' Learning and Perception of Inhalants as Body Pollutants. *Journal of Science Education and Technology*, *21*(2), 295-303. https://doi.org/10.1007/s10956-011-9319-y

Konysheva, A. V., & Ibragimova, E. N., (2017). Training of Engineers in Mathematics at University on the Basis of the Information Cybernetic Approach. *EURASIA Journal of Mathematics, Science and Technology Education*, *13*(8), 4379-4391. https://doi.org/10.12973/eurasia.2017.00933a

Kulička, J. (2014). Estimation of uncertainty for problem solving by fuzzy mathematics tools. *Frontiers in Mathematics and Science Education Research* (pp. 144-152). Science Education Research Group at Eastern Mediterranean University, Famagusta, North Cyprus.

Kulička, J., & Berková, A. (2015). The estimation of achieved knowledge of students using the fuzzy mathematics tools. *ICERI 2015: 8th International Conference of Education, Research and Innovation* (pp. 7813-7818). IATED-INT Assoc Technology Education & Development, Lauri Volpi 6, Valencia, Burjassot 46100, Spain.

Kuna, P., & Turčáni, M. (2012). Applying the principles of e-course methodology creation for the subject Discrete Mathematics in the information learning environment. *2012 15th International Conference on Interactive Collaborative Learning (ICL)*. IEEE, 345 E 47th St, NY 10017 USA.

Lorenzo, M. (2005). The Development, Implementation, and Evaluation of a Problem Solving Heuristic. *International Journal of Science and Mathematics Education*, *3*(1), 33-58. https://doi.org/10.1007/s10763-004-8359-7

Martin, S. N., Im, S., & Song, J., (2016). Editorial: Past, Present, and Future of Science, Mathematics, Engineering, and Technology Education Research and Practice in South Korea. *EURASIA Journal of Mathematics, Science, 12*(7), 1707-1710. https://doi.org/10.12973/eurasia.2016.1529a

Matsumoto, P. (2014). Exploring Interactive and Dynamic Simulations Using a Computer Algebra System in an Advanced Placement Chemistry Course. *Journal of Chemical Education*, *91*(9), 1326-1333. https://doi.org/10.1021/ed4008233

Meng, C. C., Idris, N. & Eu, L. K., (2014). Secondary Students' Perceptions of Assessments in Science, Technology, Engineering, and Mathematics (STEM). *EURASIA Journal of Mathematics, Science and Technology Education*, *10*(3), 219-227. https://doi.org/10.12973/eurasia.2014.1070a

Merino, G. A., Anino, M. M., & Ravera, E. P. (2013). Process Control as a Source of Integrator Problems for a Differential Equations Course in Bioengineering. *IEEE Latin America Transactions*, *11*(1), 640-645. https://doi.org/10.1109/TLA.2013.6502877

Moratalla de la Hoz, A., & Sanz Garcia, M. A. (2010). Rhythm: a motivation to study geometry. *Edulearn10: International Conference on Education and New Learning Technologies* (pp. 790-798). IATED-INT Assoc Technology Education & Development, Lauri Volpi 6, Valencia, Burjassot 46100, Spain.

Němec, R., Berková, A., & Hubálovský, Š. (2016). The use of method of multidisciplinary approach, modeling and simulation in learning of gear ratios. *International Journal of Mathematics and Computers in Simulation*, *10*, 46-52.

OECD (2016). OECD: Programme for International Student Assessment (PISA). Retrieved from https://www.oecd.org/pisa/

Oliveira, C. F., & Barufaldi, J. P. (2009). Aliens are us. An innovative course in astrobiology. *International Journal of Astrobiology*, *8*(1), 51-61. https://doi.org/10.1017/S1473550408004370

Potvin, P., & Hasni, A. (2014). Analysis of the Decline in Interest towards School Science and Technology from Grades 5 through 11. *Journal of Science Education and Technology*, *23*(6), 784-802. https://doi.org/10.1007/s10956-014-9512-x

Rey, L., Dios, C., Sandia, B. & Martinez-Azua, C. (2014). Innovation teaching for integrating interdisciplinary knowledge in the University of Extremadura. *ICERI 2014: 7th International conference of education, research and innovation* (pp. 4951-4958). IATED-INT Assoc Technology Education & Development, Lauri Volpi 6, Valencia, Burjassot 46100, Spain.

Rosca, S., & Todasca, C. (2014). Chemical Education - A Key Factor in Facing the Challenges of the Future. *Vision 2025: How to succeed in the global chemistry enterprise*, *1157*, 77-90.

Sellami, A., El.Kassem, R. C., Al-Qassass, H. B., & Al-Rakeb, N. A. (2017). A Path Analysis of Student Interest in STEM, with Specific Reference to Qatari Students. *EURASIA Journal of Mathematics, Science and Technology Education*, *13*(9), 6045-6067. https://doi.org/10.12973/eurasia.2017.00999a

Soria-Olivas, E., Martín-Guerrero, J. D., Martínez-Sober, M., & Ayuso-Moya, A. (2010). Description and evaluation of an introductory course to Matlab for a heterogeneous group of university students. *Computer Applications in Engineering Education*, *18*(4), 750-756. https://doi.org/10.1002/cae.20282

Tinnirello, A. M., Gago, E. A., & Dadamo, M. M. (2012). Project learning environments in mechanical engineering education. *ICERI 2012: 5th International Conference of Education, Research and Innovation* (pp. 2081-2087). IATED-INT Assoc Technology Education & Development, Lauri Volpi 6, Valencia, Burjassot 46100, Spain.

Tinnirello, A., Gago, E., & Dadamo, M. (2013). Interdisciplinary activities to improve the learning methodology performed in mechanical engineering degree studies. *EDULEARN13: 5th International Conference on Education and new Learning Technologies* (pp. 5407-5416). IATED-INT Assoc Technology Education & Development, Lauri Volpi 6, Valencia, Burjassot 46100, Spain.

Turpin, M., Matthee, M., & Kruger, A. (2015). The Teaching of Creativity in Information Systems Programmes at South African Higher Education Institutions. *African Journal of Research in Mathematics, Science and Technology Education*, *19*(3), 278-288. https://doi.org/10.1080/10288457.2015.1104838

Van Borkulo, S. P., van Joolingen, W. R., Savelsbergh, E. R. & de Jong, T. (2012). What Can Be Learned from Computer Modeling? Comparing Expository and Modeling Approaches to Teaching Dynamic Systems Behavior. *Journal of Science Education and Technology*, *21*(2), 267-275. https://doi.org/10.1007/s10956-011-9314-3

Zhu, Y., & Leung, F. K. S. (2011). Motivation and achievement: is there an East Asian model? *International Journal of Science and Mathematics Education*, *9*(5), 1189-1212. https://doi.org/10.1007/s10763-010-9255-y

http://www.ejmste.com