Educational Software for Teaching the Basics of Computer Science

MANUELA PANOIU, ANCA IORDAN, CAIUS PANOIU, LOREDANA GHIORGHIONI
Electrical Engineering and Industrial Informatics Department
Polytechnic University of Timisoara
Engineering Faculty of Hunedoara
Revolutiei street, no 5 Hunedoara, cod 331128
ROMANIA
{m.panoiu, anca.iordan, c.panoiu }@fih.upt.ro anaderol_hd@yahoo.com

Abstract: - In this paper we describe the visual environment for learning the basics of computer science: the binary, octal and hexadecimal numeration systems and the rules of storing the information in the computers memory. The software package is an interactive programming environment with graphical user interface that has user friendly interface and logic of usage. It offers all of the essential functions needed to understand the basics of binary, octal and hexadecimal numeration systems, the conversion between these systems and arithmetical operation with numbers in these numeration systems. The software offers also options needed to understand the two's complement code, the floating point numbers single and double precision. Another option of the software is for on-line testing of the students. This option can be use both for evaluation and self-evaluation. The software package is implemented in Java language.

Key-Words: - Java, numeration system, educational software, Interactive Tutorials, virtual laboratory

1 Introduction

Today, the computers are used in many areas, including in education, from the elementary school to the universities. The purpose of using computers in education can be realized by using new didactical teaching methods based on educational software that is programs useful in teaching-learning process [1] [2] and [3].

The CBL type systems (Computer Based Learning) are used in universities from the entire world. For instance, at Dundee University, Scotland, UK [2] a CBL system is integrated part in the courses from the first two years in computers field. These courses present the students the object-oriented programming, data structures and algorithms.

The advantages of using computers in education are obviously: the students have the possibility to interact with virtual environment, this method being preferred in comparison with lecturing of a classic material. By using the computer in didactical activity the learning productivity is increased: the necessary information is faster and accurate obtained, process and sends, by eliminate unnecessary time delay. It can be showing to the student some information which is inaccessible otherwise: dynamic diagrams, moving images, sounds.

Teaching computers to novices has proven to be a challenge for both teachers and students. Many students find the computers programming module difficult and disheartening and this could have an impact on their attitude to software development throughout the course and as a career choice. For staff involved in teaching computers programming it can also be very disheartening when student apparently fail to understand and be able to use even the basic data structures. This happen because the students can't understand how the information is stored in memory. They cannot "think" like a computer, in binary system and cannot make a connection between binary system and the voltage levels (0 and 5 V) used in computers hardware devices. For students who are beginners in computers science it is difficult to think in binary or hexadecimal system. These difficulties have prompted researchers to investigate tools and approaches that may ease the difficulty of teaching and learning data in computer memory [4].

In this paper is described an educational software useful for a better understanding of binary and hexadecimal numeration system, complementary code for storing integers in memory, the floating point used for real numbers and the rules of storing characters in memory. The software allows also an option for understanding the arithmetical operation with numbers in binary, octal and hexadecimal system.

2 The informatics system design and implementation

2.1 System analysis

The informatics system will be described in a clear and concise manner and will be presented the use-cases, using UML [5]. In fig. 1 is shown the use-case diagram. Each use case is a sequence of related transactions performed by an actor and the system in a dialogue.

2.2. System design

The conceptual modelling allows the identification of the most important concepts for informatics system [6]. The sequence diagrams [7] for this software was made with Argo-UML 0.24. The sequence diagrams are used primarily to show the interactions between objects in the sequential order that those interactions occur. In fig 2 are illustrated two sequence diagrams for the software. The class diagram is show in fig. 3.

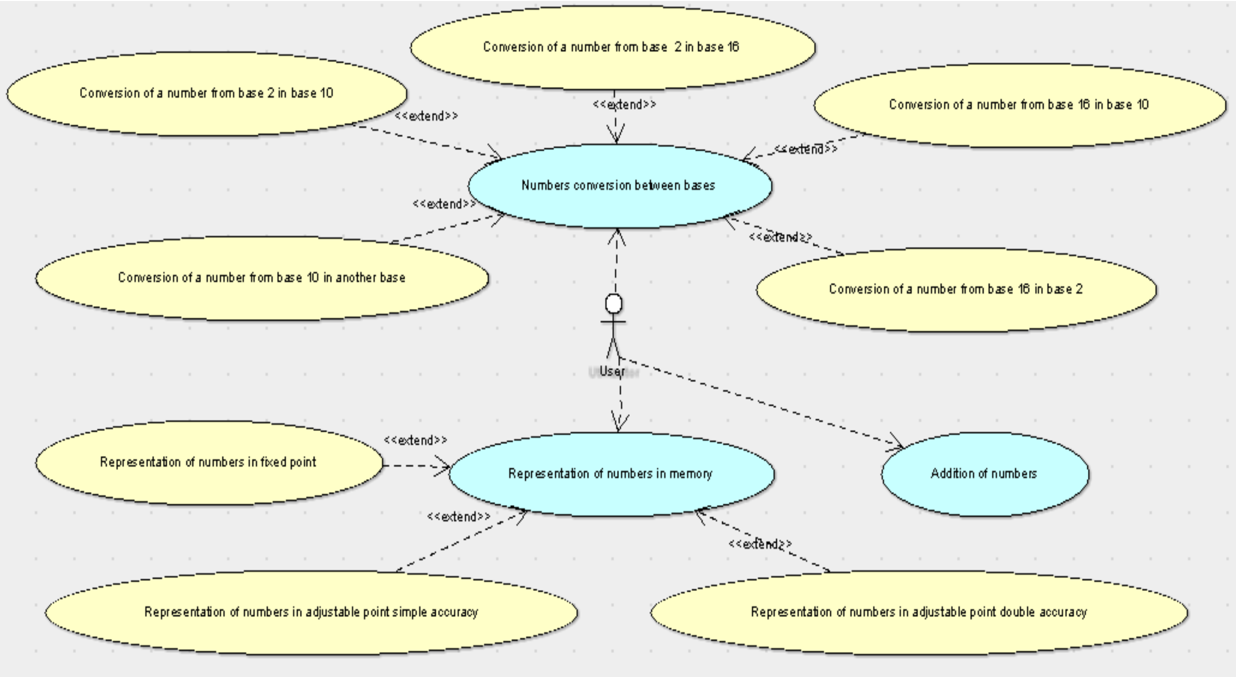

Conversion of a number from base 2 in base 16
Conversion of a number from base 2 in base 10
Conversion of a number from base 16 in base 10
<<extend>>
Numbers conversion between bases
Conversion of a number from base 10 in another base
Conversion of a number from base 16 in base 2
User
Representation of numbers in fixed point
Representation of numbers in memory
Addition of numbers
Representation of numbers in adjustable point simple accuracy
Representation of numbers in adjustable point double accuracy

Fig. 1 The use-case diagram

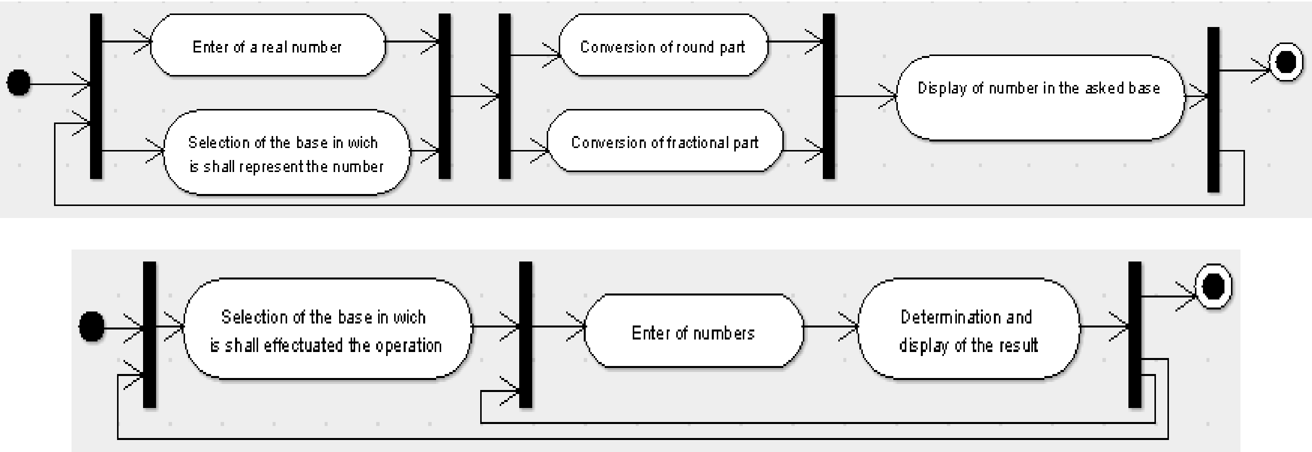

Enter of a real number
Selection of the base in wich is shall represent the number
Conversion of round part
Conversion of fractional part
Display of number in the asked base
Selection of the base in wich is shall effectuated the operation
Enter of numbers
Determination and display of the result

Fig. 2. The sequence diagrams

2.3. The component diagram

The component diagram allow to view the system is divided and the dependencies between the modules. The components diagram focus on the physical software elements. The diagram is show in fig. 4 and describes the collection of the main components.

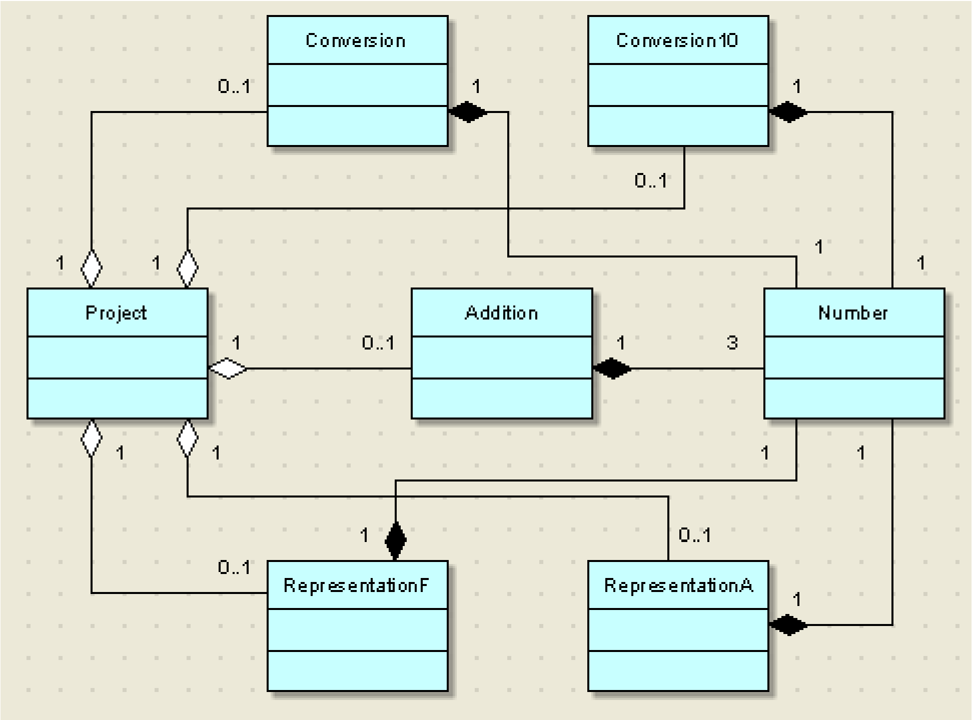


Fig. 3 The components diagram

It was implements also a lot of classes of which are presented two classes: IntNumber and FloatNumber. These classes are derived from the class Number which is one of the main classes of the software package.

```
public class IntNumber {
private int n;
private String s;
public IntNumber()
{ n=0;
  s="";
}
public IntNumber(int nn)
{ n=nn;
  s=""; }
public IntNumber(String s)
{ n=Integer.parseInt(s);
  s="";
}
int getNumber()
{ return n;
}
void setNumber(int nn)
{ n=nn;
 }
String getNumberS()
{ return s;
}
public void conv(int base, int n)
 {  if (n>0)
      { conv(base,n/base);
        s=s+String.valueOf(n%base);
    } else s="";
}
public RealNumber(String s)
{n=Float.parseFloat(s);
s="";sf="";
}
float getNumber()
{return n;
}
void setNumber(float nn)
{n=nn;
}
String getNumberS()
{ return s;
}
void convi(int base, int n)
 { if (n>0)
      { convi(base,n/base);
        si=si+String.valueOf(n%base);
    } else si=""; }
void convf(int base, float n)
{  if (n!=0.0&&sf.length()<24)
```

```
   {float p=n*base;
sf=sf+String.valueOf(p).substring(0,1)
;    System.out.println(p);
    if (p>=1)n=p-base+1;
       else n=p;
    System.out.println(n);
    convf(base,n);
   }     }
public void conv(int base)
{  int x;float y;
   String temp=String.valueOf(n);
   int poz=temp.indexOf('.');
   String sint=temp.substring(0, poz);
   String
sfrac="0."+temp.substring(poz+1,
      temp.length());
    x=Integer.valueOf(sint);
    y=Float.valueOf(sfrac);
    System.out.println(x+"  "+y);
    convi(base,x);
    convf(base,y);
    s=si+sf;
}
```

The class diagram for the main classes of the application is show in fig. 4

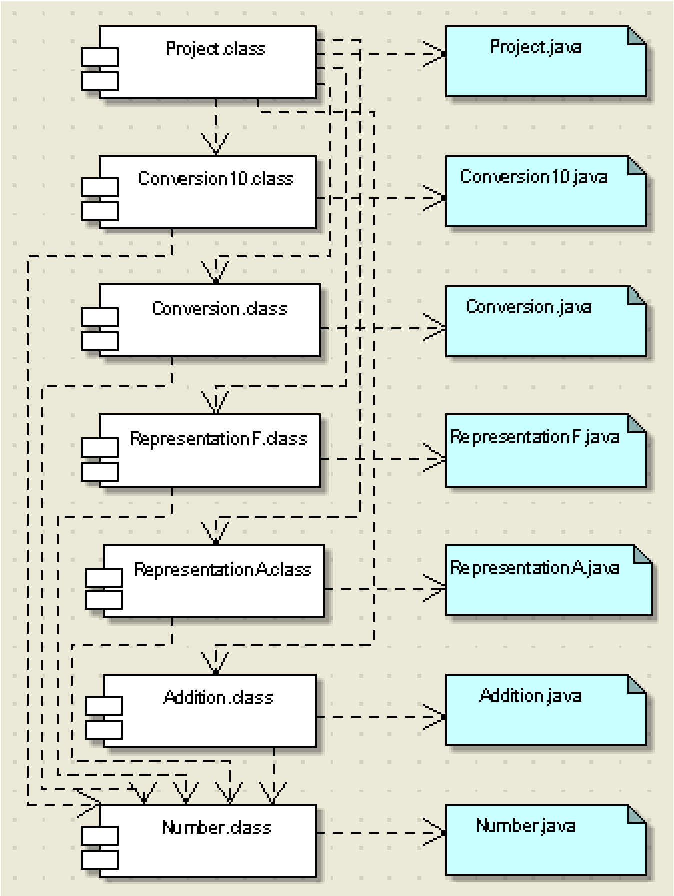


Fig. 4. The class diagram

3. Description of the interactive tutorial

The application was implemented in Java using NetBeans platform. The graphical user interface for this application is presented in figure 5. Using main menu for the application you could select the main options that are available.

From the menu „Conversion between numeration systems" it could be selected one of the chosen conversion. The first option „Conversion from decimal to binary, octal or hexadecimal system" allows visualization step by step for the mode that is used to convert a number from the decimal system to a binary one, octal or hexadecimal system these being actually the numeration systems used in computers science. The graphical user interface that indicates these steps is presented in figure 6. The student could input any number in decimal and than using the button „Run conversion" it could be seeing the steps needed to fulfil the conversion. To put into effect the animation it was used class Thread.

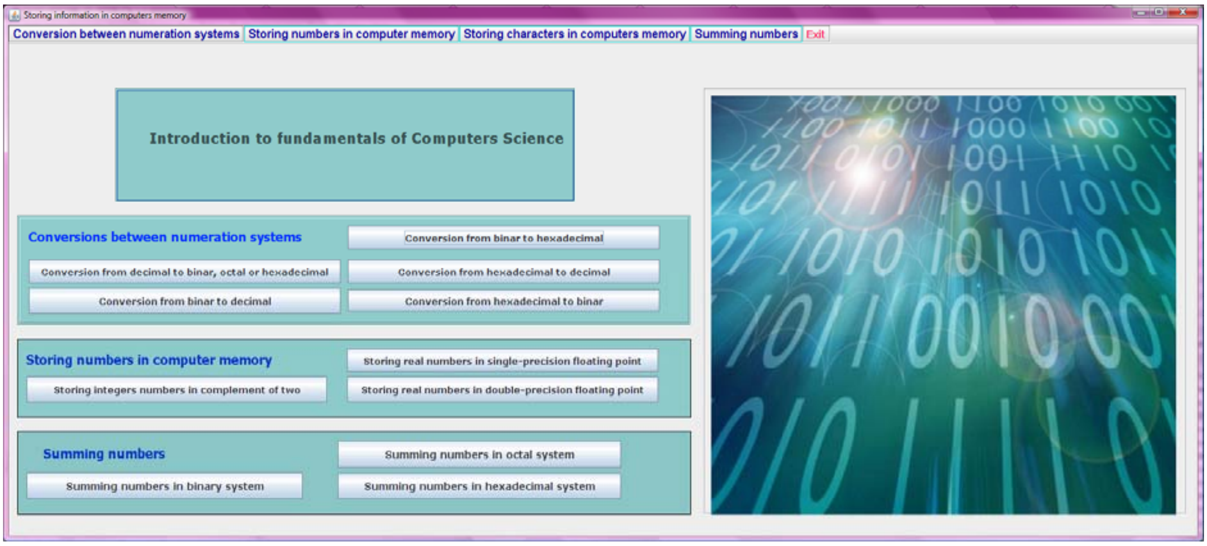
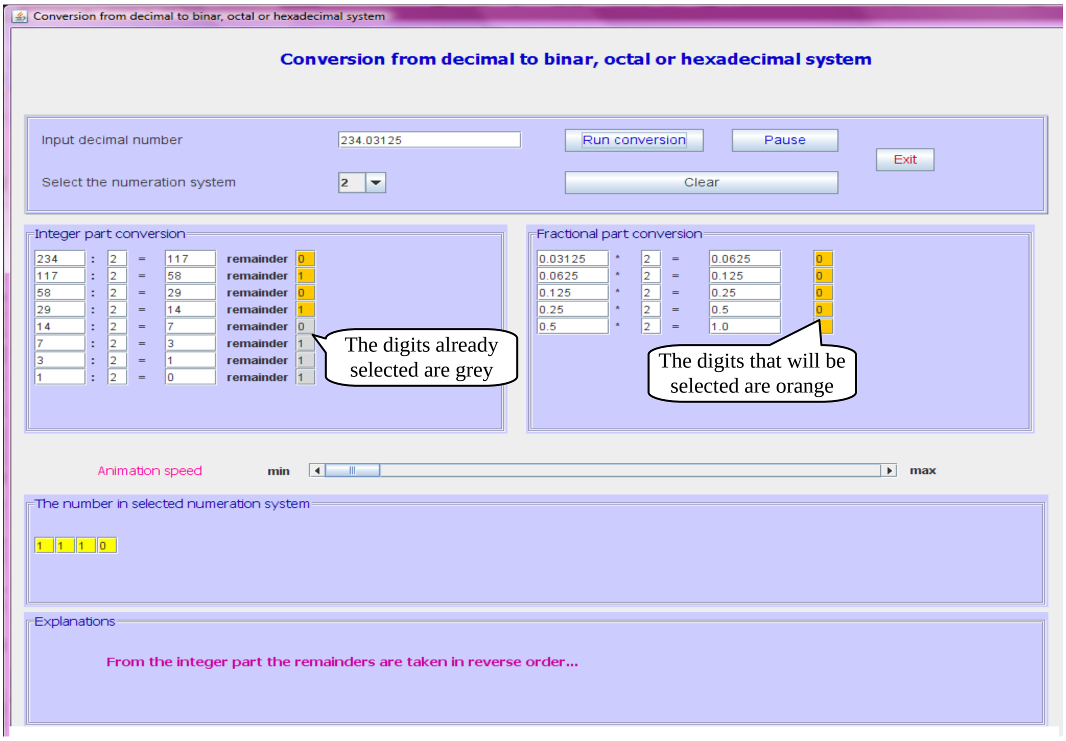

Storing information in computers memory
Conversion between numeration systems | Storing numbers in computer memory | Storing characters in computers memory | Summing numbers | Exit
Introduction to fundamentals of Computers Science
Conversions between numeration systems
Conversion from binar to hexadecimal
Conversion from decimal to binar, octal or hexadecimal
Conversion from hexadecimal to decimal
Conversion from binar to decimal
Conversion from hexadecimal to binar
Storing numbers in computer memory
Storing real numbers in single-precision floating point
Storing integers numbers in complement of two
Storing real numbers in double-precision floating point
Summing numbers
Summing numbers in octal system
Summing numbers in binary system
Summing numbers in hexadecimal system

Fig. 5. The main Graphical User Interface

This algorithm for the conversion could be seen with the speed for the animation slow or faster by selecting one. Also the visualization could be stopped and than restarted. Every step is explained into a panel. In the figure 6 is introduced a number that could be represented in a binary on a finite number of fractional digits. After obtaining the each digit the application stopped a number of milliseconds that depends of the animation speed selected. In this way the students could understand better the transformation process.

But not every number can be written with a finite number of digits in binary system. If the number obtains in the numeration system that you chose don’t have a finite number of digits at the fractional part it will be represent at the outside 24 from those. Such a situation is presented in fig. 7. In this case the students can see that the digits obtained at the fractional part are random and do not follow any rule.


Conversion from decimal to binar, octal or hexadecimal system
Run conversion
Pause
Exit
Clear
The digits already selected are grey
The digits that will be selected are orange
Animation speed
min
max

Conversion from decimal to binar, octal or hexadecimal system

Input decimal number: 234.03125

Select the numeration system: 2

Integer part conversion

234	:	2	=	117	remainder	0
117	:	2	=	58	remainder	1
58	:	2	=	29	remainder	0
29	:	2	=	14	remainder	1
14	:	2	=	7	remainder	0
7	:	2	=	3	remainder	1
3	:	2	=	1	remainder	1
1	:	2	=	0	remainder	1

Fractional part conversion

0.03125	*	2	=	0.0625	0
0.0625	*	2	=	0.125	0
0.125	*	2	=	0.25	0
0.25	*	2	=	0.5	0
0.5	*	2	=	1.0	

The number in selected numeration system

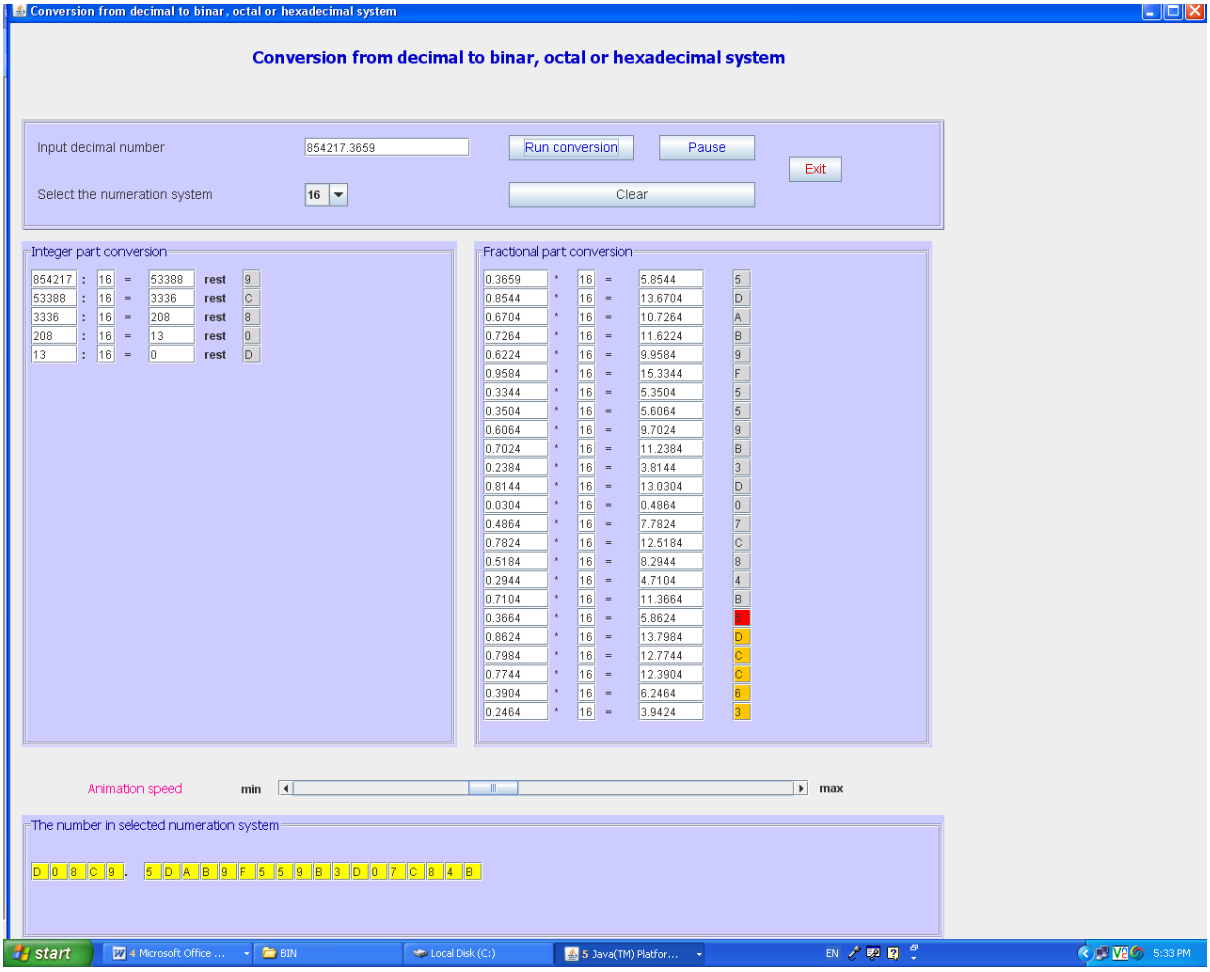
1 1 1 0

Explanations

From the integer part the remainders are taken in reverse order...

Fig. 6. The Graphical User Interface for conversion from decimal to binary system

There are cases where the digits obtained at the fractional part are periodical like in fig. 8. In the fig. 6 and 7 could see also how the number is being built by selecting the digits obtain at the integer part in an inverted order and by selecting the digits obtained at the fractional part in directed order. At the end of the transformation process into a panel could see the whole number transformed, like in fig. 6 and 7.



Conversion from decimal to binar, octal or hexadecimal system

Input decimal number 854217.3659

Select the numeration system 16

Integer part conversion

854217	:	16	=	53388	rest	9
53388	:	16	=	3336	rest	C
3336	:	16	=	208	rest	8
208	:	16	=	13	rest	0
13	:	16	=	0	rest	D

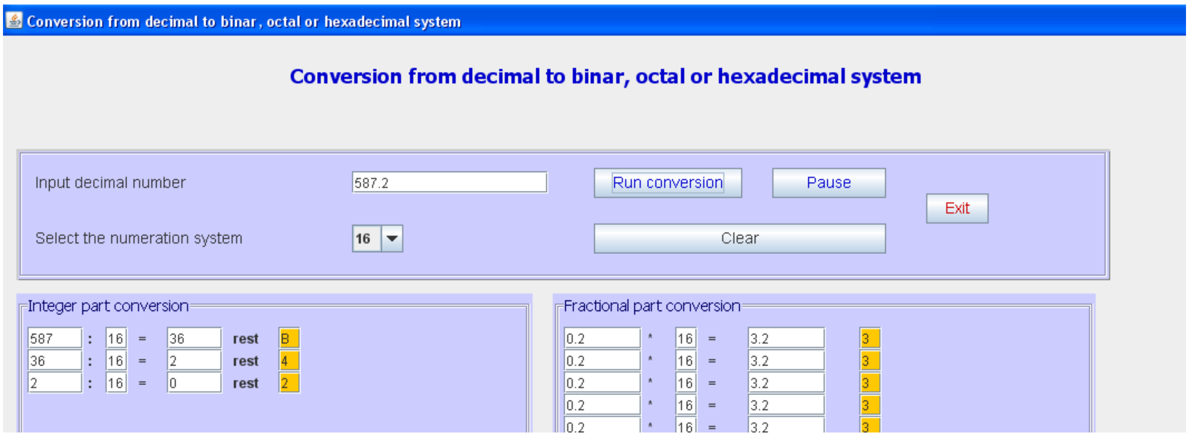
Fractional part conversion

0.3659	*	16	=	5.8544	5
0.8544	*	16	=	13.6704	D
0.6704	*	16	=	10.7264	A
0.7264	*	16	=	11.6224	B
0.6224	*	16	=	9.9584	9
0.9584	*	16	=	15.3344	F
0.3344	*	16	=	5.3504	5
0.3504	*	16	=	5.6064	5
0.6064	*	16	=	9.7024	9
0.7024	*	16	=	11.2384	B
0.2384	*	16	=	3.8144	3
0.8144	*	16	=	13.0304	D
0.0304	*	16	=	0.4864	0
0.4864	*	16	=	7.7824	7
0.7824	*	16	=	12.5184	C
0.5184	*	16	=	8.2944	8
0.2944	*	16	=	4.7104	4
0.7104	*	16	=	11.3664	B
0.3664	*	16	=	5.8624	5
0.8624	*	16	=	13.7984	D
0.7984	*	16	=	12.7744	C
0.7744	*	16	=	12.3904	C
0.3904	*	16	=	6.2464	6
0.2464	*	16	=	3.9424	3

The number in selected numeration system

D 0 8 C 9 . 5 D A B 9 F 5 5 9 B 3 D 0 7 C 8 4 B

Fig. 7. The GUI for conversion from decimal to binary system. The fractional digits obtained not follow any rule


Conversion from decimal to binar, octal or hexadecimal system
Run conversion
Pause
Exit
Clear

Conversion from decimal to binar, octal or hexadecimal system

Input decimal number 587.2

Select the numeration system 16

Integer part conversion

587	:	16	=	36	rest	B
36	:	16	=	2	rest	4
2	:	16	=	0	rest	2

Fractional part conversion

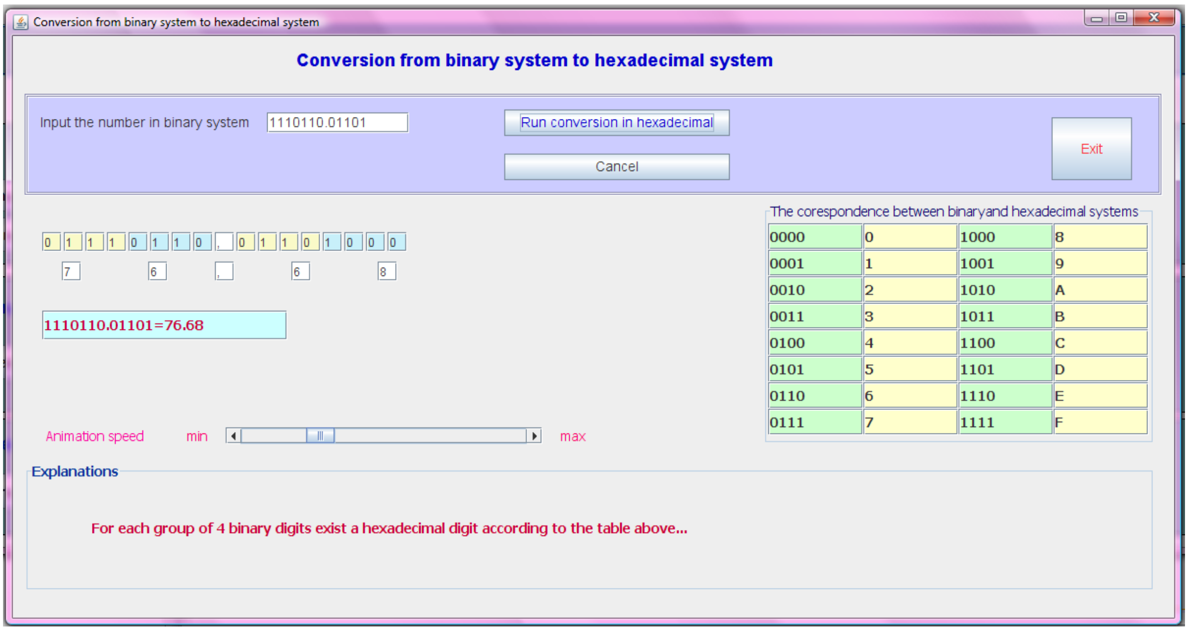
0.2	*	16	=	3.2	3
0.2	*	16	=	3.2	3
0.2	*	16	=	3.2	3
0.2	*	16	=	3.2	3
0.2	*	16	=	3.2	3

Fig. 8. The GUI for conversion from decimal to hexadecimal system. The fractional digits obtained are periodical

Another option from the main menu of the application demonstrates to the students how it could convert the number from the binary system in hexadecimal system or inverse. That graphical user interface is presented in figure 9. It could be observe that the students have the opportunity to see how that number could be converted directly from the binary base into a hexadecimal base without using the decimal base.

That is possible because exists the correspondence from the digits of the hexadecimals base and the one for the binary base showed into a panel. That correspondence could be used in the process of direct conversion.

The application gives also the possibility to convert a number from hexadecimal base into a binary base, the graphical user interface being similar whit that one presented into the figure 9.


Conversion from binary system to hexadecimal system
Run conversion in hexadecimal
Cancel
Exit
Animation speed
min
max

Conversion from binary system to hexadecimal system

Input the number in binary system 1110110.01101

0 1 1 1 0 1 1 0 , 0 1 1 0 1 0 0 0

7 6 , 6 8

1110110.01101=76.68

The corespondence between binaryand hexadecimal systems

0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

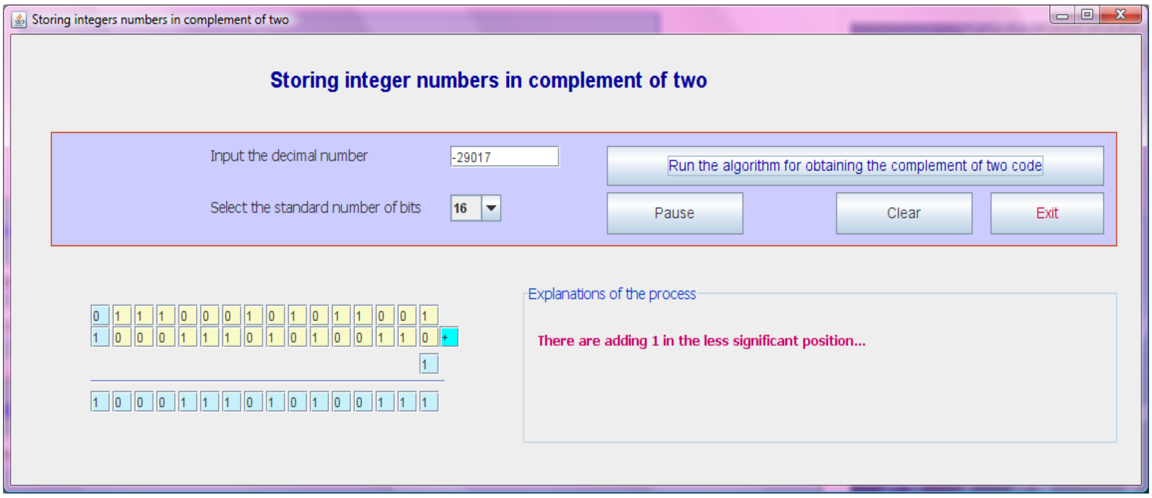
Explanations

For each group of 4 binary digits exist a hexadecimal digit according to the table above...

Fig. 9. The Graphical User Interface for conversion from binary to hexadecimal

Another option of the software, which could be selected from the main menu, is for the visualization of the rule for physical store of the numbers into memory. First option refers to the integer numbers stored in complementary code, and the next two refers to the store for the real numbers in floating point simple precision, respective double precision.

In the figure 10 it could be seen the rules for physical store in the memory for the integer numbers.


Storing integers numbers in complement of two
Run the algorithm for obtaining the complement of two code
Pause
Clear
Exit

Storing integer numbers in complement of two

Input the decimal number -29017

Select the standard number of bits 16

```
0 1 1 1 0 0 0 1 0 1 0 1 1 0 0 1
1 0 0 0 1 1 1 0 1 0 1 0 0 1 1 0 +
                              1
---------------------------------
1 0 0 0 1 1 1 0 1 0 1 0 0 1 1 1
```

Explanations of the process

There are adding 1 in the less significant position...

Fig. 10. The Graphical User Interface for showing the complement of two code for integers

The students can input any number between $[-2^{31},2^{31}-1]$. Also they could select from a jComboBox component the domain to performing chosen (8, 16 or 32 bits). Like the option presented before, by using the button „Run algorithm” it could be visualize step by step how the integer number is being stored in the memory. In fig. 10 was choosing for demonstration a negative number. After each obtained digit and after each step of algorithm the application is stopped a while and then is restarted for a better understanding of obtaining the complement of two code.

Next options from the menu „Storing numbers in computer memory” show how it makes the store for a real number in the memory. There are two options, one for the store in floating point simple precision and another for the store in floating point double precision.

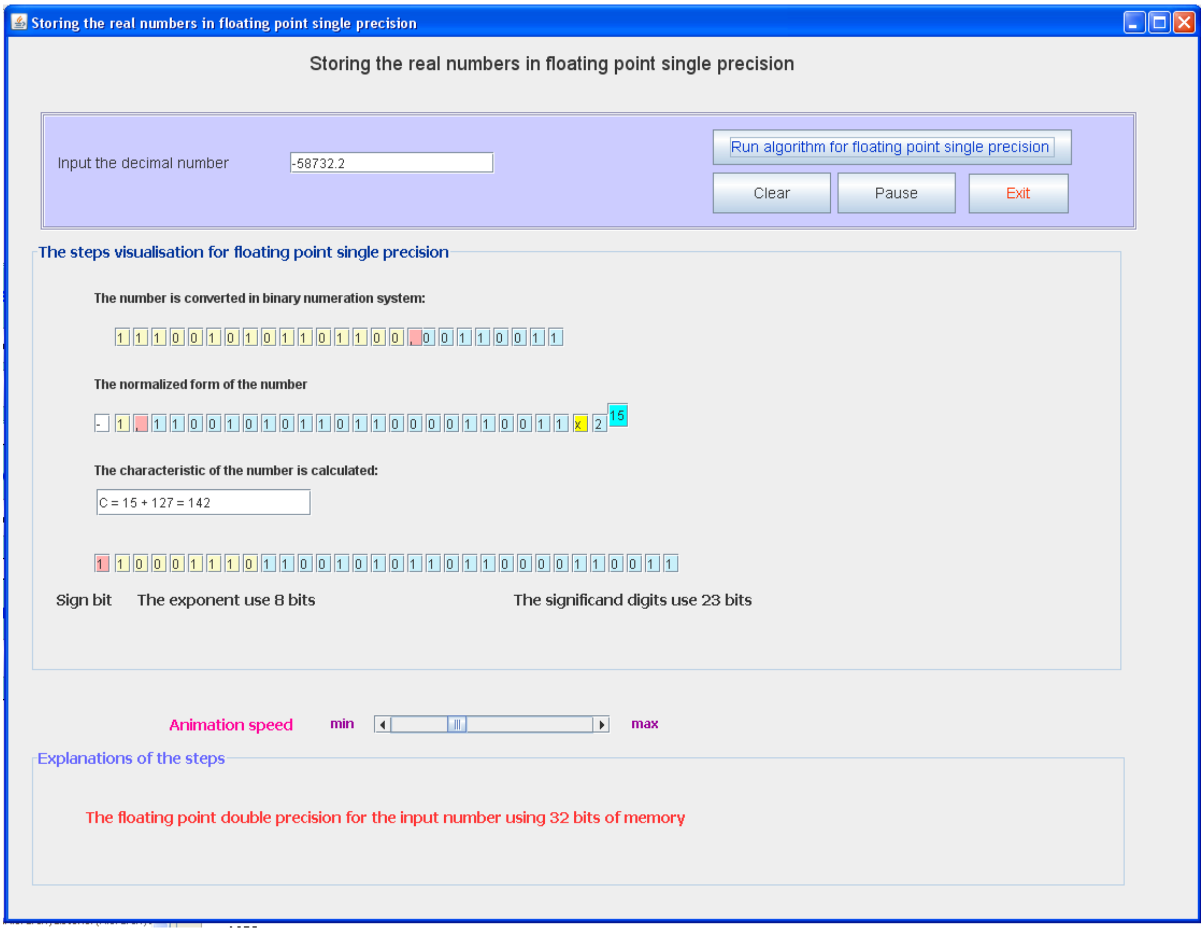
IEEE Standard 754 floating point is the most common representation today for real numbers on computers, including Intel-based PC's, Macintoshes, and most Unix platforms.

The two options from the menu are alike, the difference is the number of digits and because of that reason it will be presented only one of them and that is the one for the physical store of the numbers in floating point simple precision. The graphical user interface for that option is presented in figure 11 for floating point simple precision and in fig. 12 for floating point double precision.

It could be observed that, as well as in the case for the other options, exist the opportunity to performing the algorithm with a chosen speed for the animation. It could be stopped and than restart the animation if necessary.

Another option of the software package is for illustrating how are performed the arithmetic operation in the memory. The main arithmetic operation is summing. The graphical user interface for this operation is presented in fig. 13.

Another option of the main menu of application is for illustrating the rules for physical storing of characters in computers memory. The students can input a character, select the code (ASCII or UNICODE), and then is show the code both in binary or hexadecimal system.


Storing the real numbers in floating point single precision
Run algorithm for floating point single precision
Clear
Pause
Exit
Animation speed
min
max

Storing the real numbers in floating point single precision

Input the decimal number: -58732.2

The steps visualisation for floating point single precision

The number is converted in binary numeration system:

1 1 1 0 0 1 0 1 0 1 1 0 1 1 0 0 , 0 0 1 1 0 0 1 1

The normalized form of the number

- 1 , 1 1 0 0 1 0 1 0 1 1 0 1 1 0 0 0 0 1 1 0 0 1 1 x 2^{15}

The characteristic of the number is calculated:

C = 15 + 127 = 142

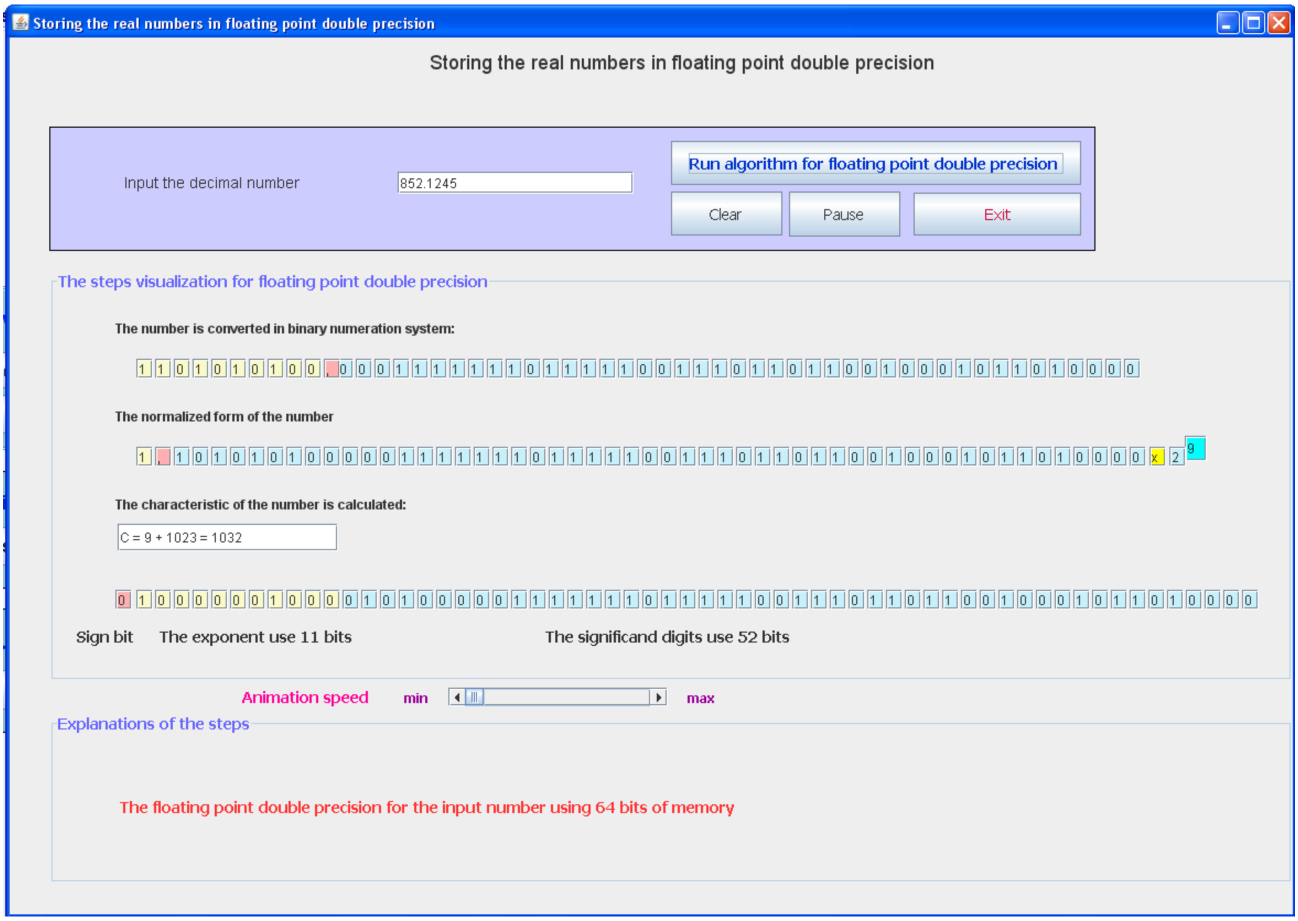
1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 0 1 1 0 1 1 0 0 0 0 1 1 0 0 1 1

Sign bit The exponent use 8 bits The significand digits use 23 bits

Explanations of the steps

The floating point double precision for the input number using 32 bits of memory

Fig. 11. The Graphical User Interface for showing the IEEE floating point standard single precision



Storing the real numbers in floating point double precision

Storing the real numbers in floating point double precision

Input the decimal number 852.1245

The steps visualization for floating point double precision

The number is converted in binary numeration system:

1 1 0 1 0 1 0 1 0 0 , 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 0 1 1 0 1 1 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 0

The normalized form of the number

1 , 1 0 1 0 1 0 1 0 0 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 0 1 1 0 1 1 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 0 x 2^9

The characteristic of the number is calculated:

C = 9 + 1023 = 1032

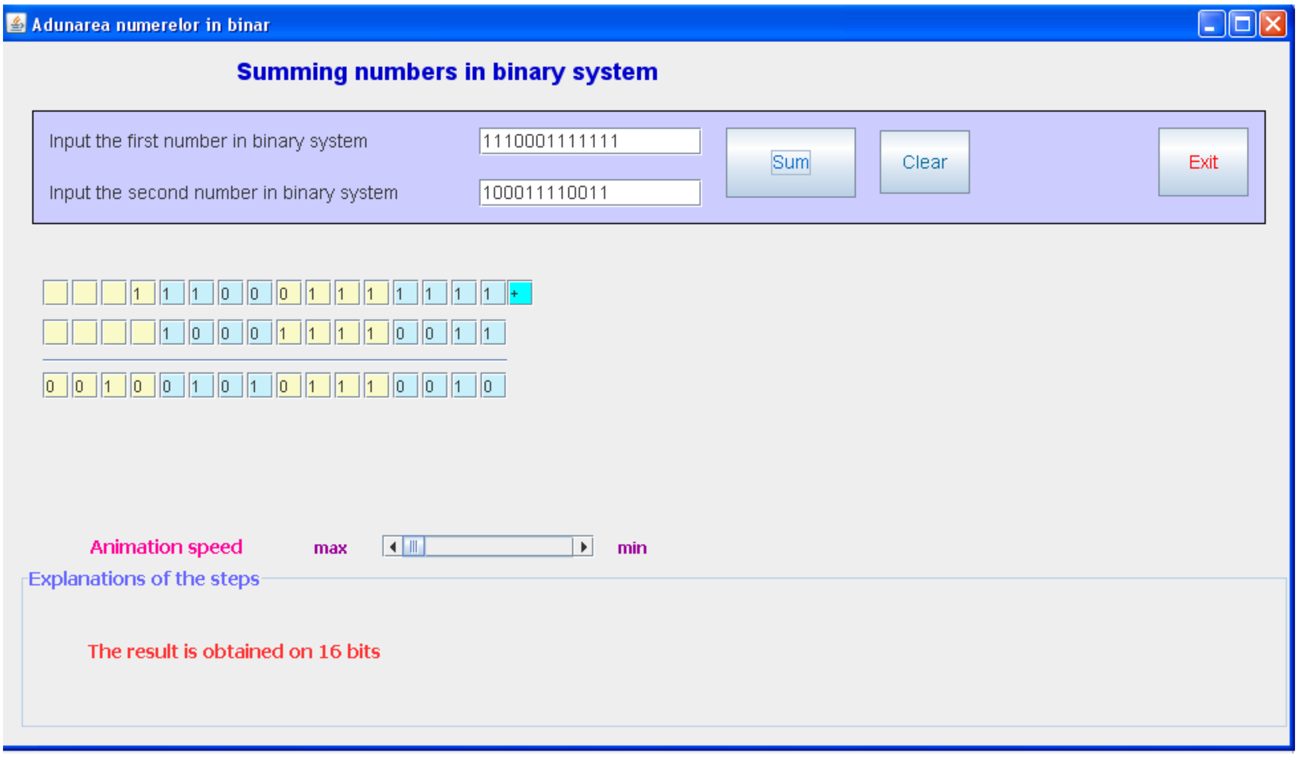
0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 0 1 1 0 1 1 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 0

Sign bit The exponent use 11 bits The significand digits use 52 bits

Explanations of the steps

The floating point double precision for the input number using 64 bits of memory

Fig. 12. The Graphical User Interface for showing the IEEE floating point standard single precision



Adunarea numerelor in binar

Summing numbers in binary system

Input the first number in binary system 1110001111111

Input the second number in binary system 100011110011

1 1 1 0 0 0 1 1 1 1 1 1 1 +

1 0 0 0 1 1 1 1 0 0 1 1

0 0 1 0 0 1 0 1 0 1 1 1 0 0 1 0

Explanations of the steps

The result is obtained on 16 bits

Fig. 13. The Graphical User Interface for summing numbers in binary system

4. Conclusions and future work

By using computer assisted learning the efficiency of learning is increased. There are some subjects in fundamentals of computer science like those presented in this software that are very difficult to learn and understand for some students. By using educational software these concepts are showing to the students in an attractive way, visually using animation and having the opportunity to interact with the application [12].

The best option is to use graphical and interactive interface of the software in two ways. On one hand, these software help the teacher in the classroom, while on the other hand, the students can work and experiment with them making their own examples, out the classroom [8] [10].

The software presented here was used for the students in Computer science from the first year of study. It was a…

Finally, as a future work, we continue developing the software in different areas of computer science. We will introduce an option for on-line evaluating of the students. This option will be used also for self-evaluating by the students.

References

[1] McDougall, A., Squires, D., „Empirical study of a new paradigm for choosing educational software", *Computer Education*, Vol. 25, no. 3, Elsevier Science, 1995.

[2] Glenn, W., Rowe, Gregor, P., \A computer based learning system for teaching computing: implementation and evaluation", *Computers and Education*, Elsevier Science, 1999.

[3] Scanlon, E., Tosunoglu, C., Jones, A., P. Butcher, S. Ross, J. Greenberg, J.,"Learning with computers: experiences of evaluation", *Computer Education* , Elsevier Science, 1998.

[4] Hanciles, B., Shankaraman, V., Munoz, J, „Multiple representations for understanding data structures", *Computer Education* vol 29. no. 1, Elsevier Science, 1997.

[5] M Fowler, K.Scott, *UML Distilled: a brief guide to the standard Object Modelling Language*, Adison Wesley, Readings MA, USA, 2000.

[6] J. Rumbauch, I. Jacobson, G. Booch, *The Unified Modeling Language reference Manual*, Adison Weslwy, 1999

[7] J. Odell, *Advanced Object Oriented Analysis & Design using UML*, Cambridge University Press, 1998

[8], C. E. Iglesias, A. G. Carbajo, M. A. S. Rosa, Interactive tools for Discrete Mathematics e-learning, *WSEAS TRANSACTIONS on ADVANCES in ENGINEERING EDUCATION*, Issue 2, vol. 5, pp. 97-103.

[9] L. M. Carmona, C. Escribano, A. Giraldo, M. A. Sastre, Interactive Web Tutorial for Integer and Modular Arithmetic and its Applications. *Proceedings of the 6th WSEAS International Conference on Education and Educational Technology (EDU'07),* Venice, Italy, November 2007

[10] M. G. Sánchez Torrubia, M. A. Sastre Rosa, V. Giménez Martínez, C. Escribano Iglesias, Visualization on Learning Mathematics Concepts for Engineering Education, *The 4th WSEAS / IASME International Conference on Engineering*

[11] Khamis N, Idris S, Issues and Solutions in Assessing Object-oriented Programming Skills in the Core Education of Computer Science and Information Technology , *12th WSEAS International Conference on Computers*, Jul 23-25, 2008 Heraklion, GREECE, Pages: 458-463, 2008

[12] Zamfir PDA, Impact of using computer applications in education on teaching-learning process, *7th WSEAS International Conference on Applied Computer and Applied Computational Science*, APR 06-08, 2008 Hangzhou, Pages: 684-688 2008

[13] Carmen Escribano Iglesias, Antonio Giraldo Carbajo, María Asunción Sastre Rosa, Calculus b-learning with java tools *WSEAS TRANSACTIONS on ADVANCES in ENGINEERING EDUCATION* Issue 5, Volume 5, May 2008, pag 295-305

[14] Diana Bri, Miguel García, Hugo Coll, Jaime Lloret, A Study of Virtual Learning Environments**,** *WSEAS TRANSACTIONS on ADVANCES in ENGINEERING EDUCATION* Issue 1, Volume 6, January 2009, pag 33-43