Interdisciplinary relations in teaching of programming

VLADIMÍR JEHLIČKA
Department of Informatics
Faculty of Education
University of Hradec Králové
Rokitanského 62, 500 03 Hradec Králové 3
CZECH REPUBLIC
vladimir.jehlicka@uhk.cz    http://www.uhk.cz/pdf

Abstract: - The presented article describes one of the potential system approaches to teaching of programming. The focus is given on the interdisciplinary interconnection between physics, informatics and mathematics. First, a simple analysis of the researched process is carried out - that means the decomposition of the light into a colour spectrum on one side and the additive composition of colours on the other side. This is followed by a mathematical description of the composition of colours from three basic components – red, green and blue - on the screen of the computer monitor. The numerical code of the resulting colour is expressed in the binary, hexadecimal and decimal numerical systems. The final part of the paper presents a simple program which enables using a computer monitor for an analysis of the additive composition of colours. It simultaneously demonstrates also advantages and disadvantages of expressing the numerical code of the composed colour in various numerical systems.

Key-Words: - decomposition of the light, additive composition of colours, numerical systems

1 Introduction

In the last years the students’ interest in humanities has been increasing in the Czech Republic, and simultaneously the interest in classical natural sciences, as mathematics, physics or chemistry, has been declining. In courses in informatics it is not problematic to give explanations concerning the basic operations of using the most varied programs. But when a certain activity has to be supported by a mathematical calculation, the students usually start being defensive, and their interest in further training vehemently declines. One of the potential causes of this situation is an isolated way of teaching mathematics as a purely theoretical subject. The not long enough time is devoted to solutions of practically aimed logical tasks and there is no continuous emphasis on interdisciplinary relations.

2 Problem Formulation

The system approach to teaching of programming will be further demonstrated on linking a task in optics with using of numerical systems in mathematics. This way, informatics, concretely programming, creates a link between physics and mathematics.

In courses in physics (the chapter concerning Optics) students are made familiar with the decomposition of the light which falls from the Sun to the Earth. This light is considered as white but its decomposition results in occurrence of the colour spectrum.

In physical laboratories the light decomposition is realized with help of a glass prism. However, students can consider this experiment as an artificially created situation. In that case they are to be referred to real life, in which they may have experienced a light decomposition on rain drops, while, simultaneously, a rainbow came into being (see Fig. 1). For the students who are more advanced in optics, a more detailed description of this natural event should be made available.

Fig. 1: A rainbow over a city

In summer months it is possible to observe a similar process of the occurrence of the rainbow also in towns, where such a decomposition of the light happens for example on water drops from fountains and waterworks, see Fig 2.

Having in mind the fact that the white light can be decomposed into a colour spectrum, we can come to a logical conclusion that the reverse process should be realizable, when the white light is re-composed from the colour spectrum.
3 Physics – an experimental additive composition of colours

In case of classical screens of colour TV sets or monitors, the final colour of each point of the screen is composed of three basic colour components - the red, green and blue one.

For a demonstration of this process it is necessary to have three sources of the light radiation with the exactly defined wave lengths and of the same light intensity. This is realizable in a highly equipped laboratory environment. But within the framework of school teaching, this is a hardly solvable problem.

The use of the Lego Mindstorms NXT 2.0 construction sets [1] can be considered as an example of simple and easily realizable experiments. A kind of a programmable construction set is concerned, which has not only several input sensors, but also output members which make e.g. a movement of the compiled programmed robots possible [2]. These construction sets are primarily intended for training of the programmers - beginners.

One of the output elements is the source of the light radiation producing a red, green or blue light. If three robots are used, when each of them radiates one of the basic colours, then it is possible to realize simple experiments with composing of colours, see Fig. 3.

As it has been stated above, no professional experimental device is concerned - a school experiment using an easily available construction set of the Lego Mindstorms NXT 2.0 is discussed. The best results can be reached in a darkened room, when the rays of the light fluxes of individual robots fall on a white surface. Fig. 4 presents, as an example, a composition of the green and red light.

4 Informatics – composing of colours on the computer screen

Within the framework of teaching of programming, the issue of solving of the quadratic equation is usually the content of one of the first seminars. A typical task, suitable for a presentation of a program branching, is concerned. However, students’ relation to programming is negatively influenced due to a purely mathematical essence of the solved problem. On the contrary, tasks which attract students’ attention because of a practical use, as it is for example stereoscopic displaying of three-dimension objects [3], are accepted very positively.

While teaching, it is not possible to skip some needed but sometimes not enjoyable topics. A sensitive approach to students is essential, as well as a continual progress from easy tasks to more difficult ones and a constant focus on keeping students’ attention. Within the framework of the computer graphics [4] it is thus suitable to deal with the creation of colours.
On a computer monitor it is possible to realize a composition of colours in various programming languages. With respect to our purposes, the development environment Delphi has been chosen as it enables to create very well-arranged and transparent programs.

As it has been stated, each point on the screen has its own colour which is composed of three basic components: red, green and blue ones. The resulted colour is expressed with a numeral which is stored in the computer memory. For generating of this colour it is possible to use e.g. the function \( \text{rgb}(\text{Red}, \text{Green}, \text{Blue}) \), where the individual parameters of the function can be of the value from 0 to 255 and they express the intensity of the given colour component. The resulted colour is expressed with a numeral which is stored in the computer memory. For generating of this colour it is possible to use e.g. the function \( \text{rgb}(\text{Red}, \text{Green}, \text{Blue}) \), where the individual parameters of the function can be of the value from 0 to 255 and they express the intensity of the given colour component. The numerical values of the three given parameters can be entered in the decimal system, so it is not essential to deal with the real way of storing the colour numbers in the computer memory. However, a deeper knowledge of the computer work, concerning the way of storing data in its memory, is useful not only for future professional programmers.

It is generally known that the smallest part of the computer memory is created by one bit, into which it is possible to write only the values zero or one. A bigger memory unit is one byte, which includes 8 bits. For storing of the numeral which expresses the colour of the particular point on the computer monitor it is necessary to allocate minimally 3 bytes, see Table 1.

<table>
<thead>
<tr>
<th>Blue</th>
<th>Green</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### Table 1: Storing of the colour number in the computer memory

5 Mathematics – numerical systems

Table 1 presents a content of a part of the computer memory, where number one is stored in all the appropriate bits. That means that all the three basic colour components have the maximal intensity, which is expressed by the numerical value of 255. The resulting colour is white. If, on the other hand, the value of zero was stored in all the bits, then all the three colour components would have the zero light intensity and the resulting colour would be black.

Let us come back to the above given value 255. Why is specifically this value given? Let us realize that for each colour component there is a limitation of one byte, i.e. 8 bits, in which just 0 or 1 can be stored. That means, into the part of the memory which is limited in this way it is possible to store maximally eight-place binary numerals. The value of the highest eight-place binary numeral expressed in the decimal system is:

\[
2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255
\]  
(1)

If each of the three colour components can have the value from 0 to 255, then the given system enables to define

\[
255^3 = 16,777,215
\]  
(2)

colour tones in total.

The white colour can be numerically expressed in the binary system as \(111111111111111111111111\) or in the decimal system as \(16,777,215\). It is obvious that both the ways are not transparent because the light intensity of the individual colour components is not visible at the first sight.

The optimal solution is to express the number of the colour in the hexadecimal system. Each colour component occupies 8 bits in the computer memory, which corresponds to an eight-place numeral written down in the binary numerical system or a two-place numeral expressed in the hexadecimal numerical system. The maximal intensity of the colour component is thus expressed in the given numerical systems as follows:

\[
11111111 = \text{FF}_{16} = 255_{10}
\]  
(3)

The white colour, which is written down in the binary system in Tab. 1, can be written in a more transparent way in the hexadecimal system, see Tab. 2.

<table>
<thead>
<tr>
<th>F</th>
<th>F</th>
<th>F</th>
<th>F</th>
<th>F</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue</td>
<td>green</td>
<td>red</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 2: Expressing of the white colour in the hexadecimal system

6 Programming – an exemplary program

A simple program called “Color” has been created for a transparent presentation of the additive composition of colours in the Delphi environment. The following print-out of the basic unit presents the concrete declaration of the type of the used form, and, above all, the sub-program \( \text{TForm1.FormCreate} \), which is evoked not only when the form is opened but also as a reaction to a shift of the sliding box of any scroll bars.

```pascal
unit Unit1;
interface
```
uses
  Windows, Messages, SysUtils,
  Variants, Classes, Graphics,
  Controls, Forms, Dialogs,
  ExtCtrls, StdCtrls;

const
  Hexa: array[0..15] of String[1] =
  ('0','1','2','3','4','5','6','7',
   '8','9','A','B','C','D','E','F');

type
  TForm1 = class(TForm)
    Image1: TImage;
    ScrollBar1: TScrollBar;
    ScrollBar2: TScrollBar;
    ScrollBar3: TScrollBar;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.FormCreate(Sender: TObject);
begin
  with Image1.canvas do
  begin
    brush.Color:=rgb(0,0,0);
    pen.Mode:=pmcopy;
    Rectangle(0,0,400,400);
    pen.width:=0;
    pen.Mode:=pmmerge;
    brush.Color:=rgb(ScrollBar1.Position,0,0);
    Ellipse(0,0,200,200);
    brush.Color:=rgb(0,ScrollBar2.Position,0);
    Ellipse(100,0,300,200);
    brush.Color:=rgb(0,0,ScrollBar3.Position);
    Ellipse(50,100,250,300);
    Label4.Caption:=IntToStr(ScrollBar1.Position);
    Label5.Caption:=IntToStr(ScrollBar2.Position);
    Label6.Caption:=IntToStr(ScrollBar3.Position);
    Label7.Caption:='RGB('+
    IntToStr(ScrollBar1.Position)+', '+
    IntToStr(ScrollBar2.Position)+', '+
    IntToStr(ScrollBar3.Position)+')';
    Label8.Caption:='$00'+'
    Hexa[(ScrollBar3.Position div 16)]
    +Hexa[(ScrollBar3.Position mod 16)]
    +Hexa[(ScrollBar2.Position div 16)]
    +Hexa[(ScrollBar2.Position mod 16)]
    +Hexa[(ScrollBar1.Position div 16)]
    +Hexa[(ScrollBar1.Position mod 16)]
    +' ... hexadecimal';
    +256*ScrollBar2.Position
    +ScrollBar1.Position)
    +' ... decimal';
  end;
end.

If any of the above given situations come into being, then the subprogram TForm1.FormCreate first draws a black coloured square and, followingly, three mutually overlaying colour circles, see Fig. 5.

To make the composition of colours possible, the Mode parameter of the drawing pen is necessary to be set from the original form pen.Mode:=pmcopy; to a new form pen.Mode:=pmmerge. When individual circles are coloured, then the colour of the circle being coloured is added to the colour of the background. Thus the numerical values of the appropriate colours are added and the resulting number gives the number of the colour filling the given area. This way, the additive composition of colours is made possible on the monitor screen.

The intensities of the composed colours can be set with help of sliding boxes on the appropriate scroll bars marked with red, green and blue colours. The current position of the sliding box is in case of each scroll bar written through a numeral in the decimal numerical system.
The area in which all the three colour circles are overlaying is coloured in the resulting summation colour. Its numerical expression is given not only individually in the RGB code, where the contributions of the individual colours are listed in the decimal system, but also as a resulting numerical value of the colour in the hexadecimal and also in the decimal numerical system.

7 Using of the “Color” program while teaching

The “Color” Program can be used as a suitable motivation for students, which supports understanding of numerical systems and mutual transformations of numerical expressions. As it has been already shown in Tab 1 and in relation (3), the expression of numbers in the binary system is hardly transparent. Then it is suitable to give a focus on a comparison of the entries in the decimal and hexadecimal numerical systems. A survey of the basic colours and their numerical expressions are given in Table 3.

Let us imagine a situation when we are to create a program which would simulate switching on and switching off of three colour reflectors and through that it would create continual colour transitions.

Table 3: Numerical expression of the basic colours

<table>
<thead>
<tr>
<th>Colour</th>
<th>Decimal</th>
<th>Hexadecimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>0</td>
<td>$00000000$</td>
</tr>
<tr>
<td>Red</td>
<td>255</td>
<td>$000000FF$</td>
</tr>
<tr>
<td>Green</td>
<td>65 280</td>
<td>$0000FF00$</td>
</tr>
<tr>
<td>Blue</td>
<td>16 711 680</td>
<td>$00FF0000$</td>
</tr>
<tr>
<td>White</td>
<td>16 777 215</td>
<td>$00FFFFFF$</td>
</tr>
</tbody>
</table>

Let us start from the black colour and let us gradually increase the intensity of the red light from the minimal value of 0 to the maximal value of 255. The value of 1 will be added in the decimal system in the cycle from 1 to 255, or the value of $00000001$ will be added in the hexadecimal system. In both the systems the expression of colours is comparable.

Let us make the same consideration for the green colour. In that case in each step of the given cycle the value of 256 will be added in the decimal system or the value of $00000100$ will be added in the hexadecimal system. The advantage of the hexadecimal system is already quite obvious here.

If the same consideration is made for the blue colour, then in each step of the given cycle the value of $256^2 = 65536$ will be added in the decimal system or the value of $00010000$ will be added in the hexadecimal system. The advantage of the hexadecimal system is then absolutely without any discussion.

And what would be a continual transition like (e.g. from the blue to red the colour)? The numerical values of both the colours are given in Table 3. In the hexadecimal system it is started from the value of $00FF0000$. The blue light is gradually reduced, that means the value of $00010000$ will be subtracted. Simultaneously, the red light will be added, thus the value of $00000001$ will be added. Let us suppose that the up-dated value of our colour will be stored into the variable called MyColor. Then in each step the new value of the colour will be set through the order:

MyColor:=MyColor-$00010000$+$00000001$;

If this operation is carried out 255 times in total in the cycle, then the numerical code of the resulting colour will be $000000FF$; that means the resulting colour will be the required red colour.

It is obvious that in this case it is not sensible to deal with the entry of the above given order with help of the decimal numerical system although the task is solvable in any numerical system.
The “Color” Program can be used for demonstration of the conversion of numerals from the decimal system into the hexadecimal one. This conversion is realized with help of the integral division and calculation of the division remainder. Concretely, the following commands are concerned:

ScrollBar3.Position div 16
ScrollBar3.Position mod 16

8 Conclusion

The system approach to the way of teaching of programming has been proved to be worth realizing. The emphasis on the interdisciplinary relations between individual subjects has been proved as increasing students’ attention and, simultaneously, also their motivation for their independent activities.

The privileged position of informatics is given by the fact that, for example, within the framework of programming it enables to interconnect an experimental teaching subject (e.g. physics) with a purely theoretical mathematics. On the given example it is possible to demonstrate the use of a relatively arid issue of numerical systems while practically creating simple programs with created colour definitions. Manifold varieties of simple school tasks linked with the given issue are quite fruitful and they have been accepted in a very positive way by students.

References: