Lucrare pentru examenul de “Programarea Calculatoarelor” an I

Pop-Andries Alexandru
Calculatoare I
Grupa II

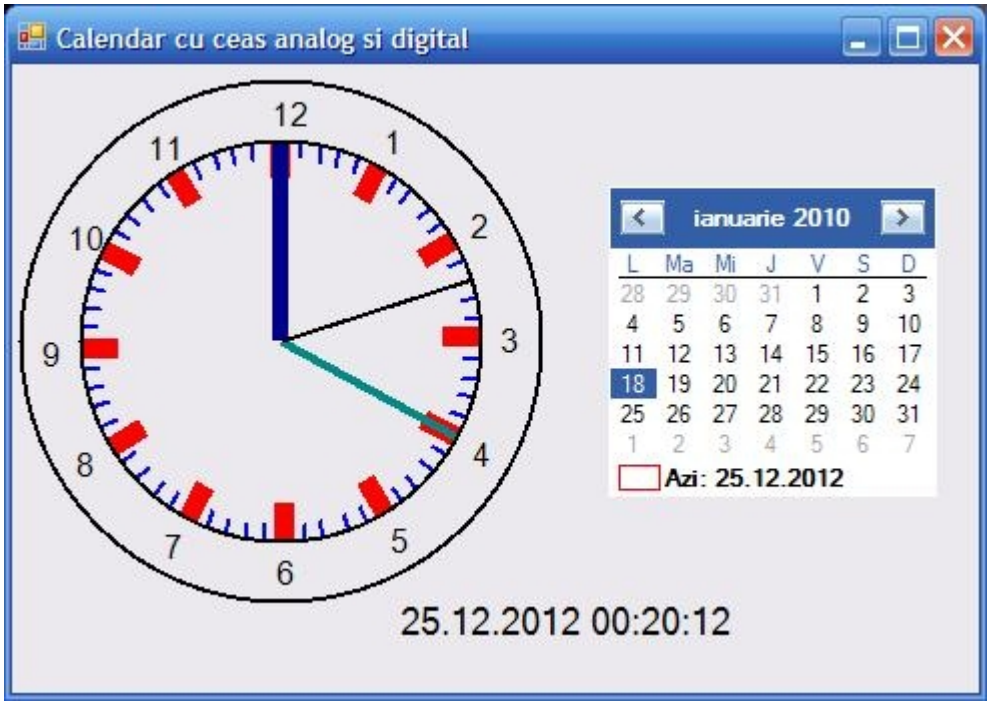
Lucrarea #3

Descriere:

Aplicatia contine un calendar care afiseaza data curenta dar utilizatorii pot vedea si restul calendarului. In partea stanga a calendarului se afla un ceas in stil analog. Cu aratatoare pentru ora, minut si secunda. Sub acestea este afisata data si ora asa cum sunt ele date de sistem.

Aplicatia a fost creata in visual c++ 2005.

Interfata programului arata in felul urmator:



Codul sursa:

```
#pragma once


namespace Calendar_cu_ceas {

	using namespace System;
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;
int centru[3],raza,ceas[301][301],contor=0,secunda_,minut_,ora_,inv=0;
int*** ceas2;
int ora_min_sec[60][3];
	/// <summary>
	/// Summary for Form1
	///
	/// WARNING: If you change the name of this class, you will need to change the
	///          'Resource File Name' property for the managed resource compiler tool
	///          associated with all .resx files this class depends on.  Otherwise,
	///          the designers will not be able to interact properly with localized
	///          resources associated with this form.
	/// </summary>
	public ref class Form1 : public System::Windows::Forms::Form
	{
	public:
		Form1(void)
		{
			InitializeComponent();
			//
			//TODO: Add the constructor code here
			//
		}

	protected:
		/// <summary>
		/// Clean up any resources being used.
		/// </summary>
		~Form1()
		{
			if (components)
			{
				delete components;
			}
		}

	private: System::Windows::Forms::PictureBox^  pictureBox1;

	private: System::Windows::Forms::Timer^  timer1;
	private: System::Windows::Forms::Label^  label1;
	private: System::Windows::Forms::MonthCalendar^  monthCalendar1;

	private: System::ComponentModel::IContainer^  components;


	protected:

	private:
		/// <summary>
		/// Required designer variable.
```

```
        /// </summary>


#pragma region Windows Form Designer generated code
        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        void InitializeComponent(void)
        {
            this->components = (gcnew System::ComponentModel::Container());
            this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());
            this->timer1 = (gcnew System::Windows::Forms::Timer(this->components));
            this->label1 = (gcnew System::Windows::Forms::Label());
            this->monthCalendar1 = (gcnew System::Windows::Forms::MonthCalendar());
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this->pictureBox1))->BeginInit();
            this->SuspendLayout();
            //
            // pictureBox1
            //
            this->pictureBox1->Location = System::Drawing::Point(-16, -12);
            this->pictureBox1->Name = L"pictureBox1";
            this->pictureBox1->Size = System::Drawing::Size(302, 302);
            this->pictureBox1->TabIndex = 1;
            this->pictureBox1->TabStop = false;
            //
            // timer1
            //
            this->timer1->Enabled = true;
            this->timer1->Interval = 1000;
            this->timer1->Tick += gcnew System::EventHandler(this, &Form1::timer1_Tick);
            //
            // label1
            //
            this->label1->AutoSize = true;
            this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 14.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(0)));
            this->label1->Location = System::Drawing::Point(190, 266);
            this->label1->Name = L"label1";
            this->label1->Size = System::Drawing::Size(93, 24);
            this->label1->TabIndex = 2;
            this->label1->Text = L"Date & time";
            //
            // monthCalendar1
            //
            this->monthCalendar1->Location = System::Drawing::Point(298, 61);
            this->monthCalendar1->Name = L"monthCalendar1";
            this->monthCalendar1->TabIndex = 3;
            //
            // Form1
            //
            this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);
            this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
            this->ClientSize = System::Drawing::Size(483, 314);
            this->Controls->Add(this->monthCalendar1);
            this->Controls->Add(this->label1);
            this->Controls->Add(this->pictureBox1);
            this->Name = L"Form1";
            this->Text = L"Calendar cu ceas analog si digital";
            this->Load += gcnew System::EventHandler(this, &Form1::Form1_Load);
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this->pictureBox1))-
```

```
>EndInit();
                this->ResumeLayout(false);
                this->PerformLayout();

            }


            System::String^ sec_a;
            System::String^ sec_b;
            System::String^ min_a;
            System::String^ min_b;
            System::String^ ora_a;
            System::String^ ora_b;
            System::String^ stil;
            System::String^ AMPM;
            int AM_PM;

#pragma endregion

private: System::Void timer1_Tick(System::Object^  sender, System::EventArgs^  e) {

                    System::Drawing::Graphics^ Desen;
                    System::Drawing::Pen^ Pix=gcnew
System::Drawing::Pen(System::Drawing::Color::Blue,2);
                    System::Drawing::Pen^ Pix_sec=gcnew
System::Drawing::Pen(System::Drawing::Color::Black,2);
                    System::Drawing::Pen^ Pix_min=gcnew
System::Drawing::Pen(System::Drawing::Color::DarkCyan,4);
                    System::Drawing::Pen^ Pix_ora=gcnew
System::Drawing::Pen(System::Drawing::Color::DarkBlue,8);
                    System::Drawing::Pen^ Pix2=gcnew
System::Drawing::Pen(System::Drawing::Color::Red,10);
                    System::Drawing::SolidBrush^ Pensula=gcnew
System::Drawing::SolidBrush(System::Drawing::Color::Black);
                    Desen=this->pictureBox1->CreateGraphics();
                    Desen->Clear(System::Drawing::Color(this->BackColor));
                    System::DateTime ora2=System::DateTime::Now;
                    System::String^ str_ora=System::Convert::ToString(ora2);

                    /*for(int k=0;k<=contor-1;k+=1)
                    for(int x=0;x<=300;x+=1)
                            for(int y=0;y<=300;y+=1)
                                    if(ceas2[k][x][y]==1)
                                    {
                                            Desen->DrawEllipse(Pix,x,y,1,0);
                                            Desen->DrawEllipse(Pix,x,y,0,1);
                                    }*/

                    /*for(int x=0;x<60;x+=1)
                                    if(ora[x][0]==1)
                                    {
                                            Desen->DrawEllipse(Pix,ora[x][1],ora[x][2],10,10);
                                            Desen->DrawEllipse(Pix,ora[x][1],ora[x][2],10,10);
                                    }*/

                    {//////////////////////////
                            int cX=0,cY=0,procent=0,lungime=15;
                    for(int x=0;x<60;x+=1)
                                    if(ora_min_sec[x][0]==1)
                                    {

                                            if(ora_min_sec[x][1]<150)
                                            {
```

```
                procent=(150-ora_min_sec[x][1])*100/150;
                procent=(procent*lungime)/100;
                cX=ora_min_sec[x][1]+procent;
        }
        if(ora_min_sec[x][1]>150)
        {
                procent=(ora_min_sec[x][1]-150)*100/150;
                procent=(procent*lungime)/100;
                cX=ora_min_sec[x][1]-procent;
        }
        if(ora_min_sec[x][1]==150) cX=ora_min_sec[x][1];


        if(ora_min_sec[x][2]<150)
        {
                procent=(150-ora_min_sec[x][2])*100/150;
                procent=(procent*lungime)/100;
                cY=ora_min_sec[x][2]+procent;
        }
        if(ora_min_sec[x][2]>150)
        {
                procent=(ora_min_sec[x][2]-150)*100/150;
                procent=(procent*lungime)/100;
                cY=ora_min_sec[x][2]-procent;
        }
        if(ora_min_sec[x][2]==150) cY=ora_min_sec[x][2];

        Desen->DrawLine(Pix,cX,cY,ora_min_sec[x]
[1],ora_min_sec[x][2]);
                }
}////////////////////////////

{//////////////////////////
        int cX=0,cY=0,procent=0,lungime=30;
for(int x=0;x<60;x+=5)
                if(ora_min_sec[x][0]==1)
                {

                        if(ora_min_sec[x][1]<150)
                        {
                                procent=(150-ora_min_sec[x][1])*100/150;
                                procent=(procent*lungime)/100;
                                cX=ora_min_sec[x][1]+procent;
                        }
                        if(ora_min_sec[x][1]>150)
                        {
                                procent=(ora_min_sec[x][1]-150)*100/150;
                                procent=(procent*lungime)/100;
                                cX=ora_min_sec[x][1]-procent;
                        }
                        if(ora_min_sec[x][1]==150) cX=ora_min_sec[x][1];


                        if(ora_min_sec[x][2]<150)
                        {
                                procent=(150-ora_min_sec[x][2])*100/150;
                                procent=(procent*lungime)/100;
                                cY=ora_min_sec[x][2]+procent;
                        }
                        if(ora_min_sec[x][2]>150)
                        {
                                procent=(ora_min_sec[x][2]-150)*100/150;
                                procent=(procent*lungime)/100;
```

```
                                                cY=ora_min_sec[x][2]-procent;
                                        }
                                        if(ora_min_sec[x][2]==150) cY=ora_min_sec[x][2];

                                        Desen->DrawLine(Pix2,cX,cY,ora_min_sec[x]
[1],ora_min_sec[x][2]);
                                }
                        }////////////////////////////

                        {//////////////////////////
                                int cX=0,cY=0,procent=0,lungime=-30,prima_data=1;
                        for(int x=0;x<60;x+=5)
                                if(ora_min_sec[x][0]==1)
                                {

                                        if(ora_min_sec[x][1]<200)
                                        {
                                                procent=(200-ora_min_sec[x][1])*100/200;
                                                procent=(procent*lungime)/100;
                                                cX=ora_min_sec[x][1]+procent;
                                        }
                                        if(ora_min_sec[x][1]>200)
                                        {
                                                procent=(ora_min_sec[x][1]-200)*100/200;
                                                procent=(procent*lungime)/100;
                                                cX=ora_min_sec[x][1]-procent;
                                        }
                                        if(ora_min_sec[x][1]==200) cX=ora_min_sec[x][1];


                                        if(ora_min_sec[x][2]<200)
                                        {
                                                procent=(200-ora_min_sec[x][2])*100/200;
                                                procent=(procent*lungime)/100;
                                                cY=ora_min_sec[x][2]+procent;
                                        }
                                        if(ora_min_sec[x][2]>200)
                                        {
                                                procent=(ora_min_sec[x][2]-200)*100/200;
                                                procent=(procent*lungime)/100;
                                                cY=ora_min_sec[x][2]-procent;
                                        }
                                        if(ora_min_sec[x][2]==200) cY=ora_min_sec[x][2];

                                        System::Drawing::Font^ font_arial=gcnew
System::Drawing::Font("Arial",12);


                                        if(prima_data==1)
                                        {
                                                Desen-
>DrawString("12",font_arial,Pensula,cX,cY);
                                                prima_data=0;
                                        }
                                        else Desen-
>DrawString(System::Convert::ToString(x/5),font_arial,Pensula,cX,cY);
                                }
                        }////////////////////////////



                                AMPM=str_ora->Remove(0,(str_ora->Length)-2);
                                if(AMPM=="PM"||AMPM=="AM")
```

```
                                        {
                                                AM_PM=3;
                                        }
                                        else AM_PM=0;

                                        sec_a=str_ora->Remove(0,(str_ora->Length)-1-AM_PM);
                                        if(AMPM=="PM"||AMPM=="AM") sec_a=sec_a->Remove((sec_a-
>Length)-AM_PM);

                                        sec_b=str_ora->Remove(0,(str_ora->Length)-2-AM_PM);
                                        sec_b=sec_b->Remove((sec_b->Length)-1-AM_PM);

                                        min_a=str_ora->Remove(0,(str_ora->Length)-4-AM_PM);
                                        min_a=min_a->Remove((min_a->Length)-3-AM_PM);

                                        min_b=str_ora->Remove(0,(str_ora->Length)-5-AM_PM);
                                        min_b=min_b->Remove((min_b->Length)-4-AM_PM);

                                        ora_a=str_ora->Remove(0,(str_ora->Length)-7-AM_PM);
                                        ora_a=ora_a->Remove((ora_a->Length)-6-AM_PM);

                                        ora_b=str_ora->Remove(0,(str_ora->Length)-8-AM_PM);
                                        ora_b=ora_b->Remove((ora_b->Length)-7-AM_PM);
                                        if(ora_b==" ") ora_b="0";

                                        secunda_=System::Convert::ToInt32(sec_b+sec_a);
                                        minut_=System::Convert::ToInt32(min_b+min_a);
                                        ora_=System::Convert::ToInt32(ora_b+ora_a);

                                        {
                                        int x=0;
                                        if(ora_>12) ora_-=12;
                                        x=ora_;
                                        if(x!=12)Desen->DrawLine(Pix_ora,150,150,ora_min_sec[x*5]
[1],ora_min_sec[x*5][2]);
                                        else Desen->DrawLine(Pix_ora,150,150,ora_min_sec[0]
[1],ora_min_sec[0][2]);
                                        x=minut_;
                                        Desen->DrawLine(Pix_min,150,150,ora_min_sec[x][1],ora_min_sec[x]
[2]);
                                        x=secunda_;
                                        Desen->DrawLine(Pix_sec,150,150,ora_min_sec[x][1],ora_min_sec[x]
[2]);
                                        }
                                        this->label1->Text=System::Convert::ToString(ora2);

                        Desen->DrawEllipse(Pix_sec,50,50,200,200);
                        Desen->DrawEllipse(Pix_sec,20,20,260,260);

                        }
private: System::Void Form1_Load(System::Object^  sender, System::EventArgs^  e) {
                        centru[1]=150;
                        centru[2]=150;
                        raza=100;
                        contor=0;
                int x=0,y=0;
                for(x=0;x<=300;x+=1)
                {
                        for(y=0;y<=300;y+=1)
                        {
                                if ( System::Math::Truncate(System::Math::Sqrt(  System::Math::Pow((x-centru[1]),2) +
System::Math::Pow((y-centru[2]),2) ))==raza )
                                {
```

```
                        ceas[x][y]=1;
                        contor+=1;
                }
                else ceas[x][y]=0;
        }
}

ceas2=new int**[contor];
for(int x=0;x<contor;x++)
{
        ceas2[x]=new int*[301];
        for(int y=0;y<301;y++)
        {
                ceas2[x][y]=new int[301];
        }
}
for(int k=0;k<=contor-1;k+=1) for(x=0;x<=300;x+=1) for(y=0;y<=300;y+=1) ceas2[k][x][y]=0;

//primul punct x==150 y<150
for(int y=0;y<150;y+=1) if(ceas[150][y]==1) ceas2[0][150][y]=1;

{//inceput bloc cu int k=1;
int k=1;

// x>150 y<150 zona 1
for(int x=151;x<=300;x+=1)
{
        for(int y=0;y<150;y+=1)
        {
                if(ceas[x][y]==1)
                {
                        ceas2[k][x][y]=1;
                        k+=1;
                }
        }
}

//limita dintre zona 1 si zona 2 x>150 y==150
for(int x=151;x<300;x+=1) if(ceas[x][150]==1) ceas2[k][x][150]=1;

// x>150 y>150 zona 2
for(int y=151;y<=300;y+=1)
{
        for(int x=151;x<=300;x+=1)
        {
                if(ceas[x][y]==1)
                {
                        ceas2[k][x][y]=1;
                        k+=1;
                }
        }
}


//limita dintre zona 2 si zona 3 x==150 y>150
for(int y=151;y<300;y+=1) if(ceas[150][y]==1) ceas2[k][150][y]=1;

// x<150 y>150 zona 3
for(int x=149;x>=0;x-=1)
{
        for(int y=300;y>150;y-=1)
        {
                if(ceas[x][y]==1)
```

```
                {
                        ceas2[k][x][y]=1;
                        k+=1;
                }
        }
}

//limita dintre zona 3 si zona 4 x<150 y==150
for(int x=0;x<150;x+=1) if(ceas[x][150]==1) ceas2[k][x][150]=1;

// x<150 y<150 zona 4
for(int y=149;y>=0;y-=1)
{
        for(int x=0;x<150;x+=1)
        {
                if(ceas[x][y]==1)
                {
                        ceas2[k][x][y]=1;
                        k+=1;
                }
        }
}

{//////////////////////////////////////
        int q=0,k=0,x=0,y=0,abc=0,br=0,def=contor,def2=0;
        def-=10;
        def2=System::Math::Truncate(contor/60);
        for(k=0;k<=def;k+=def2)
                for(x=0;x<=300;x+=1)
                        for(y=0;y<=300;y+=1)
                        {
                                //if(q==29){q+=1;inv=1;}
                                //if(q==30&&inv==2){q-=1;inv=0;}
                                if(ceas2[k][x][y]==1&&abc==0)
                                {
                                        if(q==29)
                                        {int q=30;
                                                ora_min_sec[q][0]=1;
                                                ora_min_sec[q][1]=x;
                                                ora_min_sec[q][2]=y;
                                        }
                                        if(q==30)
                                        {int q=29;
                                                ora_min_sec[q][0]=1;
                                                ora_min_sec[q][1]=x;
                                                ora_min_sec[q][2]=y;
                                        }
                                        if(q!=29&&q!=30)
                                        {
                                                ora_min_sec[q][0]=1;
                                                ora_min_sec[q][1]=x;
                                                ora_min_sec[q][2]=y;
                                        }
                                        q+=1;


                                }
                                //if(inv==1){q-=1; inv=2;}
                        }

}//////////////////////////////////////////
```

```
            delete [] **ceas2;
            delete [] *ceas2;
            delete [] ceas2;

            }// sfarsit bloc cu int k=1;
                        }
};
}
```