Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Кафедра прикладной математики и программирования

ОТЧЕТ

о выполнении лабораторной работы № 6 по дисциплине
«Математические основы компьютерной графики»

Автор работы,
студент группы ЕТ-212
_____ Ныркова Е.А.
«____» _____ 2022 г.

Руководитель работы,
старший преподаватель
_____ Шелудько А.С.
«____» _____ 2022 г.

Челябинск 2022

1 ЗАДАНИЕ

Написать программу для выполнения аффинных преобразований многоугольника на плоскости. Предварительно определить структуру данных (класс) и разработать соответствующие подпрограммы (методы). Число и координаты вершин многоугольника считать из файла. Интерфейс программы должен содержать следующие элементы управления

- перемещение фигуры;
- поворот фигуры (относительно центра фигуры);
- растяжение/сжатие фигуры;
- сохраение результата в файл;
- выход из программы.

2 ОПИСАНИЕ КЛАССА FIGURE

В кдассе Polygon находятся объекты:

- double *x;
- double *y;
- double x0, y0;
- int n;
- int *t;

Указатель t указывает на массив, используемый при рисовании фигуры. Методы класса:

- Polygon (int, double, double, double *,double *);
- Polygon();
- void Display(int);
- void ChangePos(int, int);
- void Rotate(double).

Rotate(double A) - поворачивает фигуру вокруг своего центра на A градусов. Реализация класса воплощается в файле "task.cpp".

Файл main.cpp

```
#define _USE_MATH_DEFINES
#include <math.h>
#include "graphics.h"
#include "control.h"
#include "task.h"

void init_controls(){
   init_control(UP,0,0,"up.bmp");
   init_control(DOWN,100,0,"down.bmp");
   init_control(RIGHT,200,0,"right.bmp");
   init_control(LEFT,300,0,"left.bmp");
   init_control(L_SPIN,400,0,"lspin.bmp");
   init_control(R_SPIN,500,0,"rspin.bmp");
   init_control(U_SCALE,600,0,"uscale.bmp");
   init_control(D_SCALE,700,0,"dscale.bmp");
   init_control(SAVE,700,700,"save.bmp");
   init_control(EXIT,600,700,"exit.bmp");
}

void draw_interface(){
   for(int i=0;i<N_CONTROLS;i++){
      draw_image(i);
   }
}

int main()
{
   initwindow(800, 800);
   setbkcolor(BLUE);
   clearviewport();

   double *tempx, *tempy;
   tempx=new double [8];
   tempy=new double [8];

   tempx[0]=100;
   tempx[1]=200;
   tempx[2]=300;
   tempx[3]=300;
   tempx[4]=200;
   tempx[5]=100;
   tempx[6]=0;
   tempx[7]=0;

   tempy[0]=0;
   tempy[1]=0;
   tempy[2]=100;
   tempy[3]=200;
```

```
tempy[4]=300;
tempy[5]=300;
tempy[6]=200;
tempy[7]=100;

Figure f(8,100,100,tempx,tempy);
init_controls();
int p=0;
while (true){


   p=1-p;
   setactivepage(p);
   clearviewport();
   f.Draw(COLOR(34,177,76));
   if(mousebuttons()){
      switch(select_control()){
         case NONE:
            break;
         case UP:
            f.GoTo(0,-10);
            break;
         case DOWN:
            f.GoTo(0,10);
            break;
         case RIGHT:
            f.GoTo(10,0);
            break;
         case LEFT:
            f.GoTo(-10,0);
            break;
         case L_SPIN:
            f.Spin(-M_PI/8);
            break;
         case R_SPIN:
            f.Spin(M_PI/8);
            break;
         case U_SCALE:
            f.Scale(1.2);
            break;
         case D_SCALE:
            f.Scale(0.8);
            break;
         case SAVE:
            save();
            break;
         case EXIT:
            goto M;
      }
   }
   draw_interface();
   setvisualpage(p);
```

```
        delay(30);
    }
    M:
    delete[]tempx;
    delete[]tempy;
    free_image();
    closegraph();
}
```

Файл task.h

```
#ifndef TASK_HPP
#define TASK_HPP
class Figure{
   double *x;
   double *y;
   double x0,y0;
   int n;
   public:
   Figure(int,double,double,double *,double *);
   ~Figure();
   void Draw(int);
   void GoTo(int,int);
   void Scale(double);
   void Spin(double);
};
void save();
#endif
```

Файл task.cpp

```
#define _USE_MATH_DEFINES
#include <math.h>
#include "graphics.h"
#include "task.h"
bool debug=false;
Figure::Figure(int N,double X0,double Y0,double *X,double *Y){
   x=X;
   y=Y;
   x0=X0;
   y0=Y0;
   n=N;
}

Figure::~Figure(){
   delete [] x;
   delete [] y;
}

void Figure::Draw(int C){
   setfillstyle(SOLID_FILL,C);
   int *temp;
   temp=new int[2*n]();
   for(int j=0,i=0;j<n;i+=2,j++){
      temp[i]=x[j];
      temp[i+1]=y[j];
   }
   fillpoly(n,temp);
   setfillstyle(SOLID_FILL,GREEN);
```

```
    if(debug){
        for(int i=0;i<n;i++){
            fillellipse(x[i],y[i],5,5);
            delay(200);
        }
        fillellipse(x0,y0,5,5);
    }
    delete [] temp;
}

void Figure::GoTo(int dx,int dy){
    for(int i=0;i<n;i++){
        x[i]+=dx;
        y[i]+=dy;
    }
    x0+=dx;
    y0+=dy;
}

void Figure::Scale(double S){
    for(int i=0;i<n;i++){
        x[i]=(x[i]-x0)*S+x0;
        y[i]=(y[i]-y0)*S+y0;
    }
}

void Figure::Spin(double A){
    double temp;
    for(int i=0;i<n;i++){
        temp=x[i];
        x[i]=x0+cos(A)*(x[i]-x0)-sin(A)*(y[i]-y0);
        y[i]=y0+sin(A)*(temp-x0)+cos(A)*(y[i]-y0);
    }
}

void save()
{
    int width, height;
    IMAGE *output;

    width  = getmaxx() + 1;
    height = getmaxy() + 1;
    output = createimage(width, height);

    getimage(0, 0, width - 1, height - 1, output);
    saveBMP("output.bmp", output);
    freeimage(output);
}
```

Файл control.h

```
#ifndef CONTROL_H
#define CONTROL_H

enum control_values { NONE = -1,UP,DOWN,RIGHT,LEFT,
   L_SPIN,R_SPIN,U_SCALE,D_SCALE,SAVE,EXIT,N_CONTROLS };

struct Control
{
   int left;
   int top;
   int right;
   int bottom;
};
void init_control(int, int, int, const char *);
void free_image();
void draw_image(int);
int select_control();

#endif
```

Файл control.cpp

```
#include "graphics.h"
#include "control.h"

Control controls[N_CONTROLS];
IMAGE *image[N_CONTROLS];

void init_control(int i, int left, int top,const char *file_name){
   image[i]=loadBMP(file_name);

   controls[i].left   = left;
   controls[i].top    = top;
   controls[i].right  = left + imagewidth(image[i]) - 1;
   controls[i].bottom = top  + imageheight(image[i]) - 1;

}

void free_image(){
   for(int i=0;i<N_CONTROLS;i++){
      freeimage(image[i]);
   }
}

void draw_image(int i){
   putimage(controls[i].left, controls[i].top, image[i], COPY_PUT);
}

int select_control(){
   int x, y;
```

```
    x = mousex();
    y = mousey();

    for (int i = 0; i < N_CONTROLS; i++)
    {
       if (x > controls[i].left && x < controls[i].right &&
           y > controls[i].top  && y < controls[i].bottom)
       {
          return i;
       }
    }
    return NONE;
}
```

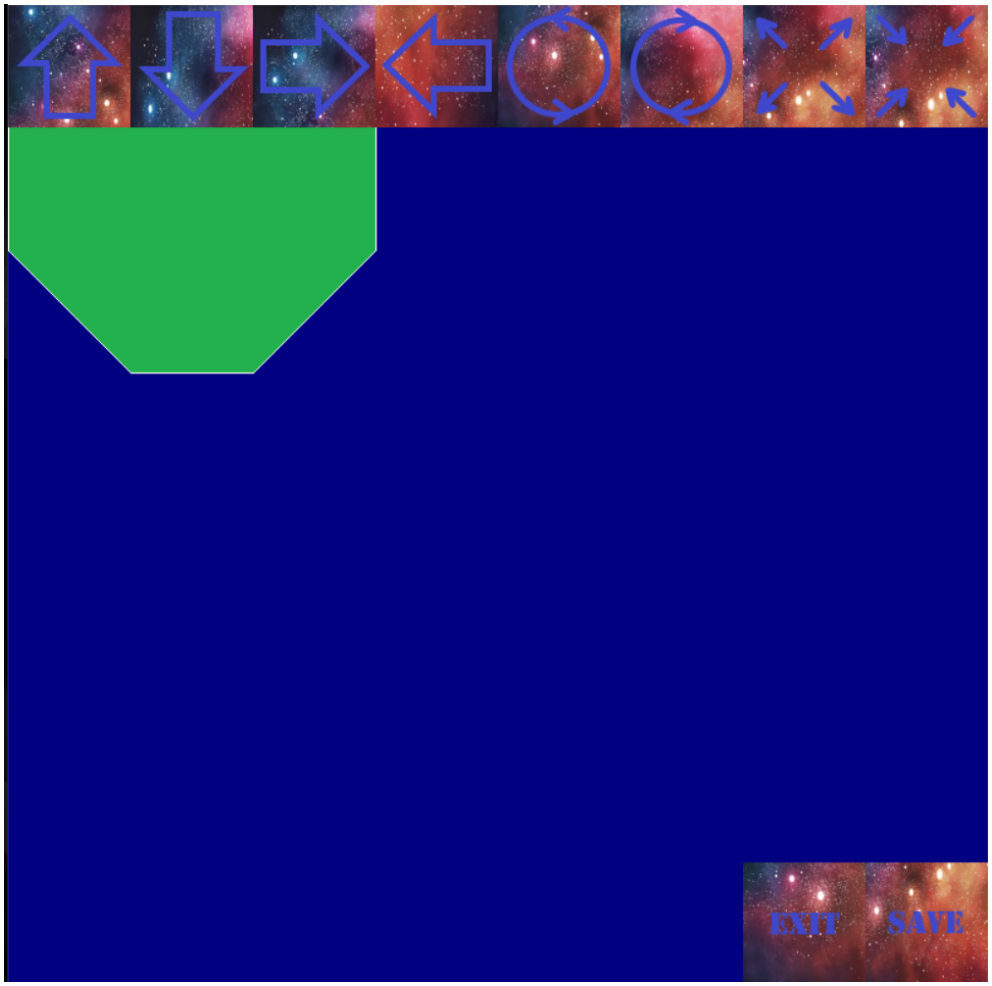

Рисунок 4.1 – Пример выполнения программы.

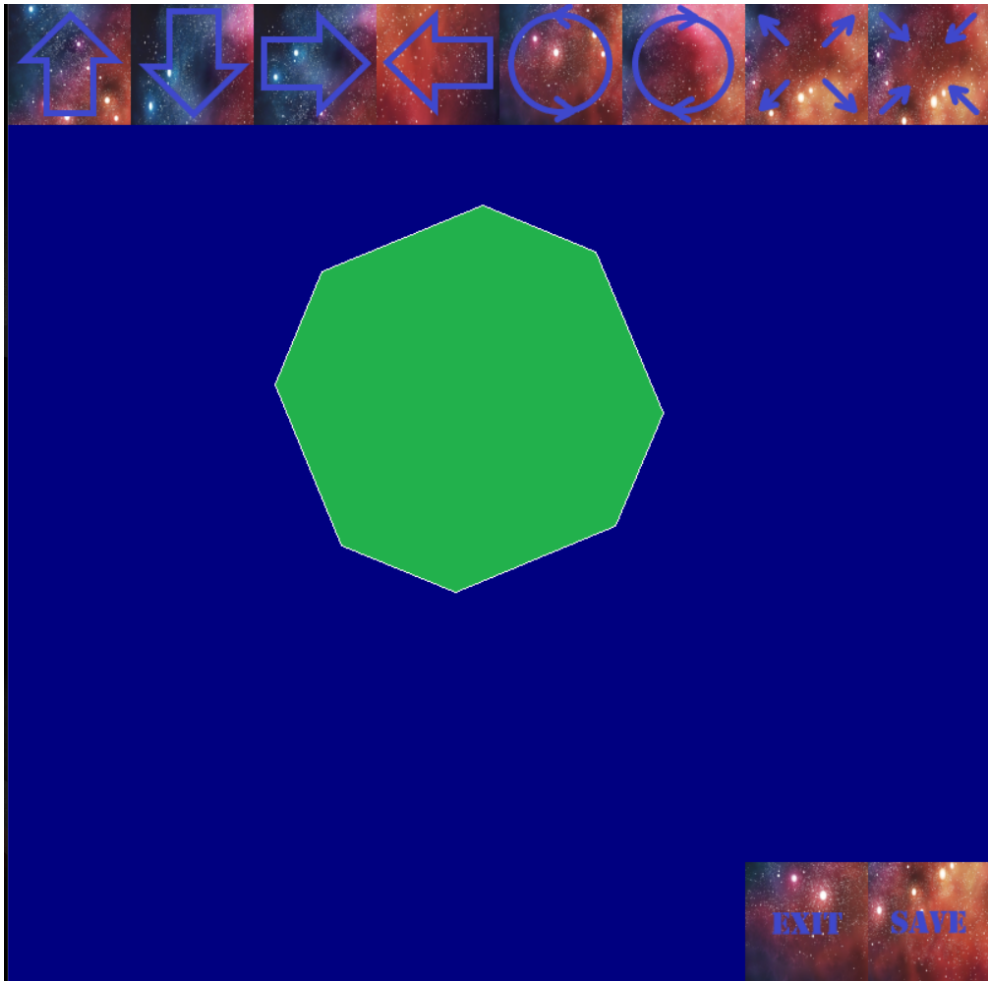

Рисунок 4.2 – Пример выполнения программы.