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
_____ Алиев Э.Ф.
«____» _____ 2022 г.

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

Челябинск 2022

1 ЗАДАНИЕ

1. 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;
- сохранение результата в файл;
- выход из программы.

структура Point в которой содержится x, y.

массив figure типа Point с углами фигуры:

-100 100

0 -100

100 -100

0 100

переменная center типа Point с координатами центра: 100, 100.

Методы:

void draw(Point, Point*) для отрисовки фигуры.

void scale(Point*, double) для изменения размера фигуры.

void spin(Point*, double) для вращения фигуры.

Файл main.cpp

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

using namespace std;

int main()
{
   initwindow(800, 600);

   IMAGE *image;

   image = loadBMP("fon.bmp");
   putimage(0, 0, image, COPY_PUT);

   create_control(LEFT, 0, 550, 100, 600);
   create_control(RIGHT, 100, 550, 200, 600);
   create_control(UP, 200, 550, 300, 600);
   create_control(DOWN, 300, 550, 400, 600);
   create_control(SPIN, 400, 550, 500, 600);
   create_control(DSPIN, 500, 550, 600, 600);
   create_control(UP_SCALE, 600, 550, 700, 600);
   create_control(DOWN_SCALE, 700, 550, 800, 600);
   create_control(SAVE, 650, 0, 750, 50);
   create_control(EXIT, 750, 0, 800, 50);

   Point *figure;
   figure = new Point[4];
   figure[0].x = -100;
   figure[0].y = 100;
   figure[1].x = 0;
   figure[1].y = -100;
   figure[2].x = 100;
   figure[2].y = -100;
   figure[3].x = 0;
   figure[3].y = 100;

   Point center = {100, 100};

   int p=0;
   while (true){
      p=1-p;
      setactivepage(p);
      putimage(0, 0, image, COPY_PUT);
      draw(center, figure);
      if(mousebuttons()){
```

```
            switch(select_control()){
               case NONE:
                  break;
               case UP:
                  center.y -= 10;
                  break;
               case DOWN:
                  center.y += 10;
                  break;
               case RIGHT:
                  center.x += 10;
                  break;
               case LEFT:
                  center.x -= 10;
                  break;
               case SPIN:
                  spin(figure, -M_PI/15);
                  break;
               case DSPIN:
                  spin(figure, M_PI/15);
                  break;
               case UP_SCALE:
                  scale(figure, 1.1);
                  break;
               case DOWN_SCALE:
                  scale(figure, 0.9);
                  break;
               case SAVE: save(); break;
               case EXIT: delete[]figure; closegraph(); return 0;
            }
         }

         setvisualpage(p);
         delay(50);
      }

      delete[]figure;
      closegraph();
   }
```

Файл task.h

```
#ifndef TASK_H
#define TASK_H

struct Point
{
   int x, y;
};

void draw(Point, Point*);
void scale(Point*, double);
void spin(Point*, double);
void save();
#endif
```

Файл task.cpp

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

void draw(Point center, Point *figure){
   setfillstyle(SOLID_FILL, COLOR(255, 255, 0));
   int *points;
   points = new int[8]();
   for(int i=0; i<4; i++){
      points[2*i] = figure[i].x+center.x;
      points[2*i+1] = figure[i].y+center.y;
   }
   fillpoly(4, points);
   delete [] points;
}

void scale(Point *figure, double s){
   for(int i=0; i<4; i++){
      figure[i].x *= s;
      figure[i].y *= s;
   }
}

void spin(Point *figure, double fi){
   double x;
   for(int i=0; i<4; i++){
      x = figure[i].x;
      figure[i].x = cos(fi)*figure[i].x-sin(fi)*figure[i].y;
      figure[i].y = sin(fi)*x+cos(fi)*figure[i].y;
   }
}
```

```
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,
   SPIN, DSPIN, UP_SCALE,DOWN_SCALE,SAVE,EXIT,N_CONTROLS };

struct Control
{
   int left;
   int top;
   int right;
   int bottom;
};
void create_control(int, int, int, int, int);
int select_control();

#endif
```

Файл control.cpp

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

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

void create_control(int i, int left, int top, int right, int bottom)
{
   controls[i].left   = left;
   controls[i].top    = top;
   controls[i].right  = right-1;
   controls[i].bottom = bottom-1;
}
```

```
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;
}
```

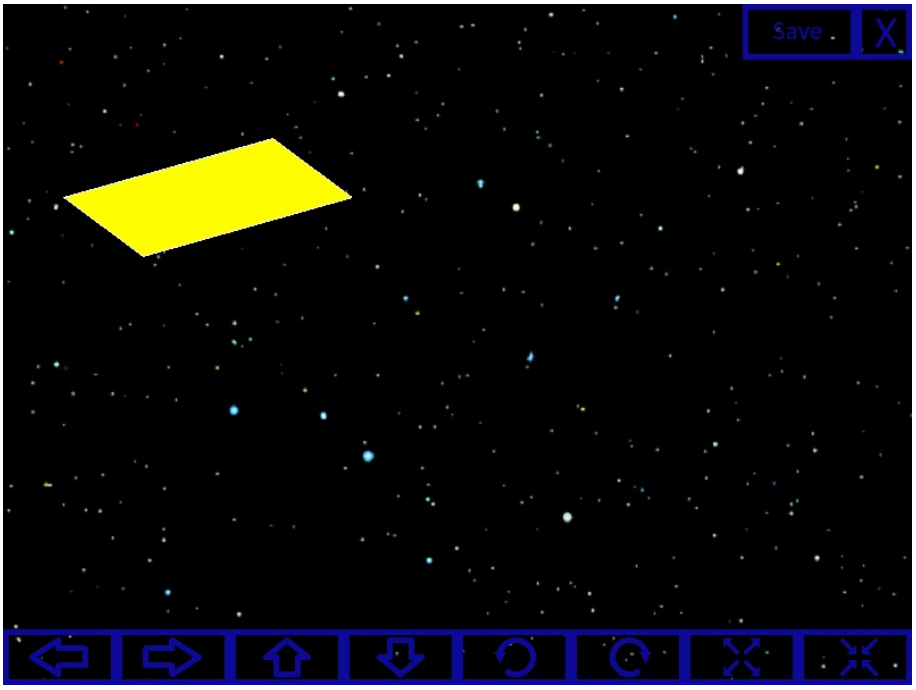

Рисунок 4.1 – Результат выполнения программы