МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Вычислительной техники

ОТЧЕТ

по лабораторной работе № 1

по дисциплине «Алгоритмы и структуры данных»

ТЕМА: РАБОТА С ИЕРАРХИЕЙ ОБЪЕКТОВ: НАСЛЕДОВАНИЕ И ПОЛИМОРФИЗМ.

Студенты гр. 0308 _____ Бобыльков Т.В.

_____ Радабольский В. С.

Преподаватель _____ Манирагена В.

Санкт-Петербург

2022

Цель работы.

Целью работы является исследование принципов объектно-ориентированного подхода в программировании, а именно полиморфизма и наследование, а также создание программы для рисования фигур согласно варианту.

Задание (вариант 33)

Дополнить рисунок фигурой «крест» (10) фигуру «физиономия» в местах «уши» (4, 5) и «эмблема на шляпе» (12).

Решение задания.

Для дополнения рисунка, согласно заданию, добавим в программу класс фигуры «крест», используя механизм наследования. Класс «cross» будет наследовать класс абстрактной фигуры «shape» в открытом виде (public), потому что фигура «крест» полностью является фигурой.

Из программы, представленной в методических указаниях не было удалено и\или переопределено никаких классов, потому что они подходили структуре нашей программы.

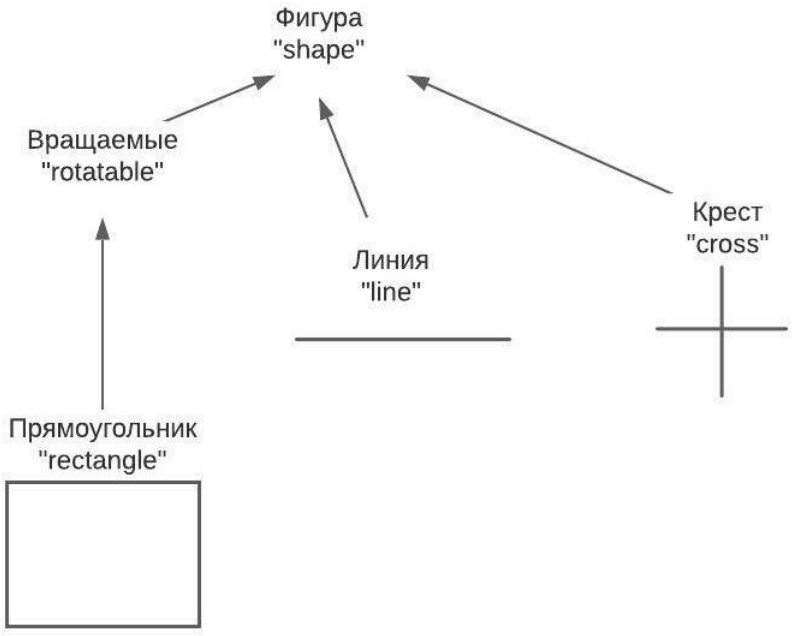
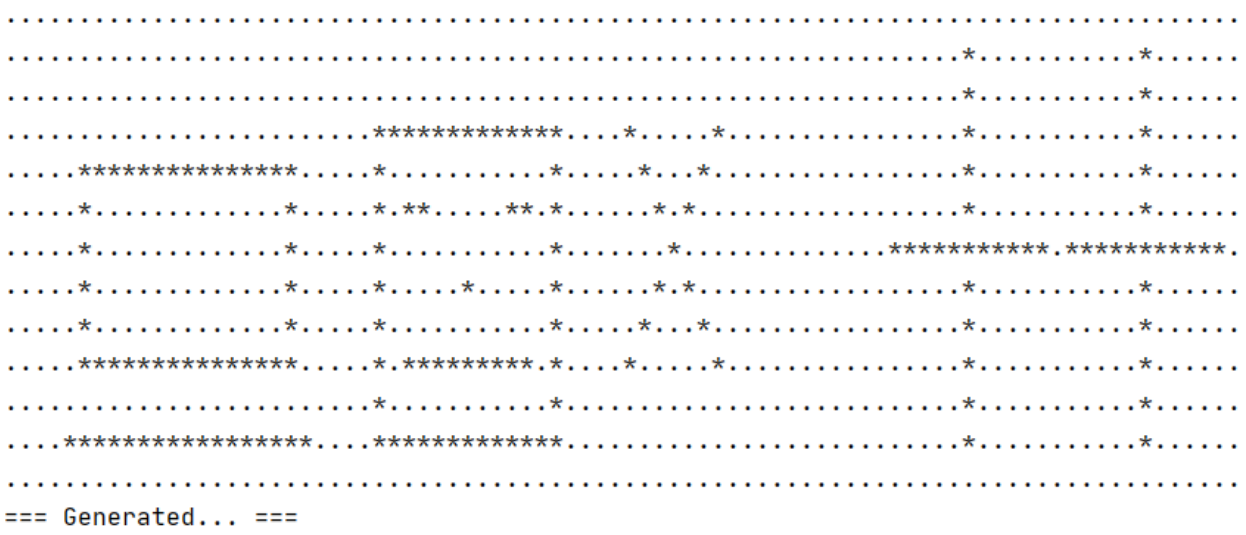


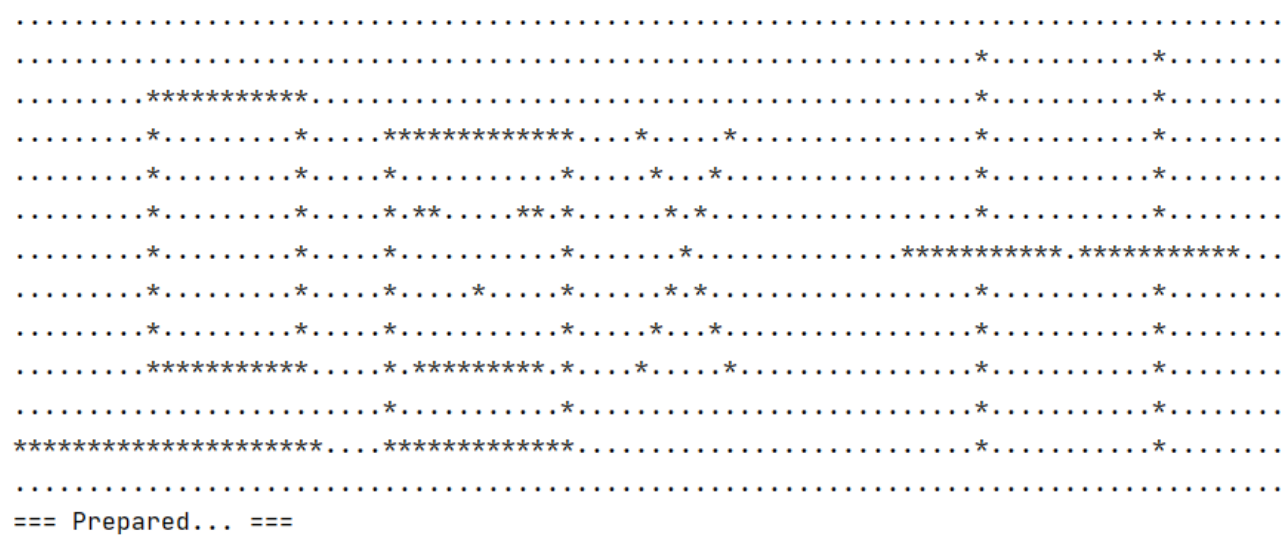
Рисунок 1. Иерархия объектов

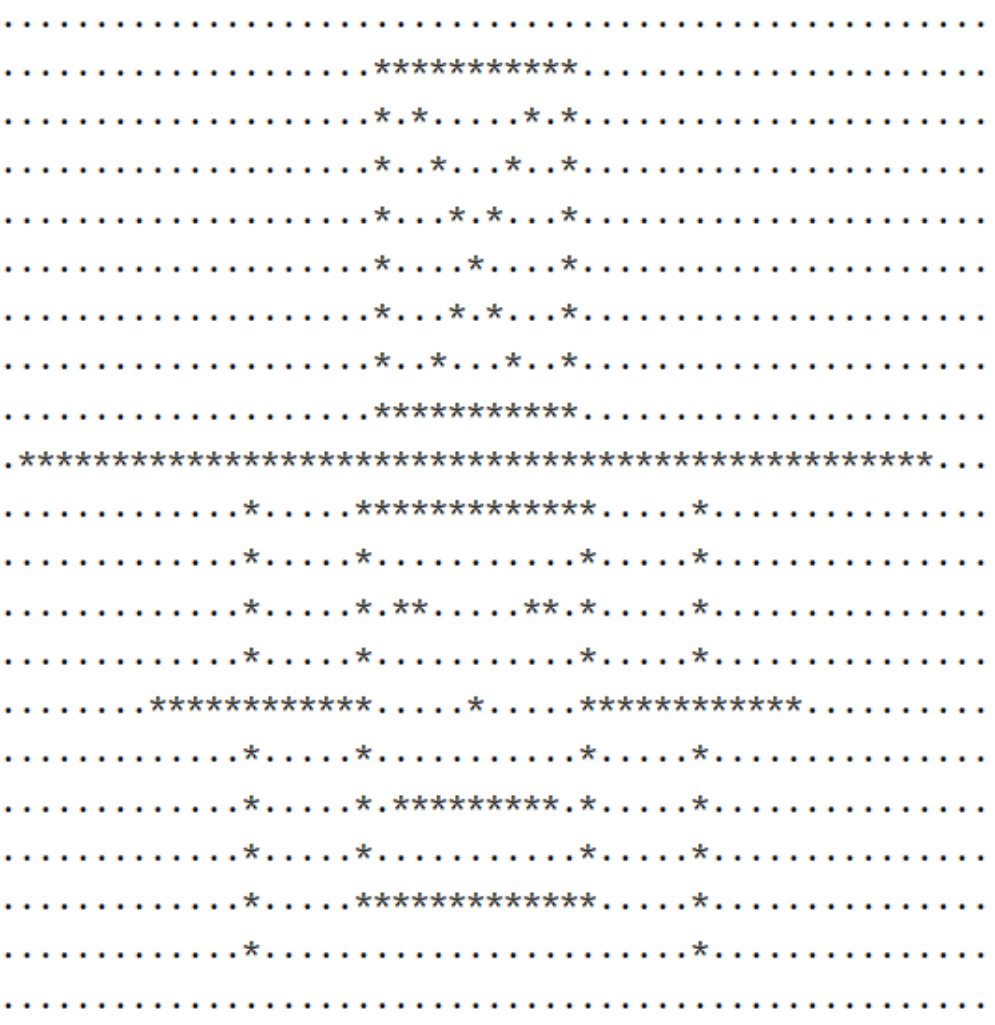
Для нового класса крест были переопределены такие функции как draw (рисование), resize (изменение размера) и move (перемещение), а также все функции, возвращающие позиционные точки, потому что класс было решено

воспринимать как центровую точку (point c_center) с фиксированным размером «лопастей» (int size).

Контрольный тест







Выводы.

В ходе выполнения работы мы научились пользоваться такими принципами ООП как полиморфизм и наследование, что помогло успешно модернизировать

код, указанный в методических указаниях. Были изучены динамические связывания для переопределения одноименных функций классов-наследников. Мы на собственном опыте смоделировали ситуацию, в которой объектно-ориентированный подход имеет высокое преимущество – дополнение уже существующих программ. Прямо на защите нам удалось за 10 строк кода создать новый класс андреевского креста, который наследует класс обычного креста.

Список использованных источников.

П. Г. Колинько, Алгоритмы и структуры данных. Лекция от 07.02.2022.

Методическое пособие «Пользовательские Контейнеры» П.Г. Колинько.

Приложение.

Исходный код программы для решения задачи на языке C++:

```
#include <iostream>
#include "screen.h"
#include "shape.h"


class cross : public shape {
    point c_center;
    int size;
public:
    cross(point c, int s): c_center(c), size(s) {};
    point north() const {return point(c_center.x, c_center.y + size);}; //точки
для привязки
    point south() const {return point(c_center.x, c_center.y - size);};
    point west() const {return point(c_center.x - size, c_center.y);};
    point east() const {return point(c_center.x + size, c_center.y);};

    point neast() const {return point(c_center.x + size, c_center.y + size);};
    point seast() const {return point(c_center.x + size, c_center.y - size);};
    point nwest() const {return point(c_center.x - size, c_center.y + size);};
    point swest() const {return point(c_center.x - size, c_center.y - size);};

    point center() const {return c_center;};


    void move(int a, int b); // перемещение
    void resize(double d); // Изменение размера

    void draw(); // переопределим рисование
};

class andrew_cross : public cross {

public:
    andrew_cross(point c, int s) {
        c_center = c;
        size = s;
    };
    void draw();
```

```
};
void andrew_cross::draw() {
    put_line(neast(), swest());
    put_line(nwest(), seast());
};

void cross ::draw() {
    put_line(north(), south());
    put_line(east(), west());
}

void cross::resize(double d) {
    size *= d;
}

void cross::move(int a, int b) {
    c_center.x += a;
    c_center.y += b;
}


// в центр
void into(shape &p, const shape &q) {
    point p_c = p.center();
    point q_c = q.center();

    p.move(q_c.x - p_c.x, q_c.y - p_c.y);
}
// слева
void left(shape &p, const shape &q) {
    point e = q.west();
    point w = p.east();
    p.move(e.x - w.x - 1, e.y - w.y);
}
// справа
void right(shape &p, const shape &q) {
    point w = q.east();
    point e = p.west();
    p.move(w.x - e.x + 1, w.y - e.y);
}

class myshape : public rectangle {
    int w, h;
    line l_eye; // Левый глаз
    line r_eye; // Правый глаз
    line mouth; // рот
public:
    myshape(point, point);
    void draw() ;
    void move(int, int);
    void resize(double) {}
};

myshape ::myshape(point a, point b) :
    rectangle(a, b),
    w(neast().x - swest().x + 1),
    h(neast().y - swest().y + 1),
    l_eye(point(swest().x +2, swest().y + h * 3/4), 2),
    r_eye(point(swest().x + w - 4, swest().y + h * 3/4), 2),
    mouth(point(swest().x + 2, swest().y + h/4), w - 4)
{}

void myshape :: draw() {
```

```
    rectangle :: draw(); // Контур лица

    point nose = center();

    put_point(nose);
}

void myshape :: move(int a, int b) {
    rectangle :: move(a, b);
    l_eye.move(a, b);
    r_eye.move(a, b);
    mouth.move(a, b);
}

int main() {

    setlocale(LC_ALL, "Rus");

    screen_init();

    // ==1. Объявление набор фигур ==
    rectangle hat(point(5,3), point(5+14,3+5));
    line brim(point(4,1), 17); // козырек
    myshape face(point(25,1), point(25+12,1+8));
    cross left_cross(point(65, 6), 5);
    cross right_cross(point(77, 6), 5);
    andrew_cross hat_cross(point(45, 6), 3);

    shape_refresh( );

    std::cout << "=== Generated... ===\n";
    std::cin.get(); //Смотреть исходный набор

    //== 2.Подготовка к сборке ==
    hat.rotate_right( );


    brim.resize(2.0);
    face.resize(2.0);

    shape_refresh( );
    std::cout << "=== Prepared... ===\n";
    std::cin.get(); //Смотреть результат поворотов/отражений
//== 3.Сборка изображения ==

    face.move(10, 10); // Лицо - в исходное положение
    up(brim, face);
    up(hat, brim);
    into(hat_cross, hat);
    left(left_cross, face);
    right(right_cross, face);
    shape_refresh( );
    std::cout << "=== Ready! ===\n";

    return 0;
}
```