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

# КУРСОВАЯ РАБОТА (КУРСОВОЙ ПРОЕКТ)

**по дисциплине «Программирование»**

**Тема: Программирование на Си 2 семестр.**

Студент гр. 1383 Федорова О.В.

Преподаватель Чайка К.В.

Санкт-Петербург

2022

**ЗАДАНИЕ**

**НА КУРСОВУЮ РАБОТУ**

Студент Федорова О.В.

Группа 1383

Тема работы : Программирование на Си, работа с изображениями.

## Вариант 5

Программа **должна** иметь CLI или GUI. Более подробно тут:
**http://se.moevm.info/doku.php/courses:programming:rules_extra_kurs**

**Общие сведения**

- 24 бита на цвет
- без сжатия
- файл всегда соответствует формату BMP (но стоит помнить, что версий у формата несколько)
- обратите внимание на выравнивание; мусорные данные, если их необходимо дописать в файл для выравнивания, должны быть нулями.
- обратите внимание на порядок записи пикселей
- все поля стандартных BMP заголовков в выходном файле должны иметь те же значения что и во входном (разумеется кроме тех, которые должны быть изменены).

Программа должна реализовывать весь следующий функционал по обработке bmp-файла

Задание

1. Инвертировать цвета в заданной окружности. Окружность определяется
    - **либо** координатами левого верхнего и правого нижнего угла квадрата, в который она вписана, **либо** координатами ее центра и радиусом
2. Обрезка изображения. Требуется обрезать изображение по заданной области. Область определяется:
    - Координатами левого верхнего угла
    - Координатами правого нижнего угла
3. Рисование треугольника. Треугольник определяется
    - Координатами его вершин
    - Толщиной линий
    - Цветом линий
    - Треугольник может быть залит или нет
    - цветом которым он залит, если пользователем выбран залитый
4. Рисование отрезка. Отрезок определяется:
    - координатами начала
    - координатами конца
    - цветом
    - толщиной

Дата сдачи реферата: 29.05.2022

Дата защиты реферата: 31.05.2022

| | | |
|---|---|---|
| Студент | | Федорова О.В. |
| Преподаватель | | Чайка К.В. |

**ВВЕДЕНИЕ**

Цель работы - выполнить задание. Получение аргументов программы с помощью командной строки. Обработать возможные ошибки от пользователя

# 1. Считывание

## 1.1. Считывание данных

Программу требуется реализовать в виде утилиты, подобной стандартным linux-утилитам

```
void printHelp(){
Здесь описан формат ввода данных для пользователя
}

int get_size(char* str) {
  Эта функция была написана для получения количества аргументов к каждой опции(строка, которую делим по запятым) для дальнейшей прореки на валидность данных и работы с ними
}

int* get_all(char* str) {
   Получаем массив с числами, которые поданы в качестве аргумента.
Например -o 1,1,20,20 обрежет изображение(тк -o), выделив прямоугольную часть изображения, координаты левого нижнего угла которой (1,1) и верхнего првого (20,20).
}
struct Configs{
    Описываю поля структуры в соответствии с заданием
```

```
};

int main(int argc, char* argv[]){
```

1)создаю флаги read и write, которые равны 0, если ранее не было записано ни одного файла на чтение и запись соответственно и больше нуля, если таковые файлы уже были получены

2)создаю основу для БМП файла и инициализирую поля структуры

3)Описываю структуру для расширенной версии опций После чего занимаюсь считыванием

```
    opt = getopt_long(argc, argv, opts, longOpts, &longIndex);
    while(opt!=-1){
      switch(opt){
           case 'f':
                получаю файл на запись
           case 'i':
                Файл на чтениеи обработка его, получение
```

массива пикселей

```
           case 'h':
           case '?':
                мелочи жизни
           case 0:
                printf("->%s\n",longOpts[longIndex].name);


      }
```

Запускаю подобное чтение заново, чтобы все данные мне опции были считаны и выполнены в процессе считывания

```
 opt = getopt_long(argc, argv, opts, longOpts, &longIndex);
    }
    optind = 0;
     opt = getopt_long(argc, argv, opts, longOpts,
&longIndex);
    while(opt!= -1){
      switch(opt){
           case 'c':
               Рисую окружности
                break;
           case 'o':
               Обрезаю
                break;
           case 'l':
               Рисую отрезки
                break;
           case 't':
               Рисую треугольники
                break;
      }
      opt = getopt_long(argc, argv, opts, longOpts,
&longIndex);
    }
```

После чего все записывается в файл для записи(если такой есть)

```
    if(write != 0 && read != 0) {
      ff = fopen(config.outfile, "wb");
      printBMP(ff, &bfh, &bih, arr);
      fclose(ff);
```

```
    }
    return 0;
}
```

**2. Обработка изображения**

Программа была разделена на несоклько файлов

В файле bmp.c и одноименном заголовочном описаны функции для чтения, записи bmp файла и получения массива пикселей из него.

**3. Работа с изображениями**

**В Файлах с соответвующими заданиям названиях описаны функции, менябщие массив пикселей так, чтобы было выполнено задание.**

**Рисование окружностей меняло массив таким образом, что точки, принадлежащие уравнению окружности инверттировались(достаточно вычесть из 255 каждый цвет, чтобы получить инверсию цвета)**

**Уравнение окружности здесь (x-x1)**2 +( y-y2)**2 <= R**2**

**Таким образом потеря пикселей, не сомтря на кривизну фигуры, минимальна, так как при работе с целыми координатами и таким уравнением, нам не придется ничего округлять**

**Для линий использовался принцип ближайших к прямой точек, в зависимости от толщины линии, искалось нужное количество строчек в каждой строке и столбце массива пикселей**

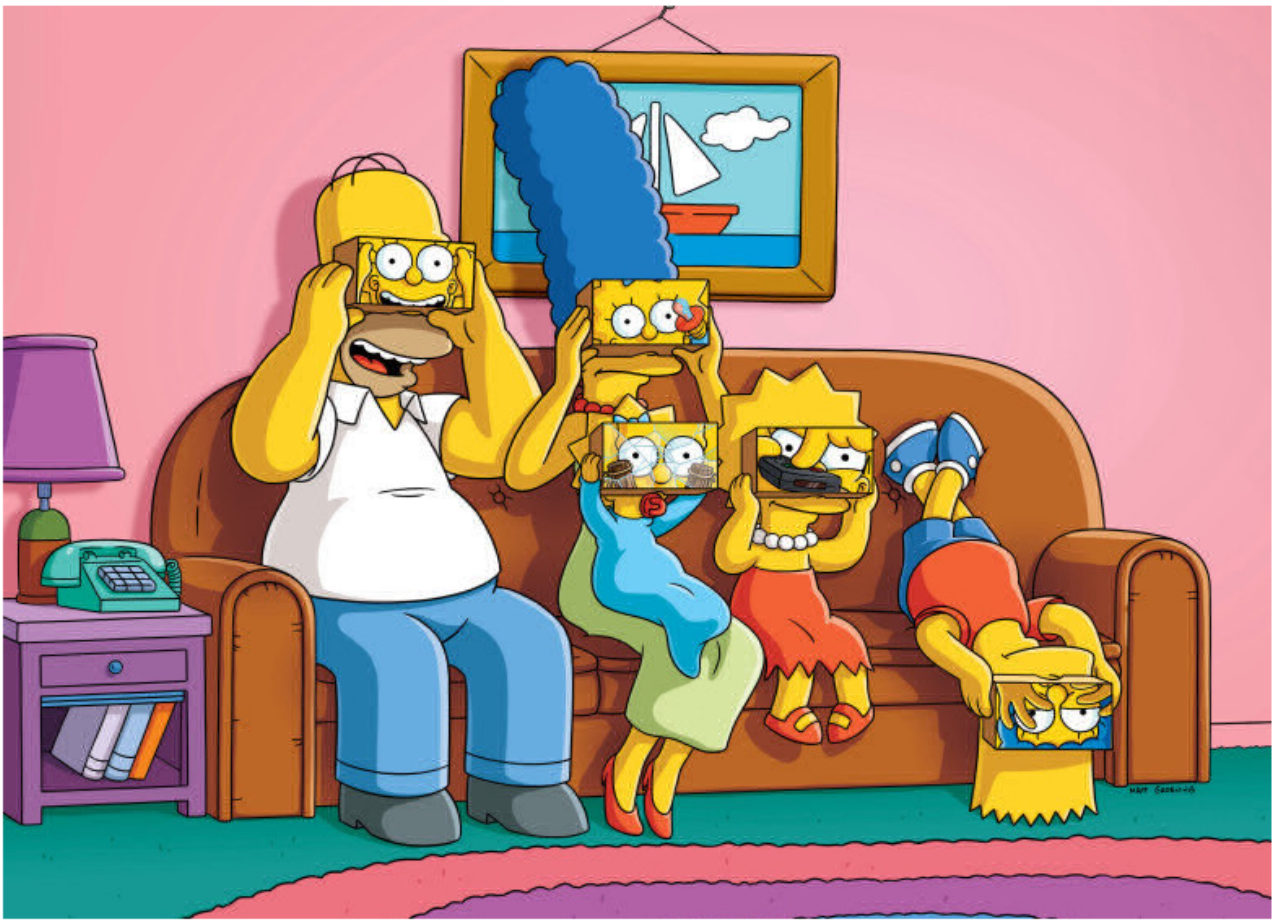
**Треугольники без заливки рисовались тремя отрезками, заливка же происходила по принципу проверки принадлежности точки треугольнику при помощи псевдоскалярного(косого) произведения векторов.**

**Обрезка изображения - главное поменять в структуре bih значения высоты и ширины иображения и перезаписать все в новый массив пикселей**

Результаты

Исходная картинка

t.bmp



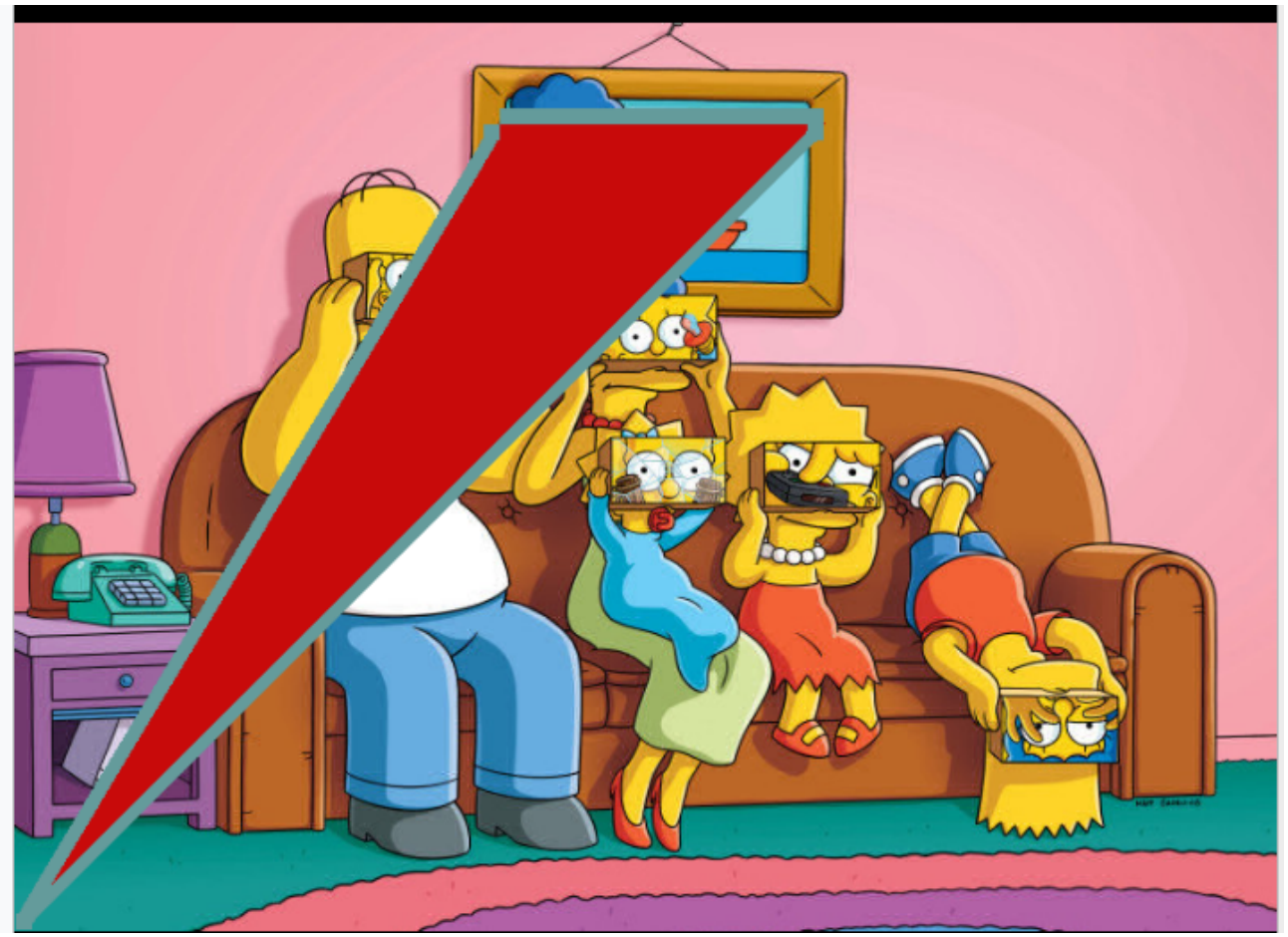
Рассмотрим, что выдает программа после некоторых ее вызовов

Ввод:

```
make && ./main  -i t.bmp -f out.bmp  -t
100,155,155,200,10,10,10,1,1,500,500,300,500
```

-цвет краев—--||-цвет заливк-||толщ|__координаты трез точек_|

Вывод



Ввод

```
make && ./main  -i t.bmp -f out.bmp -c 1000,1000,1000
-c 500,500,500 -c 250,250,250 -c 125,125,125
```

много раз инвертируем окружности(инверсия+инверсия = начальная версия)

Вывод

Ввод

```
make && ./main -c 100,100,100 -l 155,130,50,10,1,0,50,70
-t 155,133,255,10,16,16,100,100,50,80 -o 0,10,250,200
-i t.bmp -f o
ut.bmp
```

Рисуем окружность, прямую, незакрашенный треугольник и обрезаем и получаем

Вывод

И как выглядит –help