Описание полученного задания

Реализация контейнера, содержащего плоские геометрические фигуры, размещаемые в координатноей сетке.

Типы фигур:

- **Круг** - имеет целочисленные кооридната центра окружноти, радиус, цвет, функцию вычисления площади.
- **Прямоугольник** - имеет целочисленные координаты левого верхнего угла, правого нижнего угла, цвет, функцию вычисления площади.
- **Треугольник** - имеет целочисленные координаты трех вершин, цвет, функцию вычисления площади.

Дополнительная функция - сортировка контейнера алгоритмом Шелла.

Параметры ввода:

- **Случайная генерация фигур**: ./hw1 -n number `output_1_path.txt output_2_path.txt`, где параметр number отвечает за количество генерируемых фигур
- **Считывание параметров фигур из файла**: ./hw1 -f `input_path.txt` `output_1_path.txt` `output_2_path.txt`

Формат описания фигур в файле: считывание строк идет попарно, каждые 2 строки описывают одну фигуру. На первой строке подается индекс фигуры, на второй - ее параметры. Файл оканчивается символом '0', означающим конец описания фигур.

Параметры фигур:

- **Прямоугольник** - индекс фигуры = 1, параметры передаются строкой вида x_1 y_1 x_2 y_2 color_index, где x_1, y_1 - координаты левой верхней вершины.
- **Треугольник** - индекс фигуры = 2, параметры передаются строкой вида x_1 y_1 x_2 y_2 x_3 y_3 color_index, где x_i, y_i - координаты i-ой вершины вершины.
- **Круг** - индекс фигуры = 3, параметры передаются строкой вида x_1 y_1 *radius* color_index

Пример описания круга в файле:

```
3
-66 81 46 4
```

Параметр color_index должен принимать значение от 0 до 6, в противном случае цвет фигуры будет none

Структурная схема ВС с размещенной на ней программой

Таблица типов:

int	4 байта
double	8 байт
class Shape enum color int color_index __vfptr	16 байт 4 байта[0] 4 байта[4] 8 байт[8]
class Circle: Shape поля Shape int x1, y1, radius	28 байт 16 байт[0] 12 байт[16]
class Rectangle поля Shape int x_1, y_1, x_2, y_2	32 байта 16 байт[0] 16 байт[16, 20, 24, 28]
class Triangle поля Shape int $x_1, y_1, x_3, y_3, x_3, y_3$	40 байт 16 байт[0] 24 байта[16, 20, 24, 28, 32, 36]
struct container int len, size Shape storage	400008 байт 8 байт[0] 40 * 10000 = 400000 байт[8]

Память программы:

main()	
int argc char** argv container c int size	4 байта[0] 8 байт[4] 400008 байт[12] 4 байта[400020]
Container::Clear()	
int i	4 байта[0]
Container::Out()	
int i	4 байта[0]
Triangle::Area double a, b, c, p	 32 байта[0]
Container::ShellSort() int i, d, j	 12 байт[0]

stack:

main
Init Init - end
In/InRnd In/InRnd - end
Out Out - end
ShellSort ShellSort - end
Out
Out - end main - end

Тестовые прогоны

Файловый ввод:

10000 элементов: 2.063 сек.

5000 элементов: 0.520 сек.

1000 элементов: 0.03 сек.

500 элементов: 0.001 сек.

100 элементов: 0.001 сек.

На больших наборах данных программа выполненная в парадигме ООП выполняется быстрее, чем программы в функциональной парадигме.

Так же программа поддреживает обработку идеологически некорректных данных из файла(нулевой радиус, левый верхний угол ниже правого и тд)

Характеристики программы

Число исполняемых модулей: 6, включая main.cpp

Число заголовочных файлов: 6, включая rnd.h