**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 8**

Тема: Асинхронное программирование

Студент: Курносов М. Ю.

Группа: 80-208

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

1. Постановка задачи

Создать приложение, которое будет считывать из стандартного ввода данные фигур, согласно варианту задания, выводить их характеристики на экран и записывать в файл. Фигуры могут задаваться как своими вершинами, так и другими характеристиками (например, координата центра, количество точек и радиус).

Программа должна:

1. Осуществлять ввод из стандартного ввода данных фигур, согласно варианту задания.
2. Программа должна создавать классы, соответствующие введенным данных фигур.
3. Программа должна содержать внутренний буфер, в который помещаются фигуры. Для создания буфера допускается использовать стандартные контейнеры STL. Размер буфера задается параметром командной строки.
4. При накоплении буфера они должны запускаться на асинхронную обработку, после чего буфер должен очищаться.
5. Обработка должна производиться в отдельном потоке.
6. Реализовать два обработчика, которые должны обрабатывать данные буфера:
   1. Вывод информации о фигурах в буфере на экран.
   2. Вывод информации о фигурах в буфере в файл. Для каждого буфера должен создаваться файл с уникальным именем.
7. Оба обработчика должны обрабатывать каждый введенный буфер. Т.е. после каждого заполнения буфера его содержимое должно выводиться как на экран, так и в файл.
8. Обработчики должны быть реализованы в виде лямбда-функций и должны хранится в специальном массиве обработчиков. Откуда и должны последовательно вызываться в потоке – обработчике.
9. В программе должно быть ровно два потока (thread). Один основной (main) и второй для обработчиков;
10. В программе должен явно прослеживаться шаблон Publish-Subscribe. Каждый обработчик должен быть реализован как отдельный подписчик.
11. Реализовать в основном потоке (main) ожидание обработки буфера в потоке-обработчике. Т.е. после отправки буфера на обработку основной поток должен ждать, пока поток обработчик произведет вывод данных на экран и запишет в файл.

Вариант 25: треугольник, квадрат, прямоугольник.

1. Описание программы

Программа запускается с параметром, который является размером создаваемого ею буфера, где и будут хранится фигуры. Затем программа просит ввести имя файла, куда будут выписываться фигуры. Далее начинается процесс создания фигур : программа просит ввести тип фигуры(номер соотв.(1 – Треугольник 2 – Прямоугольник 3 – Квадрат)). После чего программа просит ввести координаты фигур. Важно: для квадрата программа проверяет координаты на предмет равенства всех сторон квадрата(Если не пройдёт проверку координаты фигуры обнуляются). После ввода координат фигура добавляется в буфер, программа спрашивает разрешение на продолжение работы(Для продолжения нужно ввести - 1 для выхода – 0, иначе программа покажет справку). При переполнении буфера или завершении работы программы фигуры выводятся в файл и на экран, а буфер очищается.

1. Набор тестов

Описание: в самом начале пользователь вводит имя файла , в который будут записываться фигуры при заполнении буфера.

Затем команды, которые будут предложены интерфейсом, так же в тестах видны координаты фигур, введённые через stdin.

**For file test1.txt:**

out1.txt 1 1 2 3 4 5 1 2 5 6 7 8 1 3 9 10 11 12 1 1 55 47 88 91 1 6 1 2 5 6 8 7 1 2 1 2 3 4 1 3 44 44 55 55 1 1 1 2 3 4 0

**For file test2.txt:**

out2.txt 1 1 2 3 4 5 1 2 5 6 7 8 1 3 9 10 11 12 1 1 55 47 88 91

**For file test3.txt:**

out1.txt 1 1 2 3 4 5 1 2 5 6 7 8 1 3 9 10 11 12 1 1 55 47 88 91 1 6 1 2 5 6 8 7 1 2 1 2 3 4 1 3 44 44 55 55 1 1 1 2 3 4 0

Первый тест вводит несколько фигур, делает ошибку в ведении команды, что поправляет программа, затем случается ошибка в выборе фигуры при вставке потом буфер заполняется, выводит значения и чистится, затем вводятся новые фигуры и завершается программа без заполнения буфера, что заставляет вывести незаписанные фигуры из буфера.

Второй тест вводит несколько фигур, делает ошибку в ведении команды, что поправляет программа, затем буфер заполняется, выводит значения и чистится, затем программа завершается.

Третий тест вводит несколько фигур, делает ошибку в ведении команды, что поправляет программа, затем случается ошибка в выборе фигуры при вставке потом буфер заполняется, выводит значения и чистится, затем вводятся новые фигуры и завершается программа без заполнения буфера, что заставляет вывести незаписанные фигуры из буфера.

1. Результаты выполнения тестов

**Тест №1**:

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/8laboop# ./oop\_exercise\_08 6 < test\_01.txt

Введите имя файла для вывода буффера.

out1.txtВведите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Для выполнения операции введите соответствующую цифру:

1 - Продолжить ввод значений

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введённый номер не соответствует фигуре

Продолжить?

Для выполнения операции введите соответствующую цифру:

1 - Продолжить ввод значений

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 5 6

Вершина №2: 7 6

Вершина №3: 7 8

Вершина №4: 5 8

Фигура №3

Вершина №1: 9 10

Вершина №2: 11 10

Вершина №3: 11 12

Вершина №4: 9 12

Фигура №4

Вершина №1: 55 47

Вершина №2: 88 91

Вершина №3: 33 97

Фигура №5

Вершина №1: 5 6

Вершина №2: 8 6

Вершина №3: 8 7

Вершина №4: 5 7

Фигура №6

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Буффер заполнился, поэтому он выведен в файл и опустошён

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Выход...

Фигуры, которые остались в контейнере:

Фигура №1

Вершина №1: 44 44

Вершина №2: 55 44

Вершина №3: 55 55

Вершина №4: 44 55

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

**Тест №2**:

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/8laboop# ./oop\_exercise\_08 6 < test\_02.txt

Введите имя файла для вывода буффера.

out2.txtВведите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Для выполнения операции введите соответствующую цифру:

1 - Продолжить ввод значений

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Выход...

Фигуры, которые остались в контейнере:

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 5 6

Вершина №2: 7 6

Вершина №3: 7 8

Вершина №4: 5 8

Фигура №3

Вершина №1: 9 10

Вершина №2: 11 10

Вершина №3: 11 12

Вершина №4: 9 12

Фигура №4

Вершина №1: 55 47

Вершина №2: 88 91

Вершина №3: 33 97

**Тест №3:**

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/8laboop# ./oop\_exercise\_08 6 < test\_03.txt

Введите имя файла для вывода буффера.

out1.txtВведите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Для выполнения операции введите соответствующую цифру:

1 - Продолжить ввод значений

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введённый номер не соответствует фигуре

Продолжить?

Для выполнения операции введите соответствующую цифру:

1 - Продолжить ввод значений

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 5 6

Вершина №2: 7 6

Вершина №3: 7 8

Вершина №4: 5 8

Фигура №3

Вершина №1: 9 10

Вершина №2: 11 10

Вершина №3: 11 12

Вершина №4: 9 12

Фигура №4

Вершина №1: 55 47

Вершина №2: 88 91

Вершина №3: 33 97

Фигура №5

Вершина №1: 5 6

Вершина №2: 8 6

Вершина №3: 8 7

Вершина №4: 5 7

Фигура №6

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Буффер заполнился, поэтому он выведен в файл и опустошён

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты диагональных вершин фигуры

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Введите координаты фигуры в направлении против часовой стрелки

Элемент вставлен.

Выход...

Фигуры, которые остались в контейнере:

Фигура №1

Вершина №1: 44 44

Вершина №2: 55 44

Вершина №3: 55 55

Вершина №4: 44 55

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

1. Листинг программы

**For file CMakeLists.txt:**

all: oop\_exercise\_08

oop\_exercise\_08: main.o

g++ main.o -pthread -o oop\_exercise\_08

main.o: main.cpp

g++ -pthread -c main.cpp

clean:

rm -rf \*.o oop\_exercise\_08 out\*

**For file main.cpp:**

#include "classes.hpp"

#include <thread>

#include <functional>

typedef int TP;

void help();

class Myclass{

int bufsize = 0;

int place = 0;

int nowcount = 0;

char filename[256];

Factory<TP>\* vec;

std::function<void()> f[3] = {[&]() {

//printf("\t 4 - создание фигуры\n ");

int fig = 0;

TP xl, yl, xr, yr;

std::cout << "Введите номер добавляемой фигуры:\n1 - Треугольник\n2 - Прямоугольник\n3 - Квадрат" << endl;

cin >> fig;

switch (fig)

{

case 1:

{

std::cout << "Введите координаты фигуры в направлении против часовой стрелки\n"

<< endl;

std::cin >> xl >> yl >> xr >> yr;

Triangle<TP> a(xl, yl, xr, yr);

vec[place % bufsize] = a;

std::cout << "Элемент вставлен." << endl;

break;

}

case 2:

{

std::cout << "Введите координаты диагональных вершин фигуры\n"

<< endl;

std::cin >> xl >> yl >> xr >> yr;

Rectangle<TP> a(xl, yl, xr, yr);

vec[place % bufsize] = a;

std::cout << "Элемент вставлен." << endl;

break;

}

case 3:

{

std::cout << "Введите координаты диагональных вершин фигуры\n"

<< endl;

std::cin >> xl >> yl >> xr >> yr;

Square<TP> a(xl, yl, xr, yr);

vec[place % bufsize] = a;

std::cout << "Элемент вставлен." << endl;

break;

}

default:

printf("Введённый номер не соответствует фигуре\n Продолжить?\n");

help();

return;

}

++place;

++nowcount;

return;

},

[&]() { //вывод на экран

if (nowcount == 0)

{

printf("Фигур нет.\n");

return;

}

for (int i = 0; i < nowcount; i++)

{

printf("Фигура №%d\n", i + 1);

Factory<TP> figure = vec[i];

for (int i = 0; i < figure.count; i++)

{

printf("Вершина №%d: ", i + 1);

cout << figure.a[i].first << " " << figure.a[i].second << endl;

}

}

return;

},

[&]() { //вывод в файл

if (nowcount == 0)

{

printf("Фигур нет.\n");

return;

}

FILE \*fl = fopen(filename, "a+");

for (int i = 0; i < nowcount; i++)

{

fprintf(fl, "Фигура №%d\n", i + 1);

Factory<TP> figure = vec[i];

for (int i = 0; i < figure.count; i++)

{

fprintf(fl, "Вершина №%d: %d %d\n", i + 1, figure.a[i].first, figure.a[i].second);

}

}

fprintf(fl,"\n");

fclose(fl);

return;

}};

void Clear(){

delete [] this->vec;

nowcount = 0;

this->vec = new Factory<TP>[this->bufsize];

this->place = 0;

}

void Print(){

thread t(f[2]);

f[1]();

t.join();

this->Clear();

}

public:

Myclass(){

char file[] = "outMyclass.txt";

int n = 0;

while(file[n] != '\0'){

filename[n] = file[n];

++n;

}

bufsize = 1;

delete vec;

vec = new Factory<TP>;

}

Myclass(int s, char file[256]){

int n = 0;

while(file[n] != '\0'){

filename[n] = file[n];

++n;

}

bufsize = s;

delete vec;

vec = new Factory<TP>[s];

}

Myclass(Myclass &m){

int n = 0;

while(m.filename[n] != '\0'){

filename[n] = m.filename[n];

++n;

}

this->bufsize = m.bufsize;

if(this->bufsize > 1){

delete [] this->vec;

}else{

delete this->vec;

}

this->vec = m.vec;

}

~Myclass(){

printf("Фигуры, которые остались в контейнере:\n");

this->Print();

delete [] vec;

}

void Push(){

f[0]();

if(this->nowcount == bufsize){

this->Print();

printf("Буффер заполнился, поэтому он выведен в файл и опустошён\n");

}

}

};

void help(){

printf("Для выполнения операции введите соответствующую цифру:\n ");

printf("\t 1 - Продолжить ввод значений\n ");

printf("\t 0 - Выход\n ");

}

int main(int argc, char \*\*argv)

{

char enter = -1;

int i = 0;

if (argc < 2)

{

printf("Too few args\n");

return 1;

}

int bufsize = 0;

while (argv[1][i] != '\0')

{

bufsize += argv[1][i] - 48;

bufsize \*= 10;

++i;

}

bufsize /= 10;

char filename[256];

cout << "Введите имя файла для вывода буффера." << endl;

fflush(stdin);

cin >> filename;

cout <<filename;

//fgets(filename, sizeof(filename), stdin);

fflush(stdin);

Myclass mass(bufsize, filename);

///////////////////////////////////////////////////////////////////////////////main////////////////////////////////////////

while((isdigit(enter))||enter == -1){

mass.Push();

enter = '0';

cin>>enter;

while(enter != '1'){

if(enter == '0'){

printf("Выход...\n");

return 0;

}else{

if(enter != '1'){

help();

cin>>enter;

}

}

}

}

printf("Похоже вы ввели не цифру\n Выход из программы...\n");

return 0;

}

**For file classes.hpp:**

#include <iostream>

#include <cmath>

#include <vector>

#include <algorithm>

#include <fstream>

using namespace std;

template <class T>

struct Factory

{

public:

int classid = 0;

int count = 0;

pair<T, T> a[4];

Factory(){

for (int i = 0; i < this->count; i++)

{

this->a[i].first = 0;

this->a[i].second = 0;

}

}

Factory(const Factory<T> &s){

this->count = s.count;

this->classid = s.classid;

for (int i = 0; i < count; i++)

{

this->a[i].first = s.a[i].first;

this->a[i].second = s.a[i].second;

}

}

void virtual Print(){

printf("Factory\n");

for (int i = 0; i < this->count; i++)

{

printf("Вершина №%d: ", i+1);

cout<<this->a[i].first<<" "<<this->a[i].second<<endl;

}

}

~Factory(){}

};

template <class T>

struct Rectangle : public Factory<T>

{

Rectangle(){

this->count = 4;

this->classid = 2;

}

Rectangle(const Rectangle<T> &s){

this->count = s.count;

this->classid = s.classid;

for (int i = 0; i < this->count; i++)

{

this->a[i].first = s.a[i].first;

this->a[i].second = s.a[i].second;

}

}

Rectangle(T x1, T y1, T x2, T y2){ //диагональные вершины

this->classid = 2;

this->count = 4;

this->a[0].first = x1;

this->a[0].second = y1;

this->a[2].first = x2;

this->a[2].second = y2;

this->a[1].first = x2;

this->a[1].second = y1;

this->a[3].first = x1;

this->a[3].second = y2;

}

void virtual Print(){

printf("Прямоугольник: \n");

for (int i = 0; i < this->count; i++)

{

printf("Вершина №%d: ", i+1);

cout<<this->a[i].first<<" "<<this->a[i].second<<endl;

}

}

~Rectangle(){}

};

template <class T>

struct Triangle : public Factory<T>

{

Triangle(){

this->count = 3;

this->classid = 1;

}

Triangle(const Triangle<T> &s){

this->count = s.count;

this->classid = s.classid;

for (int i = 0; i < this->count; i++)

{

this->a[i].first = s.a[i].first;

this->a[i].second = s.a[i].second;

}

}

Triangle(T x, T y, T x1, T y1)

{

this->classid = 1;

double angle = 3.14 / 180.0 \* (360 / 3);

this->count = 3;

for (int i = 0; i < 3; i++)

{

this->a[i].first = x;

this->a[i].second = y;

T nextX = ((x1 - x) \* cos(angle) + (y1 - y) \* -sin(angle)) + x1;

T nextY = ((x1 - x) \* sin(angle) + (y1 - y) \* cos(angle)) + y1;

x = x1;

y = y1;

x1 = nextX;

y1 = nextY;

}

}

void Print(){

printf("Треугольник: \n");

for (int i = 0; i < this->count; i++)

{

printf("Вершина №%d: ", i+1);

cout<<this->a[i].first<<" "<<this->a[i].second<<endl;

}

}

~Triangle(){}

};

template <class T>

struct Square : public Rectangle<T>{

Square(){

this->count = 4;

this->classid = 3;

}

Square(T x1, T y1, T x2, T y2){ //диагональные вершины

this->classid = 3;

this->count = 4;

if(x2-x1 != y2-y1){

printf("Вершины квадрата введены неверно\n");

return;

}

this->a[0].first = x1;

this->a[0].second = y1;

this->a[2].first = x2;

this->a[2].second = y2;

this->a[1].first = x2;

this->a[1].second = y1;

this->a[3].first = x1;

this->a[3].second = y2;

}

Square(const Square<T> &s){

this->count = s.count;

this->classid = s.classid;

for (int i = 0; i < this->count; i++)

{

this->a[i].first = s.a[i].first;

this->a[i].second = s.a[i].second;

}

}

void Print(){

printf("Квадрат: \n");

for (int i = 0; i < this->count; i++)

{

printf("Вершина №%d: ", i+1);

cout<<this->a[i].first<<" "<<this->a[i].second<<endl;

}

}

};

1. Выводы

По мере выполнения данной лабораторной работы были закреплены навыки написания многопоточных программ, работы с безымянными функциями, указателями на функции. Ознакомился также с основными концептами мультипрограммирования.

Литература

1. Форум Medium [Электронный ресурс]. URL: https://medium.com/swlh/c-multithreading-and-concurrency-introduction-f640ce986fa7 (дата обращения: 20.12.20).

2.Бьярне Страуструп. Программирование: принципы и практика использования C++, исправленное издание = Programming: Principles and Practice Using C++. — М.: «Вильямс», 2011. — С. 1248. — ISBN 978-5-8459-1705-8.

3.Канал YouTube https://m.youtube.com/channel/UCtLKO1Cb2GVNrbU7Fi0pM0w