Московский Авиационный Институт
(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Лабораторная работа
по курсу «ООП»

Тема:
Многопоточность.

Студент:	Черненко И.Д
Группа:	М80-206Б-18
Преподаватель:	Журавлев А.А.
Вариант:	24
Оценка:	
Дата:	

Москва
2019

1. Код программы на языке C++:

main.cpp

```
#include <iostream>
#include <memory>
#include <vector>
#include <thread>
#include "factory.h"
#include "figure.h"
#include "subscriber.h"
int main(int argc,char* argv[]) {
    if (argc != 2) {
        std::cout << "not enough arguments\n";
        return 1;
    }
    int  buffer_size = std::stoi(argv[1]);
    std::shared_ptr<std::vector<std::shared_ptr<figure>>> buffer =
std::make_shared<std::vector<std::shared_ptr<figure>>>();
    buffer->reserve(buffer_size);
    factory factory;
    std::string cmd;
    subscriber sub;
    sub.processors.push_back(std::make_shared<stream_processor>());
    sub.processors.push_back(std::make_shared<file_processor>());
    std::thread sub_thread(std::ref(sub));

    while (true) {
        std::unique_lock<std::mutex> locker(sub.mtx);
        std::cin >> cmd;
        if (cmd == "+") {
            try {
                buffer->push_back(factory.new_figure(std::cin));
            } catch (std::logic_error &e) {
                std::cout << e.what() << '\n';
                continue;
            }
            if (buffer->size() == buffer_size) {
                std::cout << "Buffer!\n";
                sub.buffer = buffer;
                sub.cond_var.notify_all();
                sub.cond_var.wait(locker, [&](){ return sub.buffer == nullptr;});
                buffer->clear();
            }
        } else if (cmd == "ext")  {
            break;
```

```
        } else {
            std::cout << "no such a command\n";
        }
    }
    sub.stop = true;
    sub.cond_var.notify_all();
    sub_thread.join();
    return 0;
}
```

2. Ссылка на репозиторий на GitHub

https://github.com/IllCher/oop_exercise_08

3. Набор testcases.

```
test_00:
+ o 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
+ q 0 0 -1 -1
+ t 0 0 0 1 1 0
ext

test_01:
+ o 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
+ q 0 0 -1 -1
+ t 0 0 0 1 1 0
+ q -1 -1 2 2
ext
```

4. Результаты выполнения тестов.

```
test_00:
Buffer!
0 0
0 0
0 0
0 0
0 0
0 0
0 0
0 0

0 0
0 -1
-1 -1
-1 0
```

0 0
0 1
1 0

test_01:
Buffer!
0 0
0 0
0 0
0 0
0 0
0 0
0 0
0 0

Buffer!
0 0
0 -1
-1 -1
-1 0

Buffer!
0 0
0 1
1 0

Buffer!
-1 -1
-1 2
2 2
2 -1

4. Объяснение результатов работы программы.

1) При запуске программы вводится одна из 2 возможных команд в виде строки.
2) + – пользователь вводит сначала тип фигуры, потом её координаты.
3) ext – корректный выход их программы.

5. Вывод.

Выполняя данную лабораторную, я получил опыт работы с потоками в C++, технологией continuous integration.