Московский Авиационный Институт
(Национальный исследовательский Университет)

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

Лабораторная работа
по курсу «ООП»

Тема:
Асинхронное программирование.

Студент:	Суворова С. А.
Группа:	М80-206Б-18
Преподаватель:	Журавлев А.А.
Вариант:	22
Оценка:	
Дата:	

Москва
2019

1.**Код на C++:**

point.h:
```
#ifndef D_POINT_H_
#define D_POINT_H_

#include <iostream>

struct point{
    double x,y;
};

std::istream& operator>>(std::istream& is, point& p);
std::ostream& operator<<(std::ostream& os,const point& p);
point operator+(point x1,point x2);
point& operator/= (point& x1, int number);

#endif
```

point.cpp:
```
#include <iostream>

#include "point.h"

std::istream& operator>> (std::istream& is, point& p){
    is >> p.x >>p.y;
    return is;
}

std::ostream& operator<< (std::ostream& os, const point& p){
    os << p.x << " " << p.y;
    return os;
}

point operator+(point x1,point x2){
    point x3;
    x3.x=x1.x+x2.x;
    x3.y=x1.y+x2.y;
    return x3;
}
point& operator/= (point& x1, int number){
    x1.x=x1.x/number;
    x1.y=x1.y/number;
    return x1;
}
```

figure.h:
```
#ifndef D_FIGURE_H_
#define D_FIGURE_H_

#include <iostream>

#include "point.h"

struct figure{
    virtual point center() const = 0;
    virtual void print(std::ostream &os) const = 0;
    virtual void help_print(std::ostream &os) const = 0;
    virtual double square() const = 0;
    virtual ~figure() {};
};
```

```
#endif
```

five_angles.h:
```
#ifndef D_FIVE_ANGLES_H_
#define D_FIVE_ANGLES_H_

#include <iostream>
#include "figure.h"

struct five_angles : figure{

    five_angles(std::istream &is);

    point center() const override;
    void print(std::ostream &os) const override;
    void help_print(std::ostream &os) const override;
    double square() const override;

private:
point one,two,three,four,five;

};

#endif
```

five_angles.cpp:
```
#include <iostream>

#include "five_angles.h"

five_angles::five_angles(std::istream &is){
    is >> one >> two >> three >> four >> five;
}

point five_angles::center() const {
    point p;
    p=one+two+three+four+five;
    p/=5;
    return p;
}

void five_angles::print(std::ostream &os) const {
    os << one << " " << two << " " << three << " " << four << " " << five
<<"\n";
}
void five_angles::help_print(std::ostream &os) const {
    os << "five_angles " << one << " " << two << " " << three << " " << four
<< " " << five <<"\n";
}

double five_angles::square() const {
    double s=0;
    s=(one.x*two.y+two.x*three.y+three.x*four.y+four.x*five.y+five.x*one.y-
two.x*one.y-
            three.x*two.y-four.x*three.y-five.x*four.y-one.x*five.y)/2;
    if(s<0){
        return -s;
    }else {
        return s;
    }
}
```

six_angles.h:
```
#ifndef D_SIX_ANGLES_H_
#define D_SIX_ANGLES_H_

#include <iostream>
#include "figure.h"


struct six_angles : figure{

    six_angles(std::istream &is);

    point center() const override;
    void print(std::ostream &os) const override;
    void help_print(std::ostream &os) const override;
    double square() const override;
private:
    point one,two,three,four,five,six;
};

#endif
```

six_angles.cpp:
```
#include <iostream>

#include "six_angles.h"

six_angles::six_angles(std::istream &is){
    is >> one >> two >> three >> four >> five >>six;
}

point six_angles::center() const {
    point p;
    p=one+two+three+four+five+six;
    p/=6;
    return p;
}

void six_angles::print(std::ostream &os) const {
    os << one << " " << two << " " << three << " " << four << " " << five <<
" " << six <<"\n";
}
void six_angles::help_print(std::ostream &os) const {
    os <<"six_angles " << one << " " << two << " " << three << " " << four <<
" " << five << " " << six <<"\n";
}

double six_angles::square() const {
    double s=0;

s=(one.x*two.y+two.x*three.y+three.x*four.y+four.x*five.y+five.x*six.y+six.x*
one.y-two.x*one.y-
       three.x*two.y-four.x*three.y-five.x*four.y-six.x*five.y-
one.x*six.y)/2;
    if(s<0){
        return -s;
    }else {
        return s;
    }
}
```

eight_angles.h:
```
#ifndef D_EIGHT_ANGLES_H_
#define D_EIGHT_ANGLES_H_

#include <iostream>
#include "figure.h"

struct eight_angles : figure{

    eight_angles(std::istream &is);

    point center() const override;
    void print(std::ostream &os) const override;
    void help_print(std::ostream &os) const override;
    double square() const override;
private:
    point one,two,three,four,five,six,seven,eight;
};

#endif
```

eight_angles.cpp:
```
#include <iostream>

#include "eight_angles.h"

eight_angles::eight_angles(std::istream &is){
    is >> one >> two >> three >> four >> five >>six >>seven >>eight;
}

point eight_angles::center() const {
    point p;
    p=one+two+three+four+five+six+seven+eight;
    p/=8;
    return p;
}

void eight_angles::print(std::ostream &os) const {
    os << one << " " << two << " " << three << " " << four << " " << five <<
" " << six << " " << seven
    << " " << eight<<"\n";
}
void eight_angles::help_print(std::ostream &os) const {
    os <<"eight_angles " << one << " " << two << " " << three << " " << four
<< " " << five << " " << six << " " << seven
       << " " << eight<<"\n";
}

double eight_angles::square() const {
    double s=0;

s=(one.x*two.y+two.x*three.y+three.x*four.y+four.x*five.y+five.x*six.y+six.x*
seven.y+seven.x*eight.y+
            eight.x*one.y-two.x*one.y-three.x*two.y-four.x*three.y-
five.x*four.y-six.x*five.y-seven.x*six.y
            -eight.x*seven.y-one.x*eight.y)/2;
    if(s<0){
        return -s;
    }else {
        return s;
    }
}
```

processor.h:

```
#ifndef D_PROCESSOR_H_
#define D_PROCESSOR_H_

#include <iostream>
#include <memory>
#include <vector>
#include <fstream>


struct processor {
    virtual void process(const std::vector<std::unique_ptr<figure>> &data)
const = 0;

};

struct screen_writer : processor {
    void process(const std::vector<std::unique_ptr<figure>> &data) const {
        for (size_t i=0;i<data.size();++i) {
            data[i]->print(std::cout);
        }
    }
};

struct file_writer : processor {
    void process(const std::vector<std::unique_ptr<figure>> &data) const {
        std::ofstream os(std::to_string(file_id)+".txt");
        file_id += 1;
        for (size_t i=0;i<data.size();++i) {
            data[i]->help_print(os);
        }
    }
    mutable int file_id = 0;
};

#endif
```

executor.h:

```
#ifndef D_EXECUTOR_H_
#define D_EXECUTOR_H_

#include <iostream>
#include <vector>
#include <memory>
#include <thread>
#include <mutex>
#include <condition_variable>
#include "figure.h"
#include "processor.h"

inline std::mutex mut;
inline std::condition_variable con;

struct exec{

    exec(){
        screen_writer pp1;
        file_writer pp2;
        std::unique_ptr<screen_writer>
p1=std::make_unique<screen_writer>(pp1);
        std::unique_ptr<file_writer> p2=std::make_unique<file_writer>(pp2);
        processors.push_back(std::move(p1));
        processors.push_back(std::move(p2));
    }

    void exec_set(std::vector<std::unique_ptr<figure>> data);

    static void worker(exec* one);

    std::thread work{worker,this};

    bool working=false;
    bool doing=false;

private:
    std::vector<std::unique_ptr<figure>> data;
    std::vector<std::unique_ptr<processor>> processors;
};

#endif
```

executor.cpp:

```
#include <iostream>

#include "executor.h"

void exec::worker(exec *one){
  while(!one->working) {
    std::unique_lock<std::mutex> lock(mut);
    con.wait(lock);
    if(one->doing) {
      std::cout << "\n";
      for (size_t i = 0; i < one->processors.size(); ++i) {
        one->processors[i]->process(one->data);
      }
      con.notify_one();
    }
  }
}

void exec::exec_set(std::vector<std::unique_ptr<figure>> data1){
  data=std::move(data1);
}
```

main.cpp:

```
#include <iostream>
#include "five_angles.h"
#include "six_angles.h"
#include "eight_angles.h"
#include "processor.h"
#include <vector>
#include <string>
#include <thread>
#include <fstream>
#include <memory>
#include <mutex>
```

```
#include <condition_variable>
#include "executor.h"


int main(int argc, char *argv[]) {
    if (argc != 2) {
        std::cerr << "ERROR" << "\n";
        return 1;
    }
    size_t buffer_size = (size_t)std::stoi(argv[1]);
    std::vector<std::unique_ptr<figure>> buffer;
    buffer.reserve(buffer_size);
    exec make;
    std::unique_ptr<figure> value;
    std::string figures;
    while (std::cin >> figures) {
        std::unique_lock<std::mutex> lock(mut);
        if(figures == "five_angles"){
            std::unique_ptr<figure> new_figure;
            new_figure=std::make_unique<five_angles>(five_angles(std::cin));
            buffer.push_back(std::move(new_figure));
        }else if(figures == "six_angles"){
            std::unique_ptr<figure> new_figure;
            new_figure=std::make_unique<six_angles>( six_angles(std::cin));
            buffer.push_back(std::move(new_figure));
        }else if(figures == "eight_angles"){
            std::unique_ptr<figure> new_figure;
            new_figure=std::make_unique<eight_angles>(
eight_angles(std::cin));
            buffer.push_back(std::move(new_figure));
        }
        make.doing= true;
        if (buffer.size() != buffer_size) {
            continue;
        }else {
            make.exec_set(std::move(buffer));
            con.notify_one();
            buffer.clear();

        }
    }
    make.doing= false;
    make.working = true;
    con.notify_one();
    make.work.join();
    return 0;
}
```

2. **Ссылка на репозиторий в GitHub:**
https://github.com/Suvorova-Sofya/oop_exercise_08

3.**Набор testcases:**
test1:

test2:
```
five_angles 1 1 2 2 3 3 4 4 5 5
five_angles 1 1 2 2 3 3 4 4 5 5
five_angles 1 1 2 2 3 3 4 4 5 5
five_angles 1 1 2 2 3 3 4 4 5 5
five_angles 1 1 2 2 3 3 4 4 5 5
```

test3:
```
five_angles 1 1 2 2 3 3 4 4 5 5
six_angles 1 1 2 2 3 3 4 4 5 5 6 6
eight_angles 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
five_angles 1 1 2 2 3 3 4 4 5 5
six_angles 1 1 2 2 3 3 4 4 5 5 6 6
eight_angles 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
five_angles 1 1 2 2 3 3 4 4 5 5
six_angles 1 1 2 2 3 3 4 4 5 5 6 6
eight_angles 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
five_angles 1 1 2 2 3 3 4 4 5 5

five_angles 1 1 2 2 3 3 4 4 5 5
six_angles 1 1 2 2 3 3 4 4 5 5 6 6
eight_angles 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
five_angles 1 1 2 2 3 3 4 4 5 5
six_angles 1 1 2 2 3 3 4 4 5 5 6 6
eight_angles 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
five_angles 1 1 2 2 3 3 4 4 5 5
six_angles 1 1 2 2 3 3 4 4 5 5 6 6
eight_angles 1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
five_angles 1 1 2 2 3 3 4 4 5 5
```

4.**Результаты выполнения программы:**

test1:

test2:
```
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
```

```
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
......................................
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5
```

test3:

```
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
```

```
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
..............
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5 6 6
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5 6 6
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5 6 6
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
1 1 2 2 3 3 4 4 5 5
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
............................................................................
```

```
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
.............................................................................
..............................
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5 6 6
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5 6 6
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
1 1 2 2 3 3 4 4 5 5
1 1 2 2 3 3 4 4 5 5 6 6
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8
1 1 2 2 3 3 4 4 5 5
```

5**. Объяснение результатов работы программы:**
Пользователь вводит фигуры и их координаты. Когда буфер будет заполнен данными фигурами, запустится обработчик, выполняющийся в отдельном потоке, который выведет всё содержимое буфера на экран и в файл.
6.**Вывод:**
В данной программе показывается ,каким образом можно производить асинхронную обработку данных, используя потоки, для уменьшения времени работы программы.