Московский авиационный институт
(Национальный исследовательский университет)

Факультет: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»
Дисциплина: «Объектно-ориентированное программирование»

Лабораторная работа № 2

Студент: Бронников М. А.
Группа: 80-204
Преподаватель: Чернышов Л.Н.
Дата: 15.10.2018
Оценка:

Москва, 2018

1. Постановка задачи

Вариант 1 (треугольник, массив)

Цель работы

Целью лабораторной работы является:

- Закрепление навыков работы с классами.
- Создание простых динамических структур данных.
- Работа с объектами, передаваемыми «по значению».

Задание

Необходимо спроектировать и запрограммировать на языке C++ класс-контейнер первого уровня, содержащий **одну фигуру (колонка фигура 1),** согласно вариантов задания (реализованную в ЛР1).
Классы должны удовлетворять следующим правилам:

- Требования к классу фигуры аналогичны требованиям из лабораторной работы 1.
- Классы фигур должны иметь переопределенный оператор вывода в поток std::ostream (<<). Оператор должен распечатывать параметры фигуры (тип фигуры, длины сторон, радиус и т.д).
- Классы фигур должны иметь переопределенный оператор ввода фигуры из потока std::istream (>>). Оператор должен вводить основные параметры фигуры (длины сторон, радиус и т.д).
- Классы фигур должны иметь операторы копирования (=).
- Классы фигур должны иметь операторы сравнения с такими же фигурами (==).
- Класс-контейнер должен содержать объекты фигур “по значению” (не по ссылке).
- Класс-контейнер должен иметь метод по добавлению фигуры в контейнер.
- Класс-контейнер должен иметь методы по получению фигуры из контейнера (определяется структурой контейнера).
- Класс-контейнер должен иметь метод по удалению фигуры из контейнера (определяется структурой контейнера).

- Класс-контейнер должен иметь перегруженный оператор по выводу контейнера в поток std::ostream (<<).
- Класс-контейнер должен иметь деструктор, удаляющий все элементы контейнера.
- Классы должны быть расположены в раздельных файлах: отдельно заголовки (.h), отдельно описание методов (.cpp).

Нельзя использовать:

- Стандартные контейнеры std.
- Шаблоны (template).
- Различные варианты умных указателей (shared_ptr, weak_ptr).

Программа должна позволять:

- Вводить произвольное количество фигур и добавлять их в контейнер.
- Распечатывать содержимое контейнера.
- Удалять фигуры из контейнера.

2. Решения задачи

Программа позволяет вводить произвольное количество фигур и добавлять их в контейнер учитывая присваиваемый пользователем индекс, распечатывать содержимое контейнера и удалять фигуры из контейнера.

ТЕСТЫ ПРОГРАММЫ:

```
max@max-X550CC:~/oop2$ ./hello
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
massive created!
massive created!
Triangle created: 0, 0, 0
Triangle created: 0, 0, 0
```

```
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  massive copied!
  Triangle created: 1, 1, 1
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:4
  Enter index:7
  Enter triangle:
  5 4 3
  Triangle changed!
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:1
```

```
  Massive:
  Size:10
  Elements:
  [0]:a=0, b=0, c=0

  [1]:a=0, b=0, c=0

  [2]:a=0, b=0, c=0

  [3]:a=0, b=0, c=0

  [4]:a=0, b=0, c=0

  [5]:a=0, b=0, c=0

  [6]:a=0, b=0, c=0

  [7]:a=5, b=4, c=3

  [8]:a=0, b=0, c=0

  [9]:a=0, b=0, c=0

  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:7
  Enter new size:8
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
```

```
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  New lenght:8
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:4
  Enter index:2
  Enter triangle:
  1 6 7
  Wrong sides! Triangle not changed!
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
```

```
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:4
  Enter index:3 4 6
  Enter triangle:
  4
  Triangle changed!
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:1
  Massive:
  Size:8
  Elements:
  [0]:a=0, b=0, c=0

  [1]:a=0, b=0, c=0

  [2]:a=0, b=0, c=0

  [3]:a=4, b=6, c=4

  [4]:a=0, b=0, c=0

  [5]:a=0, b=0, c=0

  [6]:a=0, b=0, c=0

  [7]:a=5, b=4, c=3
```

```
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:2
  Massive:
  Size:0
  Elements:
  Empty!

  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:5
  Enter index:3
  Wrong index!
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
```

```
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:8
  Enter new size:3
  Triangle created:

  0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  New lenght:3
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:5
  Enter index:2
  Enter triangle:
  6 9 4
  Triangle changed!
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
```

```
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:2
  Massive:
  Size:3
  Elements:
  [0]:a=0, b=0, c=0

  [1]:a=0, b=0, c=0

  [2]:a=6, b=9, c=4

  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:6
  Enter index:0
  Enter triangle:
  3 6 8
  Triangle changed!
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
```

```
  9-Resize №3
  0-Exit
  Enter your choise:9
  Enter new size:2
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  New lenght:2
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:3
  Massive:
  Size:2
  Elements:
  [0]:a=3, b=6, c=8

  [1]:a=0, b=0, c=0

  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
```

```
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:9
  Enter new size:5
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle created: 0, 0, 0
  Triangle deleted
  Triangle deleted
  New lenght:5
  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:3
  Massive:
  Size:5
  Elements:
  [0]:a=3, b=6, c=8

  [1]:a=0, b=0, c=0

  [2]:a=0, b=0, c=0
```

```
  [3]:a=0, b=0, c=0

  [4]:a=0, b=0, c=0

  ------------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:5
  Enter index:1
  Enter triangle:
  7 7 7
  Triangle changed!
  ------------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:2
  Massive:
  Size:3
  Elements:
  [0]:a=0, b=0, c=0

  [1]:a=7, b=7, c=7
```

```
  [2]:a=6, b=9, c=4

  ------------------------------------------------------------------
-----
  Menu
  1-Print massive №1
  2-Print massive №2
  3-Print massive №3
  4-Enter triangle in №1
  5-Enter triangle in №2
  6-Print triangle in №3
  7-Resize №1
  8-Resize №2
  9-Resize №3
  0-Exit
  Enter your choise:0
  Exit! Made by Bronnikov(№1 M8O-204)
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Massive deleted!
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Massive deleted!
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Triangle deleted
  Massive deleted!
```

Листинг программы

1) Figure.h

```
#ifndef FIGURE_H
#define	FIGURE_H

class Figure {
public:
	virtual double Square() = 0;
	virtual void   Print() = 0;
	virtual ~Figure() {};
};

#endif
```

2) Triangle.h

```
#ifndef TRIANGLE_H
#define	TRIANGLE_H
#include <cstdlib>
#include <iostream>
#include "Figure.h"

class Triangle : public Figure{
public:
	Triangle();
	Triangle(std::istream &is);
	Triangle(size_t ai,size_t bi,size_t ci);
	Triangle(const Triangle& orig);

	Triangle& operator++();
	double Square() override;
	void Print() override;
	friend Triangle operator+(const Triangle& left,const Triangle&
right);
	friend std::ostream& operator<<(std::ostream& os, const
Triangle& obj);
	friend std::istream& operator>>(std::istream& is,  Triangle&
obj);

	Triangle& operator=(const Triangle& right);

	virtual ~Triangle();
private:
	size_t a;
	size_t b;
	size_t c;
};

#endif
```

3) Massive.h

```
#ifndef MASSIVE_H
```

```
#define MASSIVE_H
#include "Figure.h"
#include "Triangle.h"

class TrMassive {
public:

  TrMassive();
  TrMassive(unsigned int l);
  TrMassive(const TrMassive& orig);

  bool Empty();
  void Destroy(int i);
  friend std::ostream& operator<<(std::ostream& os, const TrMassive&
mass);
  Triangle& operator[](const int index);
  int Lenght();
  void Resize(int l);
  ~TrMassive();

private:
  Triangle* data;
  int len;
};

#endif
```

4) Triangle.cpp

```
#include "Triangle.h"
#include <iostream>
#include <cmath>

int max(int a, int b);
int min(int a, int b);

int max(int a, int b){
               return a>b ? a:b;
}
int min(int a, int b){
       return a<b ? a:b;
}


Triangle::Triangle() : Triangle(0, 0, 0) {
}

Triangle::Triangle(size_t ai, size_t bi, size_t ci) {
       if(max(ai, max(bi, ci)) > min(bi, ci)+min(ai, max(bi, ci))){
               std::cout << "Wrong sides! Triangle not created!" <<
'\n';
       }       else if((ai>=0) && (bi>=0) && (ci>=0)){
                       a=ai;
                       b=bi;
```

```
                        c=ci;
                        std::cout << "Triangle created: " << a << ", " <<
b << ", " << c << std::endl;
        } else{
                        std::cout << "Wrong sizes! Triangle not created!"
<< '\n';
                }
}


Triangle::Triangle(std::istream &is) {
        int ai, bi, ci;
        is >> ai;
        is >> bi;
        is >> ci;
        if(max(ai, max(bi, ci)) >= min(bi, ci)+min(ai, max(bi, ci))){
                std::cout << "Wrong sides! Triangle not created!" <<
'\n';
        }
        else if(ai>=0&&bi>=0&&ci>0){
                a=ai;
                b=bi;
                c=ci;
                std::cout << "Triangle created: " << a << ", " << b << ",
" << c << std::endl;
        }
        else{
                std::cout << "Wrong sizes! Triangle not created!" <<
'\n';
        }
}


Triangle::Triangle(const Triangle& orig) {
        std::cout << "Triangle copy created" << std::endl;
        a = orig.a;
        b = orig.b;
        c = orig.c;
}

double Triangle::Square(){
        double p = double(a + b + c) / 2.0;
        return sqrt(p * (p - double(a))*(p - double(b))*(p -
double(c)));
}

Triangle& Triangle::operator=(const Triangle& right) {

        if (this == &right){
                return *this;
        }
        a = right.a;
        b = right.b;
        c = right.c;
```

```
	return *this;
}

Triangle& Triangle::operator++() {

	++a;
	++b;
	++c;

	return *this;
}

Triangle operator+(const Triangle& left,const Triangle& right) {

	return Triangle(left.a+right.a,left.b+right.b,left.c+right.c);
}

Triangle::~Triangle() {
	std::cout << "Triangle deleted" << std::endl;
}

std::ostream& operator<<(std::ostream& os, const Triangle& obj) {

	os << "a=" << obj.a << ", b=" << obj.b << ", c=" << obj.c <<
std::endl;
	return os;
}

void Triangle::Print(){
	std::cout << *this;
	return;
}

std::istream& operator>>(std::istream& is, Triangle& obj) {
	int ai, bi, ci;
	is >> ai;
	is >> bi;
	is >> ci;
	if(max(ai, max(bi, ci)) >= min(bi, ci)+min(ai, max(bi, ci))){
		std::cout << "Wrong sides! Triangle not changed!" <<
'\n';
	}
	else if(ai>=0&&bi>=0&&ci>0){
		obj.a=ai;
		obj.b=bi;
		obj.c=ci;
		std::cout << "Triangle changed! " << std::endl;
	}
	else{
		std::cout << "Wrong sizes! Triangle not changed!" <<
'\n';
	}
```

```
        return is;
}
```

5)Massive.cpp

```
#include "Massive.h"
#include "Figure.h"
#include "Triangle.h"
#include <iostream>
#include <cstdlib>

TrMassive::TrMassive() : TrMassive(0){}

TrMassive::TrMassive(unsigned int l) {
  data=nullptr;
  len=l;
  if(len>0){
    data = new Triangle[len];
  }
  std::cout << "massive created!" << std::endl;
}

TrMassive::TrMassive(const TrMassive& orig) {
  len=orig.len;
  data= new Triangle[len];
  for(int i=0; i<len; i++){
    data[i]=orig.data[i];
  }
  std::cout << "massive copied!" << '\n';
}

bool TrMassive::Empty(){
  return (len==0);
}

Triangle& TrMassive::operator[](const int index){
  if ((index >= len)||(index < 0)){
    std::cout << "Wrong index! Returning element with index 0!" <<
'\n';
    return data[0];
  }
  return data[index];
}

int TrMassive::Lenght(){
  return len;
}

TrMassive::~TrMassive(){
  delete[] data;
  len=0;
  std::cout << "Massive deleted!" << '\n';
}
```

```
void TrMassive::Destroy(int i){
  Triangle ex(0, 0, 0);
  data[i]=ex;
  return;
}


void TrMassive::Resize(int l){
  if(l<0){
    std::cout << "Wrong size!" << '\n';
    return;
  }
  Triangle* data1;
  if(l==0){
    data1 = nullptr;
  } else{
    data1 = new Triangle[l];
  }
  if (l<len){
    for (int i = 0; i < l; i++) {
      data1[i]=data[i];
    }} else{
      for(int i=0; i < len; i++){
        data1[i]=data[i];
      }}
  delete[] data;
  len=l;
  data=data1;
  data1=nullptr;
  return;
}

std::ostream& operator<<(std::ostream& os, const TrMassive& mass){
  std::cout << "Massive:" << '\n';
  std::cout << "Size:" << mass.len << '\n' << "Elements:" << std::endl;
  if(mass.len==0){
    std::cout << "Empty!" << '\n';
  }
  for(int i=0; i<mass.len; i++){
    std::cout << "[" << i << "]" << ":"<< mass.data[i] << '\n';
  }
  return os;
}
```

6) TListitem.cpp

```
/*Podgornaya, 12, trapeze, doubly-list, 15.10.2018*/
/*2 Лабораторная работа, 12 вариант*/
/*Необходимо спроектировать и запрограммировать на языке C++
класс-контейнер первого уровня
(связанный список, т.к. не было уточнения какой конкретно
список, использую двусвязный список),
```

```
содержащий одну фигуру (трапеция), согласно вариантов задания
(реализованную в ЛР1).*/
#include "TListItem.h"
#include <iostream>

TListItem::TListItem(const Trapeze &obj)
{
     this->item = obj;
     this->next = nullptr;
     this->prev = nullptr;
}
Trapeze TListItem::GetFigure() const
{
     return this->item;
}
TListItem* TListItem::GetNext()
{
     return this->next;
}
TListItem* TListItem::GetPrev()
{
     return this->prev;
}
void TListItem::SetNext(TListItem *item)
{
     this->next = item;
}
void TListItem::SetPrev(TListItem *item)
{
     this->prev = item;
}
std::ostream&  operator<<(std::ostream  &os,  const  TListItem
&obj)
{
     os << "(" << obj.item << ")" << std::endl;
     return os;
}
```

8) main.cpp

```
#include "Massive.h"
#include "Triangle.h"
#include "Figure.h"
```

```
#include <iostream>
//Бронников Максим Андреевич М8О-204Б-17
//Класс контейнер: массив
//Класс фигуры: треугольник
/* Необходимо спроектировать и запрограммировать на языке C++
класс-контейнер первого уровня, содержащий одну фигуру ( колонка фигура
1), согласно вариантов задания (реализованную в ЛР1).
Классы должны удовлетворять следующим правилам:
    • Требования к классу фигуры аналогичны требованиям из лабораторной
работы 1.
    • Классы фигур должны иметь переопределенный оператор вывода в
поток std::ostream (<<). Оператор должен распечатывать параметры фигуры
(тип фигуры, длины сторон, радиус и т.д).
    • Классы фигур должны иметь переопределенный оператор ввода фигуры
из потока std::istream (>>). Оператор должен вводить основные параметры
фигуры (длины сторон, радиус и т.д).
    • Классы фигур должны иметь операторы копирования (=).
    • Классы фигур должны иметь операторы сравнения с такими же
фигурами (==).
    • Класс-контейнер должен соджержать объекты фигур “по значению” (не
по ссылке).
    • Класс-контейнер должен иметь метод по добавлению фигуры в
контейнер.
    • Класс-контейнер должен иметь методы по получению фигуры из
контейнера (опередеряется структурой контейнера).
    • Класс-контейнер должен иметь метод по удалению фигуры из
контейнера (опередеряется структурой контейнера).
    • Класс-контейнер должен иметь перегруженный оператор по выводу
контейнера в поток std::ostream (<<).
    • Класс-контейнер должен иметь деструктор, удаляющий все элементы
контейнера.
    • Классы должны быть расположенны в раздельных файлах: отдельно
заголовки (.h), отдельно описание методов (.cpp).

Нельзя использовать:
    • Стандартные контейнеры std.
    • Шаблоны (template).
    • Различные варианты умных указателей (shared_ptr, weak_ptr).

Программа должна позволять:
    • Вводить произвольное количество фигур и добавлять их в контейнер.
    • Распечатывать содержимое контейнера.
    • Удалять фигуры из контейнера.*/
int main(){
  int i;
  TrMassive mass1(10);
  TrMassive mass2;
  TrMassive mass3(mass1);
  Triangle b(1, 1, 1);
  while(1){
   std::cout <<
"--------------------------------------------------------------------
" << '\n';
```

```
   std::cout << "Menu\n1-Print massive №1\n2-Print massive №2\n3-Print
massive №3\n4-Enter triangle in №1\n5-Enter triangle in №2\n6-Print
triangle in №3\n7-Resize №1\n8-Resize №2\n9-Resize №3\n0-Exit\nEnter
your choise:";
   std::cin >> i;
   switch (i) {
    case 1:
      std::cout << mass1 << '\n';
      break;
    case 2:
      std::cout << mass2 << '\n';
      break;
    case 3:
      std::cout << mass3 << '\n';
      break;
    case 4:
      std::cout << "Enter index:";
      std::cin >> i;
      if(0 < i < mass1.Lenght()){
        std::cout << "Enter triangle:" << '\n';
        std::cin >> mass1[i];
      } else {
        std::cout << "Wrong index!" << '\n';
      }
      break;
    case 5:
      std::cout << "Enter index:";
      std::cin >> i;
      if(0 < i < mass2.Lenght()){
        std::cout << "Enter triangle:" << '\n';
        std::cin >> mass2[i];
      } else {
        std::cout << "Wrong index!" << '\n';
      }
      break;
    case 6:
      std::cout << "Enter index:";
      std::cin >> i;
      if(0 < i < mass3.Lenght()){
        std::cout << "Enter triangle:" << '\n';
        std::cin >> mass3[i];
      } else {
        std::cout << "Wrong index!" << '\n';
      }
        break;
    case 7:
      std::cout << "Enter new size:";
      std::cin >> i;
      mass1.Resize(i);
      std::cout << "New lenght:" << mass1.Lenght() <<'\n';
      break;
    case 8:
      std::cout << "Enter new size:";
      std::cin >> i;
```

```
      mass2.Resize(i);
      std::cout << "New lenght:" << mass2.Lenght() <<'\n';
      break;
    case 9:
      std::cout << "Enter new size:";
      std::cin >> i;
      mass3.Resize(i);
      std::cout << "New lenght:" << mass3.Lenght() <<'\n';
      break;
    case 0:
      std::cout << "Exit! Made by Bronnikov(№1 M8O-204)" << '\n';
      return 0;
      break;
}}}
```

Вывод:

В этой лабораторной было предложено написать свою структуру данных. В моем случае это массив. С чем я удачно справился. Конечно, все эти контейнеры есть в стандартной библиотеке шаблонов, но любому профессиональному программисту не составит труда реализовать свой массив, стек, очередь или любую другую широко используемую структуру данных. Фигуры передаются в массив “по значению” чтобы в них хранился сам объект, а не его копия.

СПИСОК ЛИТЕРАТУРЫ

1. Справочник по языку c и c++ [Электронный ресурс].
URL : http://www.c-cpp.ru (дата обращения : 15.10.2018)

2.Видеоуроки по программированию на c++ [Электронный ресурс].
URL :
https://www.youtube.com/watch?v=kRcbYLK3OnQ&list=PLQOaTSbfxUtCrKs0nicOg2npJQYSPGO9r
(дата обращения : 14.10.2018)

3. Ошибки c и c++ в Microsoft Visual Studio 2017[Электронный ресурс].
URL :
https://docs.microsoft.com/ru-ru/cpp/error-messages/compiler-errors-1/c-cpp-build-errors?view=vs-2017
(дата обращения : 14.10.2018)