**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 7**

Тема: Проектирование структуры классов

Студент: Курносов М. Ю.

Группа: 80-208

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

**Постановка задачи**

**Вариант 25**

Спроектировать простейший «графический» векторный редактор.

Требование к функционалу редактора:

· создание нового документа

· импорт документа из файла

· экспорт документа в файл

· создание графического примитива (согласно варианту задания)

· удаление графического примитива

· отображение документа на экране (печать перечня графических объектов и их характеристик в std::cout)

· реализовать операцию undo, отменяющую последнее сделанное действие. Должно действовать для операций добавления/удаления фигур.

Требования к реализации:

· Создание графических примитивов необходимо вынести в отдельный класс – Factory.

· Сделать упор на использовании полиморфизма при работе с фигурами;

· Взаимодействие с пользователем (ввод команд) реализовать в функции main;

Вариант:

|  |  |  |  |
| --- | --- | --- | --- |
| 25 | Треугольник | Квадрат | Прямоугольник |

**Описание программы**

Программа состоит из одного файла main.cpp. Где реализованы классы графических примитивов, их методы и взаимодействие с пользователем. При запуске программы создается динамический массив std::vector и пользователю предоставляется доступ к меню, с помощью которого он может добавлять, удалять и производить другие операции, установленные заданием. Программа принимает на вход только значения типа int (согласно заданию). Программа способна уведомлять пользователя о неправильном вводе и отменить текущую операцию. Добавление и удаление осуществляется в виде работающих по индексу функций. Так же можно отменить несколько последних действий удаления или вставки, для чего создан отдельный массив фигур и массив, хранящий последовательность нескольких последних введённых команд, на момент написания отчёта кол-во отменяемых операций – 5.

**Набор тестов**

|  |  |  |
| --- | --- | --- |
| For file test\_01.txt: | For file test\_02.txt: | For file test\_03.txt: |
| 4  1  1  2  3  4  4  2  1  2  3  4  4  3  1  2  3  4  6  3  oru  5  2  6  0 | 4  1  1  2  3  4  4  2  0  2  3  4  4  3  1  2  3  4  6  5  0  5  2  6  7  6  0 | 4 1  1  2  3  4  4  2  0  2  3  4  4  3  1  2  3  4  6  3  save  5  0  5  2  6  7  6  5  1  6  5  1  6  5  1  6  0 |

|  |  |  |
| --- | --- | --- |
| For file test\_04.txt:  4  1  1  2  3  4  4  2  0  2  3  4  4  3  1  2  3  4  6  3  save  5  0  5  2  6  7  6  5  1  6  5  1  6  5  1  6  2  save  6  0 | For file test\_05.txt:  4  1  1  2  3  4  4  2  0  2  3  4  4  3  1  2  3  4  6  3  save  5  0  5  2  6  7  6  1  6  2  save  6  0 |  |

Пояснение к тестам:

В 1 тесте мы тестируем вставку элементов и удаление их из коллекции, а так же их вывод.

Во 2 тесте главным образом проверяется отмена операции.

В 3 тесте мы проверяем удаление и удаление из пустого массива.

В 4 тесте проверяется импорт и экспорт.

В 5 тесте вновь проверяется экспорт и импорт, но перед экспортом создаётся новый файл, а не удаляются все элементы.

**Результат выполнения тестов**

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/7laboop# g++ -o p main.cpp

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/7laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - создание нового документа

2 - импорт документа из файла

3 - экспорт документа в файл

4 - создание фигуры

5 - удаление фигуры

6 - Распечатать все фигуры

7 - Отмена

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Экспорт произведён.

Фигура удалена

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/7laboop# ./p < test\_02.txt

Для выполнения операции введите соответствующую цифру:

1 - создание нового документа

2 - импорт документа из файла

3 - экспорт документа в файл

4 - создание фигуры

5 - удаление фигуры

6 - Распечатать все фигуры

7 - Отмена

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Элемента с таким номером нет.

Фигура удалена

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Действие отменено.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/7laboop# ./p < test\_03.txt

Для выполнения операции введите соответствующую цифру:

1 - создание нового документа

2 - импорт документа из файла

3 - экспорт документа в файл

4 - создание фигуры

5 - удаление фигуры

6 - Распечатать все фигуры

7 - Отмена

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Экспорт произведён.

Элемента с таким номером нет.

Фигура удалена

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Действие отменено.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Фигура удалена

Фигура №1

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Фигура удалена

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Фигура удалена

Фигур нет.

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/7laboop# ./p < test\_04.txt

Для выполнения операции введите соответствующую цифру:

1 - создание нового документа

2 - импорт документа из файла

3 - экспорт документа в файл

4 - создание фигуры

5 - удаление фигуры

6 - Распечатать все фигуры

7 - Отмена

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Экспорт произведён.

Элемента с таким номером нет.

Фигура удалена

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Действие отменено.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Фигура удалена

Фигура №1

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Фигура удалена

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Фигура удалена

Фигур нет.

Импорт произведён.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/7laboop# ./p < test\_05.txt

Для выполнения операции введите соответствующую цифру:

1 - создание нового документа

2 - импорт документа из файла

3 - экспорт документа в файл

4 - создание фигуры

5 - удаление фигуры

6 - Распечатать все фигуры

7 - Отмена

0 - Выход

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Введите номер добавляемой фигуры:

1 - Треугольник

2 - Прямоугольник

3 - Квадрат

Элемент вставлен.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Экспорт произведён.

Элемента с таким номером нет.

Фигура удалена

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Действие отменено.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Открыт новый файл.

Фигур нет.

Импорт произведён.

Фигура №1

Вершина №1: 1 2

Вершина №2: 3 4

Вершина №3: 0 4

Фигура №2

Вершина №1: 0 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 0 4

Фигура №3

Вершина №1: 1 2

Вершина №2: 3 2

Вершина №3: 3 4

Вершина №4: 1 4

Выход из программы...

**Листинг программы**

([git@github.com:Arhangel333/oop\_exercise\_06.git](mailto:git@github.com:Arhangel333/oop_exercise_06.git))

**For file main.cpp:**

#include <iostream>

#include <cmath>

//#include "vector.hpp"

#include <vector>

#include <algorithm>

#include <fstream>

using namespace std;

typedef int TP;

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;

}

}

};

/\* создание нового документа

импорт документа из файла

экспорт документа в файл

создание графического примитива (согласно варианту задания)

удаление графического примитива

отображение документа на экране (печать перечня графических объектов и их характеристик в std::cout)

реализовать операцию undo, отменяющую последнее сделанное действие. Должно действовать для операций добавления/удаления фигур.

\*/

void help()

{

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

printf("\t 1 - создание нового документа\n ");

printf("\t 2 - импорт документа из файла\n ");

printf("\t 3 - экспорт документа в файл\n ");

printf("\t 4 - создание фигуры\n ");

printf("\t 5 - удаление фигуры\n ");

printf("\t 6 - Распечатать все фигуры\n ");

printf("\t 7 - Отмена\n ");

printf("\t 0 - Выход\n ");

}

int main()

{

int i = 1;

char enter = -1;

TP xl, yl, xr, yr;

const int bufsize = 5;

int numb[bufsize], undo[bufsize], un = 0;

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

{

numb[i] = -1;

undo[i] = -1;

}

Factory<TP> nd[bufsize];

vector<Factory<TP>> vec;

help();

std::cin >> enter;

while (isdigit(enter))

{

switch (enter - 48)

{

case 1://printf("\t 1 - создание нового документа\n ");

{

vec.clear();

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

{

numb[i] = -1;

undo[i] = -1;

}

un = 0;

printf("Открыт новый файл.\n ");

break;

}

case 2://printf("\t 2 - импорт документа из файла\n ");

{

char filename[260];

int c,n1,n2;

cin>>filename;

ifstream fin(filename);

while(fin>>c){

switch (c)

{

case 1:

{

fin >> xl >> yl >> xr >> yr >> n1 >> n2;

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

vec.push\_back(d);

break;

}

case 2:

{

fin >> xl >> yl >> n1 >> n2 >> xr >> yr>> n1 >> n2;

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

vec.push\_back(a);

break;

}

case 3:

{

fin >> xl >> yl >> n1 >> n2 >> xr >> yr>> n1 >> n2;

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

vec.push\_back(a);

break;

}

default:printf("Введённый номер не соответствует фигуре\n");

break;

}

};

fin.close();

printf("Импорт произведён.\n ");

break;

}

case 3://printf("\t 3 - экспорт документа в файл\n ");

{

char filename[260];

cin>>filename;

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

for\_each(vec.begin(), vec.end(), [&](Factory<TP> &tr) {

fprintf(fl, "%d\n", tr.classid);

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

{

fprintf(fl, "%d %d\n", tr.a[i].first, tr.a[i].second);

}

});

printf("Экспорт произведён.\n ");

fclose(fl);

break;

}

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

{

int fig = 0;

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

cin>>fig;

switch (fig)

{

case 1:

{

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

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

vec.push\_back(a);

undo[un % bufsize] = enter - 48;

un++;

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

break;

}

case 2:

{

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

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

vec.push\_back(a);

undo[un % bufsize] = enter - 48;

un++;

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

break;

}

case 3:

{

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

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

vec.push\_back(a);

undo[un % bufsize] = enter - 48;

un++;

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

break;

}

default:printf("Введённый номер не соответствует фигуре\n");

break;

}

break;

}

case 5://printf("\t 5 - удаление фигуры\n ");

{

int fig = -1;

cin>>fig;

vector<Factory<TP>>::iterator itbeg(vec.begin()), itend(vec.end()), it(vec.begin() + fig-1);

if((it < itbeg)||(it >= itend)){

printf("Элемента с таким номером нет.\n");

break;

}else{

nd[un % bufsize] = vec[fig-1];

vec.erase(it);

undo[un % bufsize] = enter - 48;

numb[un % bufsize] = fig-1;

un++;

printf("Фигура удалена\n");

break;

}

}

case 6://printf("\t 6 - Распечатать все фигуры\n ");

{

if(vec.size() == 0){

printf("Фигур нет.\n");

}

for\_each(vec.begin(), vec.end(), [&](Factory<TP> &tr) {

printf("Фигура №%d\n", i);

i++;

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

{

printf("Вершина №%d: ", i+1);

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

}

});

i = 1;

break;

}

case 7://printf("\t 7 - Отмена\n ");

{

/\* for (int i = 0; i < bufsize; i++)

{

printf("%d ", undo[i]);

}

printf("\n"); \*/

if (undo[un-1 % bufsize] == -1)

{

printf("В буфере нет информации о введённых операциях\nВозможно вы отменили слишком много действий(размер буыера %d)\n", bufsize);

}else{

un--;

switch (undo[un % bufsize])

{

case 4://create

{

vec.pop\_back();

printf("Действие отменено.\n");

break;

}

case 5://delete

{

vector<Factory<TP>>::iterator it(vec.begin() + numb[un % bufsize]);

vec.insert(it, Factory<TP>(nd[un % bufsize]));

printf("Действие отменено.\n");

}

default:

break;

}

}

break;

}

case 0://printf("\t 0 - Выход\n ");

{

std::cout << "Выход из программы..." << endl;

return 0;

}

default:

{

help();

break;

}

}

enter = 48;

std::cin >> enter;

}

if(!(isdigit(enter))){

std::cout << "Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-) \n\n"<< endl;

}

return 0;

}

**Вывод**

Получены практические навыки в проектировании структуры классов приложения. Закреплены навыки работы с классами и стандартными библиотеками, итераторами, библиотекой работы с файлами в с++.

**Литература:**

1. Классы с++ [Электронный ресурс]URL: <https://codelessons.ru/cplusplus/klassy-v-c-rukovodstvo-dlya-nachinayushhix.html> (Дата обращения 05.12.20).
2. Вектор std::vector [Электронный ресурс] URL:  
   https://ravesli.com/urok-95-vvedenie-v-std-vector-vektory/ (Дата обращения 18.12.20).
3. STL Итераторы С++ URL[Электронный ресурс]: <https://ravesli.com/urok-198-iteratory-stl/#toc-2> (Дата обращения 18.11.20).
4. Шаблоны и шаблонные функции в C++[Электронный ресурс]:<https://code-live.ru/post/cpp-template-functions/> (Дата обращения 05.12.20).
5. Работа с файлами в С++ [Электронный ресурс] URL: http://cppstudio.com/post/446/ (Дата обращения 18.12.20).