Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №5 за
4 семестр
По дисциплине: «ООТиСП»
Тема: «СТАНДАРТНАЯ БИБЛИОТЕКА ШАБЛОНОВ»

Выполнил: Студент
2 курса
Группы ПО-6(1)
Лавренчик Д.О.
Проверил:
Булей Е.В

Брест 2022

Цель. Освоить технологию обобщенного программирования с использованием библиотеки стандартных шаблонов (STL) языка C++.

Вариант 9

Написать и отладить три программы. Первая программа демонстрирует использование контейнерных классов для хранения встроенных типов данных.

Вторая программа демонстрирует использование контейнерных классов для хранения пользовательских типов данных.

Третья программа демонстрирует использование алгоритмов STL.

В программе № 1 выполнить следующее:

1. Создать объект-контейнер в соответствии с вариантом задания и заполнить его данными, тип которых определяется вариантом задания.

2. Просмотреть контейнер.

3. Изменить контейнер, удалив из него одни элементы и заменив другие.

4. Просмотреть контейнер, используя для доступа к его элементам итераторы.

5. Создать второй контейнер этого же класса и заполнить его данными того же типа, что и первый контейнер.

6. Изменить первый контейнер, удалив из него n элементов после заданного и добавив затем в него все элементы из второго контейнера.

7. Просмотреть первый и второй контейнеры.

В программе № 2 выполнить то же самое, но для данных пользовательского типа.

В программе № 3 выполнить следующее:

1. Создать контейнер, содержащий объекты пользовательского типа. Тип контейнера выбирается в соответствии с вариантом задания.

2. Отсортировать его по убыванию элементов.

3. Просмотреть контейнер.

4. Используя подходящий алгоритм, найти в контейнере элемент, удовлетворяющий заданному условию.

5. Переместить элементы, удовлетворяющие заданному условию в другой (предварительно пустой) контейнер. Тип второго контейнера определяется вариантом задания.

6. Просмотреть второй контейнер.

7. Отсортировать первый и второй контейнеры по возрастанию элементов.

8. Просмотреть их.

9. Получить третий контейнер путем слияния первых двух.

10. Просмотреть третий контейнер.

11 .Подсчитать, сколько элементов, удовлетворяющих заданному условию, содержит третий контейнер.

12.Определить, есть ли в третьем контейнере элемент, удовлетворяющий заданному условию.

Код программы:

```
#include <iostream>
#include <set>
#include <Windows.h>
#include <iterator>
#include <algorithm>
#include <stack>

using namespace std;

class Rect {
private:
    int a;
    int b;
public:
    Rect(int a = 0, int b = 0) {
        this->a = a;
        this->b = b;
    }

    int getA() const{
        return this->a;
    }

    int getB() const {
        return this->b;
    }
    int getS()
    {
        return a * b;
    }

    friend ostream& operator<<(ostream& out, const Rect& rect) {
        out << "Прямоугольник со сторонами: a=" << rect.a
            << ", b=" << rect.b;
```

```
        return out;
    }

    friend std::istream& operator>> (std::istream& in, Rect& rect) {
        in >> rect.a;
        in >> rect.b;
        return in;
    }

    bool operator!=(const Rect& rect) {
        return (this->a != rect.a && this->b != rect.b);
    }

    bool operator< (const Rect& rect) const {
        int squere1 = this->a * this->b;
        int squere2 = rect.a * rect.b;
        return squere1 < squere2;
    }

    ~Rect() { }
};
class Rect1 {
private:
    int a;
    int b;
public:
    Rect1(int a = 0, int b = 0) {
        this->a = a;
        this->b = b;
    }

    int getA() const{
        return this->a;
    }
    int getS()
    {
        return a * b;
    }

    int getB() const{
        return this->b;
    }

    friend ostream& operator<<(ostream& out, const Rect1& rect) {
        out << "Прямоугольник со сторонами: a=" << rect.a
            << ", b=" << rect.b;
        return out;
    }

    friend std::istream& operator>> (std::istream& in, Rect1& rect) {
        in >> rect.a;
        in >> rect.b;
        return in;
    }

    bool operator!=(const Rect1& rect) {
        return (this->a != rect.a && this->b != rect.b);
    }
    bool operator=(const Rect1& rect) {
        if (this->a=rect.a&& this->b==rect.b)
        {
            return true;
        }
        else
        {
```

```
            return false;
        }
    }

    bool operator< (const Rect1& rect) const {
        int squere1 = this->a * this->b;
        int squere2 = rect.a * rect.b;
        return squere1 > squere2;
    }
    bool comp(Rect1 rect1, Rect1 rect2);
    ~Rect1() { }
};
bool Rect1::comp(Rect1 rect1, Rect1 rect2){
    int squere1 = rect1.getA() * rect1.getB();
    int squere2 = rect2.getA() * rect2.getB();
    return squere1 > squere2;
}



void printWithIterators(set<Rect> cont) {
    for (auto it = cont.begin(); it != cont.end(); ++it) {
        cout << *it << " " << endl;
    }
}
void printWithIterators(set<Rect1> cont) {
    for (auto it = cont.begin(); it != cont.end(); ++it) {
        cout << *it << " " << endl;
    }
    cout << endl;
}


void task1() {
    int size = 0, temp = 0;
    srand(time(0));
    set <int> st;
    cout << "ЗАДАНИЕ 1!!!" << endl;
    cout << "введите размер: ";
    cin >> size;
    for (int i = 0; i < size; i++) {
        cout << "введите знач:";
        cin >> temp;
        st.insert(temp);
    }
    set <int>::iterator it = st.begin();
    for (auto i : st)
    {
        cout << i << endl;
    }
    cout << "введите значение элемента для удаления: ";
    cin >> temp;
    st.erase(temp);
    it = st.begin();
    for (; it != st.end(); it++)
    {
        cout << *it << endl;
    }

    set <int> sts;
    cout << "введите размер2: ";
    cin >> size;
    for (int i = 0; i < size; i++)
    {
        temp = i + rand() % 10;
```

```
        sts.insert(temp);
    }
    cout << "Содержимое set: "<<endl;
    copy(sts.begin(), sts.end(), ostream_iterator<int>(cout, " \n"));
    cout << endl;
    cout << "введите значение элемента для удаления: ";
    cin >> temp;
    it = sts.find(temp);
    sts.erase(++it, sts.end());
    copy(sts.begin(), sts.end(), inserter(st, st.end()));
    it = st.begin();
    for (; it != st.end(); it++)
    {
        cout << *it << endl;
    }
}
void task2() {
    set<Rect> st;
    Rect obj1(2, 4);
    Rect obj2(3, 6);
    Rect obj3(8, 19);
    Rect obj4(1, 4);
    Rect obj5(10, 7);
    cout << "ЗАДАНИЕ 2!!!" << endl;
    set <Rect>::iterator it = st.begin();
    st.insert(obj1);
    st.insert(obj2);
    st.insert(obj3);
    st.insert(obj4);
    st.insert(obj5);
    printWithIterators(st);
    cout<<endl;
    st.erase(st.begin());
    Rect obj6(3, 2);
    st.insert(obj6);
    printWithIterators(st);
    set <Rect> sts;
    for (int i = 0; i < 5; i++)
    {
        sts.insert(Rect(rand() % 10 + 1, rand() % 10 + 1));
    }

    cout << "Содержимое set: ";
    copy(sts.begin(), sts.end(), ostream_iterator<Rect>(cout, " "));
    cout << endl;
    st.erase(--it, st.end());
    copy(sts.begin(), sts.end(), inserter(st, st.end()));
    for (it = st.begin(); it != st.end(); ++it)
    {
        cout << *it << endl;
    }
}
void task3()
{
    int temp=0;
    set<Rect1> st;
    set<Rect1> sts;
    stack<Rect> stak;
    stack<Rect> stac;
    st.insert(Rect1(3,2));
    st.insert(Rect1(6,1));
    //1
    for (int i = 0; i < 5; i++)
    {
        st.insert(Rect1(rand() % 10 + 1, rand() % 10 + 1));
```

```
    }
    set<Rect1>::iterator it = st.begin();
    //3
    cout << "Первый контейнер" << endl;
    printWithIterators(st);
    cout << endl;
    //4,5
    cout << "введите площадь = ";
    cin >> temp;
    for (it=st.begin();it!=st.end();++it)
    {
        if (((*it).getA() * (*it).getB())==temp)
        {
            stak.push(Rect((*it).getA(),(*it).getB()));
        }
    }
    //6
    cout << "второй контейнер" << endl;
    cout << stak.top() << endl;
    set<Rect> set1;
    //7
    for ( it = st.begin(); it != st.end(); ++it)
    {
        set1.insert(Rect((*it).getA(), (*it).getB()));
    }
    //8
    cout << "отсортированный первый контейнер" << endl;
    printWithIterators(set1);
    cout << "второй контейнер" << endl;
    cout << stak.top() << endl;
    //9
    set<Rect>::iterator its = set1.begin();
    for (; its!=set1.end(); ++its)
    {
        stac.push(Rect((*its).getA(), (*its).getB()));
    }
    stac.push(stak.top());
    cout << "третий контейнер" << endl;
    //10
    while (!stac.empty())
    {
        cout << stac.top() << endl;
        stac.pop();
    }
    //11
    for (its = set1.begin(); its != set1.end(); ++its)
    {
        stac.push(Rect((*its).getA(), (*its).getB()));
    }
    stac.push(stak.top());
    int sum = 0;
    while (!stac.empty())
    {
        if (stac.top().getS() == temp)
        {
            sum++;
        }
        stac.pop();
    }
    cout << "количество подходящих элементов =" << sum << endl;
}
int main()
{
    srand(time(0));
```

```
    SetConsoleOutputCP(1251);
    SetConsoleCP(1251);
    task2();
    return 0;
}
```

Пример работы:

```
ЗАДАНИЕ 1!!!
введите размер: 3
введите знач:1
введите знач:2
введите знач:3
1
2
3
введите значение элемента для удаления: 2
1
3
введите размер2: 3
Содержимое set:
6
9
10

введите значение элемента для удаления: 9
1
3
6
9
```

```
ЗАДАНИЕ 2!!!
Прямоугольник со сторонами: a=1, b=4
Прямоугольник со сторонами: a=2, b=4
Прямоугольник со сторонами: a=3, b=6
Прямоугольник со сторонами: a=10, b=7
Прямоугольник со сторонами: a=8, b=19

Прямоугольник со сторонами: a=3, b=2
Прямоугольник со сторонами: a=2, b=4
Прямоугольник со сторонами: a=3, b=6
Прямоугольник со сторонами: a=10, b=7
Прямоугольник со сторонами: a=8, b=19
Содержимое set: Прямоугольник со сторонами: a=1, b=1 Прямоугольник со сторонами: a=10, b=3
=8, b=4 Прямоугольник со сторонами: a=5, b=8 Прямоугольник со сторонами: a=9, b=9
Прямоугольник со сторонами: a=1, b=1
Прямоугольник со сторонами: a=3, b=2
Прямоугольник со сторонами: a=2, b=4
Прямоугольник со сторонами: a=3, b=6
Прямоугольник со сторонами: a=10, b=3
Прямоугольник со сторонами: a=8, b=4
Прямоугольник со сторонами: a=5, b=8
Прямоугольник со сторонами: a=10, b=7
Прямоугольник со сторонами: a=9, b=9
```

```
Прямоугольник со сторонами: a=8, b=8
Прямоугольник со сторонами: a=6, b=7
Прямоугольник со сторонами: a=3, b=10
Прямоугольник со сторонами: a=5, b=5
Прямоугольник со сторонами: a=4, b=6
Прямоугольник со сторонами: a=3, b=2


введите площадь = 6
второй контейнер
Прямоугольник со сторонами: a=3, b=2
отсортированный первый контейнер
Прямоугольник со сторонами: a=3, b=2
Прямоугольник со сторонами: a=4, b=6
Прямоугольник со сторонами: a=5, b=5
Прямоугольник со сторонами: a=3, b=10
Прямоугольник со сторонами: a=6, b=7
Прямоугольник со сторонами: a=8, b=8
второй контейнер
Прямоугольник со сторонами: a=3, b=2
третий контейнер
Прямоугольник со сторонами: a=3, b=2
Прямоугольник со сторонами: a=8, b=8
Прямоугольник со сторонами: a=6, b=7
Прямоугольник со сторонами: a=3, b=10
Прямоугольник со сторонами: a=5, b=5
Прямоугольник со сторонами: a=4, b=6
Прямоугольник со сторонами: a=3, b=2
количество подходящих элементов =2
```

Вывод: Освоил технологию обобщенного программирования с использованием библиотеки стандартных шаблонов (STL) языка C++.