Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Кафедра **«**Информационные технологии и автоматизированные системы**»**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №11**

Дисциплина: «Основы алгоритмизации и программирования»

Семестр 2

Тема: Последовательные контейнеры библиотеки STL

Вариант 6.

Выполнил работу

Студент группы РИС-22-1Б

Поважный В. Е.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

г. Пермь-2023

**Введение**

Для выполнения лабораторной работы требуется сделать отчет программы. Создание программ – отличный способ практики программирования.

**Постановка задачи**

Задача 1:

1. Создать последовательный контейнер.

2. Заполнить его элементами стандартного типа (тип указан в варианте).

3. Добавить элементы в соответствии с заданием

4. Удалить элементы в соответствии с заданием.

5. Выполнить задание варианта для полученного контейнера.

6. Выполнение всех заданий оформить в виде глобальных функций.

Указания варианта:

1. Контейнер - список; 2. Тип элементов - double

Задача 2:

1. Создать последовательный контейнер.

2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.

3. Добавить элементы в соответствии с заданием

4. Удалить элементы в соответствии с заданием.

5. Выполнить задание варианта для полученного контейнера.

6. Выполнение всех заданий оформить в виде глобальных функций.

Указание варианта:

Тип элементов Pair (см. лабораторную работу №3).

Задача 3

1. Создать параметризированный класс, используя в качестве контейнера последовательный контейнер.

2. Заполнить его элементами.

3. Добавить элементы в соответствии с заданием

4. Удалить элементы в соответствии с заданием.

5. Выполнить задание варианта для полученного контейнера.

6. Выполнение всех заданий оформить в виде методов параметризированного класса.

Указание варианта:

Параметризированный класс – Список (см. лабораторную работу №7).

Задача 4

1. Создать адаптер контейнера.

2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.

3. Добавить элементы в соответствии с заданием

4. Удалить элементы в соответствии с заданием.

5. Выполнить задание варианта для полученного контейнера.

6. Выполнение всех заданий оформить в виде глобальных функций.

Указание варианта:

Адаптер контейнера – очередь с приоритетами.

Задача 5

1. Создать параметризированный класс, используя в качестве контейнера адаптер контейнера.

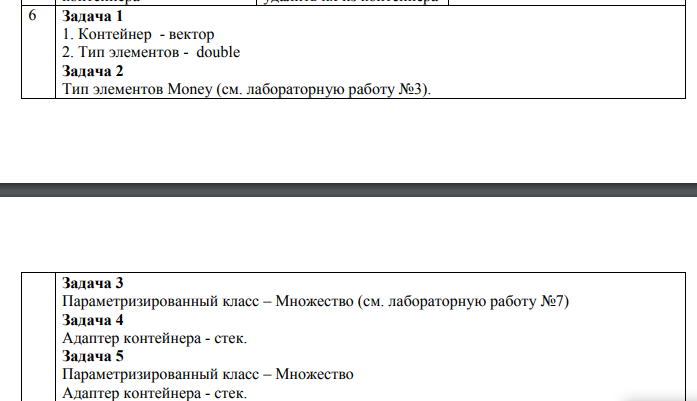
2. Заполнить его элементами.

3. Добавить элементы в соответствии с заданием

4. Удалить элементы в соответствии с заданием.

5. Выполнить задание варианта для полученного контейнера.

6. Выполнение всех заданий оформить в виде методов параметризованного класса.



**Код программы**

**Задача 1**

#include <iostream>

#include <vector>

#include <cstdlib>

using namespace std;

typedef vector<float> Vec;

Vec make\_vector(int n)

{

Vec v;

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

v.push\_back(rand() % 100 - 50);

return v;

}

void print\_vector(Vec v)

{

for (int i = 0; i < v.size(); i++)

cout << v[i] << " ";

cout << endl;

}

int srednee(Vec v)

{

int s = 0;

for (int i = 0; i < v.size(); i++)

s += v[i];

return s / v.size();

}

void add\_vector(Vec& v, int el, int pos)

{

v.insert(v.begin() + pos, el);

}

int max(Vec v)

{

int m = v[0];

int n = 0;

for (int i = 0; i < v.size(); i++)

if (m < v[i])

{

m = v[i];

n = i;

}

return n;

}

void del\_vector(Vec& v, int pos)

{

v.erase(v.begin() + pos);

}

int min(Vec v)

{

int m = v[0];

int n = 0;

for (int i = 0; i < v.size(); i++)

if (m > v[i])

{

m = v[i];

n = i;

}

return n;

}

void delenie(Vec& v)

{

int m = min(v);

for (int i = 0; i < v.size(); i++)

v[i] = v[i] / v[m];

}

void main()

{

try

{

vector<float> v;

vector<float>::iterator vi = v.begin();

int n;

cout << "Size: ";

cin >> n;

v = make\_vector(n);

print\_vector(v);

int el = srednee(v);

cout << "Pos: ";

int pos;

cin >> pos;

if (pos > v.size())

throw 1;

add\_vector(v, el, pos);

int n\_del;

cout << "Delete: " << endl;

cin >> n\_del;

del\_vector(v, n\_del);

print\_vector(v);

}

catch (int)

{

cout << "error!";

}

}

**Задача 2**

**#include** <vector>

#include "Header.h"

#include <iostream>

using namespace std;

typedef vector<Money> Vec;

Vec make\_vector(int n)

{

Vec v;

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

{

Money t;

cin >> t;

v.push\_back(t);

}

return v;

}

void print\_vector(Vec v)

{

for (int i = 0; i < v.size(); i++)

cout << v[i] << " ";

cout << endl;

}

Money srednee(Vec v)

{

int m = 0;

int s = 0;

for (int i = 0; i < v.size(); i++)

{

m += v[i].get\_rub();

s += v[i].get\_kop();

}

int n = v.size();

Money p;

p.set\_rub(m / n);

p.set\_kop(s / n);

return p;

}

void add\_vector(Vec& v, Money el, int pos)

{

v.insert(v.begin() + pos, el);

}

int max(Vec v)

{

Money m = v[0];

int n = 0;

for (int i = 0; i < v.size(); i++)

if (m < v[i])

{

m = v[i];

n = i;

}

return n;

}

void del\_vector(Vec& v, int pos)

{

v.erase(v.begin() + pos);

}

int min(Vec v)

{

Money m = v[0];

int n = 0;

for (int i = 0; i < v.size(); i++)

if (m > v[i])

{

m = v[i];

n = i;

}

return n;

}

void delenie(Vec& v)

{

int m = min(v);

for (int i = 0; i < v.size(); i++)

v[i] = v[i] / v[m];

}

void main()

{

try

{

vector<Money> v;

vector<Money>::iterator vi = v.begin();

int n;

cout << "Kolvo elementov: ";

cin >> n;

v = make\_vector(n);

print\_vector(v);

Money el = srednee(v);

cout << "middle number pos: ";

int pos;

cin >> pos;

if (pos > v.size())

throw 1;

add\_vector(v, el, pos);

print\_vector(v);

cout << "Max Element: " << v[max(v)] << "\n";

del\_vector(v, max(v));

delenie(v);

print\_vector(v);

}

catch (int)

{

cout << "error!";

}

}

**Задача 3**

**#include "Vector.h"**

#include <iostream>

#include "Money.h"

using namespace std;

void main()

{

Vector<Money>vec(2);

vec.Print();

Money s = vec.Srednee();

cout << "Middle = " << s << endl;

cout << "Pos: ";

int p;

cin >> p;

p = vec.Min();

vec.Add(s, p);

vec.Print();

p = vec.Max();

vec.Del(p);

vec.Print();

vec.Delenie();

vec.Print();

}

**Задача 4**

**#include <iostream>**

#include <stack>

#include <vector>

#include "Money.h"

using namespace std;

typedef stack<Money> St;

typedef vector<Money> Vec;

St make\_stack(int n)

{

St s;

Money t;

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

{

cin >> t;

s.push(t);

}

return s;

}

void print\_stack(St s)

{

Vec v;

while (!s.empty())

{

cout << s.top() << endl;

v.push\_back(s.top());

s.pop();

}

}

Vec copy\_stack\_to\_vector(St s)

{

Vec v;

while (!s.empty())

{

v.push\_back(s.top());

s.pop();

}

return v;

}

St copy\_vector\_to\_stack(Vec v)

{

St s;

for (int i = 0; i < v.size(); i++)

s.push(v[i]);

return s;

}

Money Max(St s)

{

Money m = s.top();

Vec v = copy\_stack\_to\_vector(s);

while (!s.empty())

{

if (s.top() > m)

m = s.top();

s.pop();

}

s = copy\_vector\_to\_stack(v);

return m;

}

void Delete\_from\_stack(St& s)

{

Vec v;

Money t;

while (!s.empty())

{

t = s.top();

s.pop();

}

s = copy\_vector\_to\_stack(v);

}

Money Min(St s)

{

Money m = s.top();

Vec v = copy\_stack\_to\_vector(s);

while (!s.empty())

{

if (s.top() < m)

m = s.top();

s.pop();

}

s = copy\_vector\_to\_stack(v);

return m;

}

void Delenie(St& s)

{

Money m = Min(s);

Vec v;

Money t;

while (!s.empty())

{

t = s.top();

v.push\_back(t / m);

s.pop();

}

s = copy\_vector\_to\_stack(v);

}

void Add\_to\_stack(St& s, Money el)

{

s.push(el);

}

void del\_to\_queue(St& s, int pos)

{

int i = 1;

Vec v;

Money t;

while (!s.empty())

{

t = s.top();

if (i != pos)

v.push\_back(t);

s.pop();

i++;

}

s = copy\_vector\_to\_stack(v);

}

int main()

{

Money t;

St s;

int n;

cin >> n;

s = make\_stack(n);

print\_stack(s);

cout << "Max: " << Max(s) << endl;

cout << "Remove Pos: ";

int pos;

cin >> pos;

del\_to\_queue(s, pos);

print\_stack(s);

**}**

**Задача 5**

**#include "Money.h"**

#include <iostream>

#include <stack>

#include <vector>

#include "Vector.h"

using namespace std;

int main()

{

Vector<Money> v(3);

v.Print();

cout << "Min: " << v.Min() << endl;

v.Add();

v.Print();

cout << "Remove pos:";

int pos;

cin >> pos;

v.Del(pos);

v.Print();

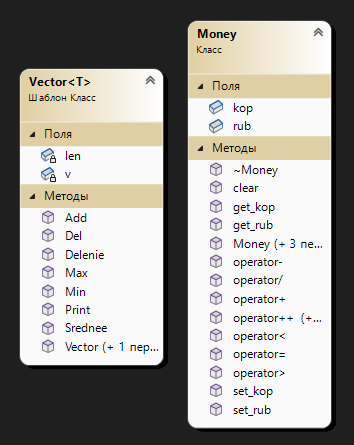
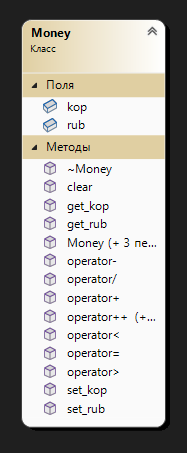
cout << "Summ of min, max and element: " << endl;

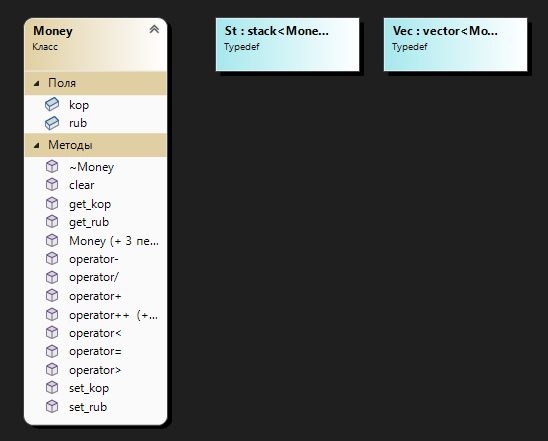
v.Summa();

v.Print();

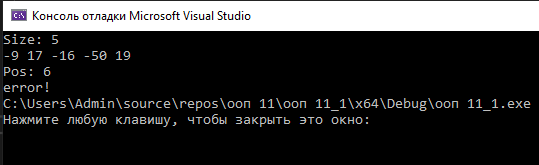
**}**

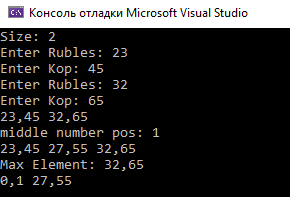
**UML диаграммы**

****

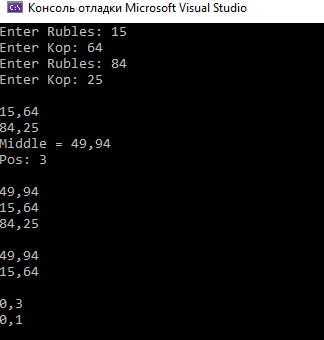
******

**Вывод программы**

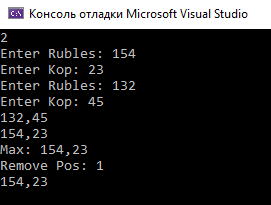
вывод программы в задании под номером 1.

****

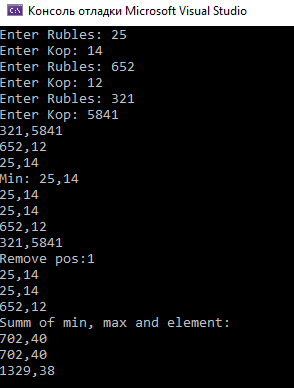
вывод программы в задании под номером 2.



вывод программы в задании под номером 3.

****

вывод программы в задании под номером 4.



вывод программы в задании под номером 5.

**Заключение**

Для решения задачи потребовались знания языка программирования, а конкретнее C++. Программа выполняет те условия, что были указаны в постановке задачи и работает без проблем.