Министерство образования и науки Российской Федерации

Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.13**

Дисциплина: «Информатика»

Тема: Объектно-ориентированное программирование. Стандартные обобщённые алгоритмы STL

Вариант 8

Выполнил:

Студент группы РИС-20-1б

Тараканов Д. М.

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

Пермь

2021 год

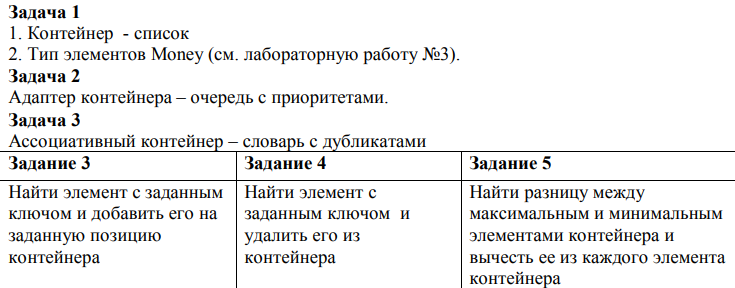
# **Цель работы**

1. Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
2. Разработка программы, использующая обобщённые алгоритмы библиотеки STL в ОО программе.

# **Постановка задачи**

1. Создать последовательный контейнер.
2. Создать ассоциативный контейнер.
3. Создать адаптер контейнер.
4. Заполнить контейнеры элементами (стандартными, пользовательский тип данных)
5. Добавить элементы в соответствии с заданием.
6. Удалить элементы в соответствии с заданием.
7. Написать демонстрационную программу, иллюстрирующую выполнение задачи.

(8 вариант)



# **Анализ задачи**

1. Определить какие действия предстоит выполнить:

* Разработать класс Money который хранит в себе значения рублей и копеек.
* Организовать методы для ввода данных, селекторы и модификатор, конструкторы и деструкторы, перегрузки операторов.

class Money {

long int rubles;

int penny;

public:

Money();

Money(long int, int);

Money(const Money&);

Money& operator = (const Money&);

int get\_rubles();

int get\_penny();

void set\_rubles(long int);

void set\_penny(int);

bool operator == (const Money&);

bool operator > (const Money&);

bool operator < (const Money&);

Money& operator + (int);

friend istream& operator>> (istream& in, Money&);

friend ostream& operator<< (ostream& out, const Money&);

friend fstream& operator>> (fstream& fin, Money&);

friend fstream& operator<< (fstream& fout, const Money&);

~Money();

};

* Разработать глобальные функции для решения задач 1, 2, 3. В пример возьмем глобальные функции из 1 задачи

TList make\_list(int quantity) {

Money el;

TList list;

for (int i = 0; i < quantity; i++) {

cout << i + 1 << ")\n"; cin >> el;

list.push\_back(el);

}

return list;

}

void print\_list(TList list) {

int lenght = list.size();

cout << "\nСписок:\n";

for (int i = 0; i < lenght; i++) {

cout << list.front() << " ";

list.pop\_front();

}

cout << endl;

}

void add\_list(TList& list, const Money element) {

it = find(list.begin(), list.end(), element);

if (it != list.end()) {

int pos;

cout << "\nЭлеменет в списке найден!\n\nПозиция нового элемента в списке > "; cin >> pos;

try {

if (pos > list.size()) {

throw 1.0;

}

TList temp\_list;

int lenght = list.size();

for (int i = 0; i < pos - 1; i++) {

temp\_list.push\_back(list.front());

list.pop\_front();

}

temp\_list.push\_back(element);

for (int i = pos; i < lenght + 1; i++) {

temp\_list.push\_back(list.front());

list.pop\_front();

}

list.clear();

list = temp\_list;

temp\_list.clear();

}

catch (double) {

cout << "\nОшибка индекса\n";

}

print\_list(list);

}

else {

cout << "\nЭлемент в списке не найден!\n";

}

}

void del\_list(TList& list, const Money element) {

it = find(list.begin(), list.end(), element);

if (it != list.end()) {

list.erase(it);

print\_list(list);

}

else {

cout << "\nЭлемент в списке не найден!\n";

}

}

Money max\_list(TList list) {

it = list.begin();

Money max = (\*it);

while (it != list.end()) {

if (max < (\*it)) {

max = (\*it);

}

it++;

}

return max;

}

Money min\_list(TList list) {

it = list.begin();

Money min = (\*it);

while (it != list.end()) {

if (min > (\*it)) {

min = (\*it);

}

it++;

}

return min;

}

void decrease(Money& element) {

element - res;

}

1. C какими типами данных предстояло работать:

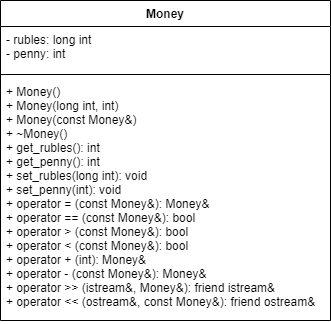
* Для хранения кол-ва рублей реализована переменная rubles типа long int.

long int rubles;

* Для хранения кол-ва копеек реализована переменная penny типа int.

int penny;

# **UML диаграмма**



# **Код программы на языке C++**

Задача №1

#include <C:\Users\apicd\OneDrive\Рабочий стол\Labs\Lab 18 (Class)\Lab 18.12 (Associative STL)\Lab 18.12 (Task 2)\Money.cpp>

#include <list>

#include <iostream>

#include <algorithm>

using namespace std;

typedef list<Money> TList;

TList::iterator it;

Money res;

TList make\_list(int quantity) {

Money el;

TList list;

for (int i = 0; i < quantity; i++) {

cout << i + 1 << ")\n"; cin >> el;

list.push\_back(el);

}

return list;

}

void print\_list(TList list) {

int lenght = list.size();

cout << "\nСписок:\n";

for (int i = 0; i < lenght; i++) {

cout << list.front() << " ";

list.pop\_front();

}

cout << endl;

}

void add\_list(TList& list, const Money element) {

it = find(list.begin(), list.end(), element);

if (it != list.end()) {

int pos;

cout << "\nЭлеменет в списке найден!\n\nПозиция нового элемента в списке > "; cin >> pos;

try {

if (pos > list.size()) {

throw 1.0;

}

TList temp\_list;

int lenght = list.size();

for (int i = 0; i < pos - 1; i++) {

temp\_list.push\_back(list.front());

list.pop\_front();

}

temp\_list.push\_back(element);

for (int i = pos; i < lenght + 1; i++) {

temp\_list.push\_back(list.front());

list.pop\_front();

}

list.clear();

list = temp\_list;

temp\_list.clear();

}

catch (double) {

cout << "\nОшибка индекса\n";

}

print\_list(list);

}

else {

cout << "\nЭлемент в списке не найден!\n";

}

}

void del\_list(TList& list, const Money element) {

it = find(list.begin(), list.end(), element);

if (it != list.end()) {

list.erase(it);

print\_list(list);

}

else {

cout << "\nЭлемент в списке не найден!\n";

}

}

Money max\_list(TList list) {

it = list.begin();

Money max = (\*it);

while (it != list.end()) {

if (max < (\*it)) {

max = (\*it);

}

it++;

}

return max;

}

Money min\_list(TList list) {

it = list.begin();

Money min = (\*it);

while (it != list.end()) {

if (min > (\*it)) {

min = (\*it);

}

it++;

}

return min;

}

void decrease(Money& element) {

element - res;

}

void main() {

system("chcp 1251");

int quantity;

cout << "\nCоздание и заполнение списка\n\nКол-во элементов в списке > ";

do {

cin >> quantity;

if (quantity < 1) {

cout << "\nВведите положительное кол-во элемнетов > ";

}

} while (quantity < 1);

TList list;

list = make\_list(quantity);

print\_list(list);

Money key;

cout << "\nДобавление существующего элемента списка\n\nИскомый элемент >\n"; cin >> key;

add\_list(list, key);

cout << "\nУдаление элементов из списка\n\nИскомый элемент >\n"; cin >> key;

del\_list(list, key);

cout << "\nУменьшение всех элеменетов списка\n";

Money max = max\_list(list), min = min\_list(list);

res = max;

res - min;

cout << "\nMax элемент списка = " << max << "\nMin элемент списка = " << min << "\nРазность Max и Min = " << res << endl;

for\_each(list.begin(), list.end(), decrease);

print\_list(list);

system("pause");

}

Задача №3

#include <C:\Users\apicd\OneDrive\Рабочий стол\Labs\Lab 18 (Class)\Lab 18.12 (Associative STL)\Lab 18.12 (Task 2)\Money.cpp>

#include <iostream>

#include <map>

using namespace std;

typedef multimap<int, Money> TMap;

TMap::iterator it;

Money res;

TMap make\_map(int quantity) {

TMap map;

for (int i = 0; i < quantity; i++) {

Money el;

cout << endl << i + 1 << ")\n"; cin >> el;

map.emplace(i+1, el);

}

return map;

}

void print\_map(TMap map) {

cout << "\nСловарь:\n"; cout << endl;

it = map.begin();

while(it != map.end()) {

cout << it->first << " : " << it->second << " " << endl;

it++;

}

}

void add\_map(TMap& map, const Money element) {

bool flag = true;

it = map.begin();

if (it->second == element) {

flag = false;

}

while (flag) {

it++;

if (it->second == element) {

flag = false;

}

}

if (it != map.end()) {

int pos;

cout << "\nЭлеменет в словаре найден!\n\nПозиция нового элемента в словаре > "; cin >> pos;

try {

if (pos > map.size()) {

throw 1.0;

}

else {

TMap temp\_map;

it = map.begin();

for (int i = 0; i < pos - 1; i++) {

temp\_map.emplace(i+1, it->second);

it++;

}

temp\_map.emplace(pos - 1, element);

for (int i = pos; i < map.size() + 1; i++) {

temp\_map.emplace(i+1, it->second);

it++;

}

map.clear();

map = temp\_map;

temp\_map.clear();

}

}

catch (double) {

cout << "\nОшибка индекса\n";

}

print\_map(map);

}

else {

cout << "\nЭлемент в списке не найден!\n";

}

}

void del\_map(TMap& map, const Money element) {

bool flag = true;

it = map.begin();

if (it->second == element) {

flag = false;

}

while (flag) {

it++;

if (it->second == element) {

flag = false;

}

}

if (it != map.end()) {

map.erase(it);

print\_map(map);

}

else {

cout << "\nЭлемент в списке не найден!\n";

}

}

Money max\_map(TMap map) {

it = map.begin();

Money max = (\*it).second;

while (it != map.end()) {

if (max < (\*it).second) {

max = (\*it).second;

}

it++;

}

return max;

}

Money min\_map(TMap map) {

it = map.begin();

Money min = (\*it).second;

while (it != map.end()) {

if (min > (\*it).second) {

min = (\*it).second;

}

it++;

}

return min;

}

void decrease\_map(TMap& map) {

it = map.begin();

while (it != map.end()) {

(\*it).second - res;

it++;

}

}

void main() {

system("chcp 1251");

int quantity;

cout << "\nCоздание и заполнение словаря\n\nКол-во элементов в словаре > ";

do {

cin >> quantity;

if (quantity < 1) {

cout << "\nВведите положительное кол-во элемнетов > ";

}

} while (quantity < 1);

TMap map;

map = make\_map(quantity);

print\_map(map);

Money key;

cout << "\nДобавление существующего элемента списка\n\nИскомый элемент >\n"; cin >> key;

add\_map(map, key);

cout << "\nУдаление элементов из списка\n\nИскомый элемент >\n"; cin >> key;

del\_map(map, key);

cout << "\nУменьшение всех элеменетов списка\n";

Money max = max\_map(map), min = min\_map(map);

res = max;

res - min;

cout << "\nMax элемент списка = " << max << "\nMin элемент списка = " << min << "\nРазность Max и Min = " << res << endl;

decrease\_map(map);

print\_map(map);

system("pause");

}

# **Скриншоты тестов**

