Министерство высшего образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

Тема: «Лабораторная работа №11»

Выполнил

Студент группы РИС-22-2б

Прядеин И.А.

Проверил доцент кафедры

ИТАС

Полякова О. А.

Пермь 2023

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.

**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.

**Задача 3.**

1. Создать параметризованный 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 параметризованного класса.

**Задача 4.**

1. Создать адаптер контейнера.
2. Заполнить его элементами пользовательского класса (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.

**Задача 5.**

1. Создать параметризованный 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.

**Вариант 8:**

**Задача 1**

1. Контейнер - список
2. Тип элементов - double

**Задача 2**

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

**Задача 3**

Параметризованный класс - Вектор (см. лабораторную работу №7)

**Задача 4**

Адаптер контейнера - очередь с приоритетами.

**Задача 5**

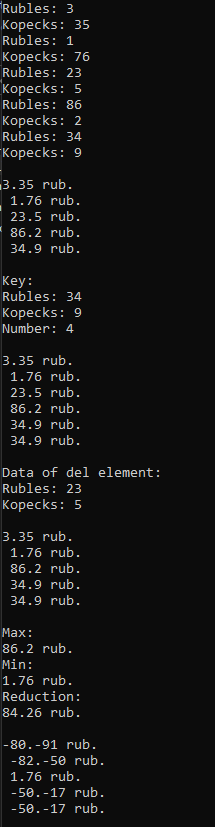
Параметризованный класс - Вектор

Адаптер контейнера - очередь с приоритетами.

|  |  |  |
| --- | --- | --- |
| **Задание 3** | **Задание 4** | **Задание 5** |
| Найти элемент с заданным ключом и добавить его на заданную позицию контейнера | Найти элемент с заданным ключом и удалить его из контейнера | Найти разницу между максимальным и минимальным элементами контейнера и вычесть её из каждого элемента контейнера |

**UML диаграмма:**

**Скриншот результата выполнения программы**



**Ответы на вопросы:**

1. Из каких частей состоит библиотека STL?

**Ответ:** STL - Standart Template Library, стандартная библиотека шаблонов состоит из двух основных частей: набора контейнерных классов и набора обобщённых алгоритмов.

1. Какие типы контейнеров существуют в STL?

**Ответ:** Контейнеры STL можно разделить на два типа: последовательные и ассоциативные.

1. Что нужно сделать для использования контейнера STL в своей программе?

**Ответ:** Для использования контейнера в программе необходимо включить в неё соответствующий заголовочный файл. Тип объектов, сохраняемых в контейнере, задаётся с помощью аргумента шаблона, например:

#include <vector>

#include <list>

#include “Person.h”

...

vector<int> v;

list<Person> l;

1. Что представляет собой итератор?

**Ответ:** Итератор - это обобщение концепции указателей: они ссылаются на элементы контейнера.

1. Какие операции можно выполнять над итераторами?

**Ответ:** К основным операциям, выполняемым с любыми итераторами, относятся:

* Разыменование итератора: если p - итератор, то \*p - значение объекта, на который он ссылается.
* Присваивание одного итератор другому.
* Сравнение итераторов на равенство и неравенство (== и !=).
* Перемещение его по всем элементам контейнера с помощью префиксного (++p) или постфиксного (p++) инкремента.

1. Каким образом можно организовать цикл для перебора контейнера с использованием итератора?

**Ответ:** for (i = first; i != last; ++i);

1. Какие типы итераторов существуют?

**Ответ:** В STL существуют следующие типы итераторов:

* входные,
* выходные,
* прямые,
* двунаправленные итераторы,
* итераторы произвольного доступа.

1. Перечислить операции и методы, общие для всех контейнеров.

**Ответ:** Операции и методы, общие для всех контейнеров:

* Операции равенства и неравенства
* Операция присваивания
* clear - Удаляет все объекты
* insert - Добавляет один элемент или диапазон элементов
* erase - Удаляет один элемент или диапазон элементов
* size\_type size() const - Возвращает число элементов
* size\_type max\_size() const - Возвращает максимально допустимый размер контейнера
* bool empty() const - Возвращает true, если контейнер пуст
* iterator begin() - Возвращает итератор на начало контейнера
* iterator end() - Возвращает итератор на конец контейнера
* reverse\_iterator begin() - Возвращает реверсивный итератор на конец контейнера
* reverse\_iterator end() - Возвращает реверсивный итератор на начало контейнера

1. Какие операции являются эффективными для контейнера vector? Почему?

**Ответ:** Контейнер vector эффективно обрабатывает произвольную выборку элементов с помощью операции индексации [] или метода at.

1. Какие операции являются эффективными для контейнера list? Почему?

**Ответ:** Вставка и удаление работают эффективно для любой позиции элемента в списке.

1. Какие операции являются эффективными для контейнера deque? Почему?

**Ответ:** Контейнер deque эффективно поддерживает вставку и удаление первого элемента (так же, как и последнего).

1. Перечислить методы, которые поддерживает последовательный контейнер vector.

**Ответ:** Методы, которые поддерживает vector:

* push\_back() - Добавление в конец
* pop\_back() - Удаление из конца
* insert - Вставка в произвольное место
* erase - Удаление из произвольного места
* [], at - Доступ к произвольному элементу
* swap - Обмен векторов
* clear() - Очистить вектор

1. Перечислить методы, которые поддерживает последовательный контейнер list.

**Ответ:** Методы, которые поддерживает list:

* push\_back(T& key) - Добавление в конец
* pop\_back() - Удаление из конца
* push\_front(T& key) - Добавление в начало
* pop\_front() - Удаление из начала
* insert - Вставка в произвольное место
* erase - Удаление из произвольного места
* swap - Обмен списков
* clear() - Очистить вектор
* splice - Сцепка списка

1. Перечислить методы, которые поддерживает последовательный контейнер deque.

**Ответ:** Методы, которые поддерживает deque:

* push\_back(T& key) - Добавление в конец
* pop\_back() - Удаление из конца
* push\_front(T& key) - Добавление в начало
* pop\_front() - Удаление из начала
* insert - Вставка в произвольное место
* erase - Удаление из произвольного места
* [], at - Доступ к произвольному элементу

1. Задан контейнер vector. Как удалить из него элементы с 2 по 5?

**Ответ:**

vector<int> v;

vector<int>::iterator first = v.begin();

vector<int>::iterator last = v.end();

advance(first, 2);

advance(last, 5);

v.erase(first, last);

1. Задан контейнер vector. Как удалить из него последний элемент?

**Ответ:**

vector<int> v;

v.pop\_back();

1. Задан контейнер list. Как удалить из него элементы с 2 по 5?

**Ответ:**

list<int> L;

list<int>::iterator first = L.begin();

list<int>::iterator last = L.end();

advance(first, 2);

advance(last, 5);

L.erase(first, last);

1. Задан контейнер list. Как удалить из него последний элемент?

**Ответ:**

list<int> L;

L.pop\_back();

1. Задан контейнер deque. Как удалить из него элементы со 2 по 5?

**Ответ:**

deque<int> d;

deque<int>::iterator first = d.begin();

deque<int>::iterator last = d.end();

advance(first, 2);

advance(last, 5);

d.erase(first, last);

1. Задан контейнер deque. Как удалить из него последний элемент?

**Ответ:**

deque<int> d;

d.pop\_back();

1. Написать функцию для печати последовательного контейнера с использованием итератора.

**Ответ:**

void Print(vector v) {

vector<int>::iterator it;

for (it = v.begin(); it != v.end(); it++)

cout << \*it << “ “;

cout << “\n”;

}

1. Что представляет собой адаптеры контейнеров?

**Ответ:** Специализированные последовательные контейнеры - стек, очередь и очередь с приоритетами - не являются самостоятельными контейнерными классами, а реализованны на основе рассмотренных выше классов, поэтому они называются адаптерами контейнеров.

1. Чем отличаются друг от друга объявления stack<int>s и stack<int, list<int>> s?

**Ответ:** Объявление stack<int> s создаёт стек на базе двусторонней очереди.

Смысли же реализации stack<int, list<int>> s заключается в том, что специализированный класс просто переопределяет интерфейс класса-прототипа, ограничивая его только теми методами, которые нужны новому классу.

1. Перечислить методы, которые поддерживает контейнер stack.

**Ответ:** Методы класса stack:

* push() - добавление в конец;
* pop() - удаление из конца:
* top() - получение текущего элемента стека;
* empty() - проверка пустой стек или нет;
* size() - получение размера стека.

1. Перечислить методы, которые поддерживает контейнер queue.

**Ответ:** Методы класса queue:

* push() - добавление в конец очереди;
* pop() - удаление из начала очереди;
* front() - получение первого элемента очереди;
* back() - получение последнего элемента очереди;
* empty() - проверка пустая очередь или нет;
* size() - получение размера очереди.

1. Чем отличается друг от друга контейнеры queue и priority\_queue?

**Ответ:** Очередь с приоритетами отличается от обычной очереди тем, что для извлечения выбирается максимальный элемент из хранимых в контейнере.

1. Задан контейнер stack. Как удалить из него элемент с заданным номером?

**Ответ:**

void del\_num(stack<int>& s, int index) {

stack<int> temp;

for (int I = 0; I < index; I++) {

temp.push(s.top());

s.pop();

}

s.pop();

while (temp.size()) {

s.push(temp.top());

temp.pop();

}

}

1. Задан контейнер queue. Как удалить из него элемент с заданным номером?

**Ответ:**

queue<int> q{ 2, 3, 4, 6, 1 };

int index;

cout << “Index: “;

cin >> index;

queue<int>::iterator it(q.begin(), index);

q.erase(it);

1. Написать функцию для печати контейнера stack с использованием итератора.

**Ответ:**

1. Написать функцию для печати контейнера queue с использованием итератора.

**Ответ:**

void printQueue(queue<int> q) {

for (queue<int>::iterator it = q.begin(); it != q.end(); it++)

cout << \*it << endl;

}