МИНОБРНАУКИ РФ

ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»

Кафедра «Программное обеспечение»

Отчет

по лабораторной работе № 3

по объектно-ориентированному программированию

Вариант №1

Выполнил студент группы Б04-191-3: Р.А. Гумметов

Принял: К.С. Чернышев

Ижевск 2020

Постановка задачи

Используя шаблоны и исключительные ситуации реализовать класс контейнер очередь (FIFO).

Текст программы

#include <iostream>

#include <string>

using namespace std;

template <class T>

class Element

{

public:

T Value;

Element<T>\* Next;

Element(T value)

{

Value = value;

Next = NULL;

}

};

template <class T>

class Queue

{

Element<T>\* Head;

Element<T>\* Tail;

public:

Queue()

{

Head = NULL;

Tail = NULL;

}

void Enqueue(T value)

{

if (Head == NULL)

{

Element<T>\* el = new Element<T>(value);

Head = Tail = new Element<T>(value);

}

else

{

Tail->Next = new Element<T>(value);

Tail = Tail->Next;

}

}

T Dequeue()

{

try

{

if (Head == NULL)

throw exception("Queue is empty");

T value = Head->Value;

if (Head != Tail)

{

Element<T>\* temp = Head->Next;

delete Head;

Head = temp;

}

else

{

delete Head;

Head = NULL;

Tail = NULL;

}

return value;

}

catch (const exception & ex)

{

cout << ex.what() << endl;

}

}

};

int main()

{

Queue<int> queueInt;

queueInt.Enqueue(10);

queueInt.Enqueue(18);

queueInt.Enqueue(23);

cout << queueInt.Dequeue() << endl;

cout << queueInt.Dequeue() << endl;

cout << queueInt.Dequeue() << endl;

Queue<string> queueString;

queueString.Enqueue("string 1");

queueString.Enqueue("string 2");

cout << queueString.Dequeue() << endl;

cout << queueString.Dequeue() << endl;

}

Пример работы программы

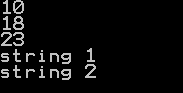


Рис. 1

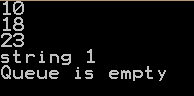


Рис. 2

Вопросы:

1)  Что такое шаблоны, и где они применяются?

Шаблон - повторяемая [архитектурная конструкция](https://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F), представляющая собой решение проблемы [проектирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) в рамках некоторого часто возникающего [контекста](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D0%B5%D0%BA%D1%81%D1%82). Шаблоны применяются в задачах, которые часто возникают ходе проектирования объектно-ориентированных программ.

2) Зачем нужны исключительные ситуации?

Обработка исключений обеспечивает механизм, позволяющий отделить обработку ошибок или других исключительных обстоятельств от общего потока выполнения кода.

3) Какие из принципов ООП используются в данной работе?

В данной работе используются такие принципы ООП, как инкапсуляция и абстракция.

4)  В чем отличие динамического и статического полиморфизма?

В случае статического полиморфизма адрес метода, который вызывается определяется при компиляции, а в случае динамического - в рантайме.