Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет информационных технологий и управления

Кафедра экономической информатики

Дисциплина: Объектно-ориентированное программирование

Контрольная работа №3

Студент: гр. 724402 Чернявский Я.А.

Руководитель: Сторожев Д.А.

Минск 2019

**Задание:** Реализовать шаблон класса Circle, реализующий однонаправленное кольцо. Для представления элемента кольца использовать шаблон структуры Node. Определить функции добавления элемента в кольцо и поиска в кольце.

Листинг кода:

#include <iostream>

using namespace std;

template <class T>

class Circle

{

struct Node

{

T data;

Node\* next;

};

protected:

Node\* last; // указатель на последний элемент

public:

Circle(); // конструктор по умолчанию

Circle(T); // конструктор с параметрами

Circle(const Circle&); // конструктор копирования

~Circle(); // деструктор

void addItem(T);

void print() const;

void find(T) const;

};

template <class T>

Circle <T>::Circle() :last(NULL)

{

// кольцо не содержит элементов

}

template <class T>

Circle<T>::Circle(T data)

{

last = new Node;

last->data = data;

last->next = last;

}

template <class T>

Circle<T>::Circle(const Circle& circle)

{

last = NULL;

if (circle.last != NULL)

{

Node\* cur = circle.last->next;

while (cur != circle.last) // проходим элементы пока не дойдем до последнего

{

this->addItem(cur->data);

cur = cur->next;

}

// копирование последнего элемента

this->addItem(circle.last->data);

}

}

template <class T>

Circle<T>::~Circle()

{

Node\* temp, \* cur;

// удаление элементов и освобождение ресурсов

if (last != NULL) // если кольцо содержит элементы

{

cur = last->next;

last->next = NULL; // разрываем кольцо

while (cur)

{

temp = cur;

cur = cur->next;

delete temp;

}

}

}

template <class T>

void Circle<T>::addItem(T data)

{

// если добавляется певый элемент

if (last == NULL)

{

last = new Node;

last->next = last;

last->data = data;

}

else // добавление второго и последующих элементов

{

Node\* oldLast = last;

Node\* newNode = new Node;

newNode->data = data;

newNode->next = last->next;

last = newNode;

oldLast->next = newNode;

}

}

template <class T>

void Circle<T>::print() const

{

if (last != NULL) // если кольцо не пустое

{

Node\* cur = last->next;

while (cur != last) // выводим элементы пока не дойдем до последнего

{

std::cout << cur->data << " ";

cur = cur->next;

}

// вывод последнего элемента

std::cout << last->data << endl;

}

}

template <class T>

void Circle<T>::find(T pattern) const

{

int c = 0;

if (last != NULL) // если кольцо не пустое

{

Node\* cur = last->next;

while (cur != last) // перебираем элементы пока не дойдем до последнего

{

if (cur->data == pattern)

std::cout << "Совпадение найдено в " << c << " элементе кольца\n";

cur = cur->next;

c++;

}

// проверяем последнй элемент

if (last->data == pattern)

std::cout << "Совпадение найдено в " << c << " элементе кольца\n";

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

Circle <int> c1;

c1.addItem(15);

c1.addItem(30);

c1.addItem(15);

c1.addItem(7);

std::cout << "Вывод элементов кольца c1: ";

c1.print();

// создание объекта с2 с использованием конструктора копирования

Circle <int> c2 = c1;

c2.addItem(99);

c2.addItem(15);

std::cout << "Вывод элементов кольца c2: ";

c2.print();

std::cout << "Поиск элементов кольца с2, содержащих число 15:\n";

c2.find(15);

return 0;

}

Пример работы:

