

Практическая работа № 4

Связный список

Цель работы: Изучить способы применения связного списка

Оборудование: Windows 10, Visual Studio, Microsoft Word.

1. КРАТКАЯ ТЕОРИЯ

Основное назначение связного списка — предоставление механизма для хранения и доступа к произвольному количеству данных. Как следует из названия, это достигается связыванием данных вместе в список.

Прежде чем мы перейдем к рассмотрению связного списка, давайте вспомним, как хранятся данные в массиве.

данные в массиве хранятся в непрерывном участке памяти, разделенном на ячейки определенного размера. Доступ к данным в ячейках осуществляется по ссылке на их расположение — индексу.

Это отличный способ хранить данные. Большинство языков программирования позволяют так или иначе выделить память в виде массива и оперировать его содержимым. Последовательное хранение данных увеличивает производительность (*data locality*), позволяет легко итерироваться по содержимому и получать доступ к произвольному элементу по индексу.

ем не менее, иногда массив — не самая подходящая структура.

Предположим, что у нашей программы следующие требования:

- Прочитать некоторое количество целых чисел из источника (метод `NextValue`), пока не встретится число `0xFFFF`.
- Передать считанные числа в метод `ProcessItems`

Поскольку в требованиях указано, что считанные числа передаются в метод

					АиСД.09.03.02.110000.0000ПР			
Изм	Лист	№ докум.	Подпись	Дата				
Разраб.		Куличенко Е.В.			Практическая работа №4 Связный список ..	Лит	Лист	Листов
Провер.		Береза А.Н.					1	
Н.контр.						ИСОuП(ф) ДГТУ ИСТ-Tb21		
Утв.								

Исходный код

```
#include <iostream>

using namespace std;
struct Node
{
    int data;
    Node* next;
};

class List
{
private:
    Node* head; //"голова" связанного списка

public:
    List() //конструктор класса без параметров
    {
        head = NULL; //первого элемента пока нет
    }

    //метод, добавляющий новый узел в список
    void addNode(int d)
    {
        Node* nd = new Node; //динамически создаем новый узел

        nd->data = d;           //задаем узлу данные
        nd->next = NULL;        //новый узел в конце, поэтому NULL

        if (head == NULL)      //если создаем первый узел
            head = nd;
        else                    //если узел уже не первый
        {
            Node* current = head;

            //ищем в цикле предшествующий последнему узел
            while (current->next != NULL)
                current = current->next;

            //предшествующий указывает на последний
            current->next = nd;
        }
    }

    //метод, выводящий связанный список на экран
    void printList()
    {
        Node* current = head;

        while (current != NULL)
        {
            cout << current->data << endl;
            current = current->next;
        }
    }
};

int main()
{
    List myList;

    myList.addNode(5);
}
```

					АиСД.09.03.02.110000.0000ПР	Лист
						2
Изм	Лист	№ докум.	Подпись	Дата		

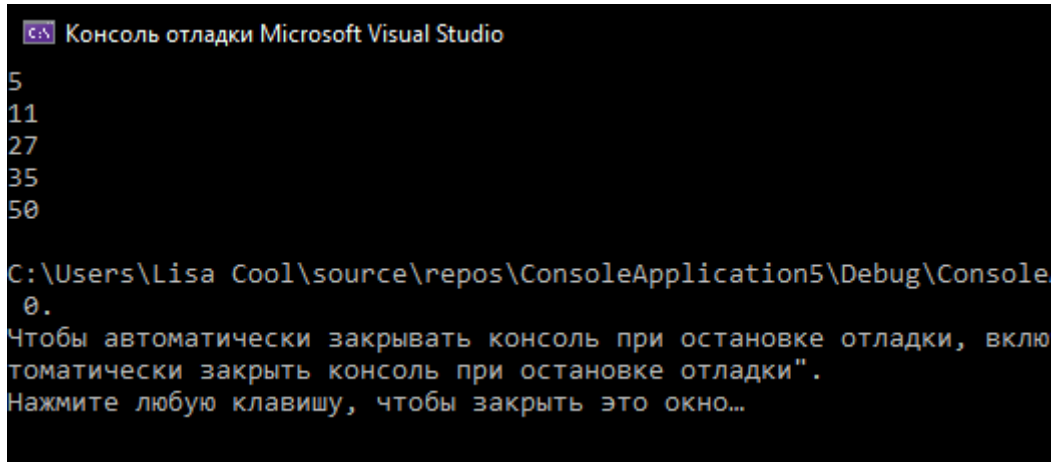
```

myList.addNode(11);
myList.addNode(27);
myList.addNode(35);
myList.addNode(50);

myList.printList();

return 0;
}

```



```

C:\> Консоль отладки Microsoft Visual Studio

5
11
27
35
50

C:\Users\Lisa Cool\source\repos\ConsoleApplication5\Debug\ConsoleApplication5.exe
0.
Чтобы автоматически закрывать консоль при остановке отладки, включите опцию "Оформить вывод в консоль" в меню "Отладка".
Нажмите любую клавишу, чтобы закрыть это окно...

```

					АиСД.09.03.02.110000.0000ПР	Лист
						3
Изм	Лист	№ докум.	Подпись	Дата		