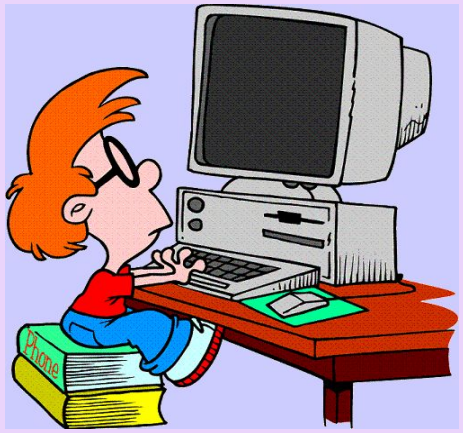
Информатика
Массивы и списки

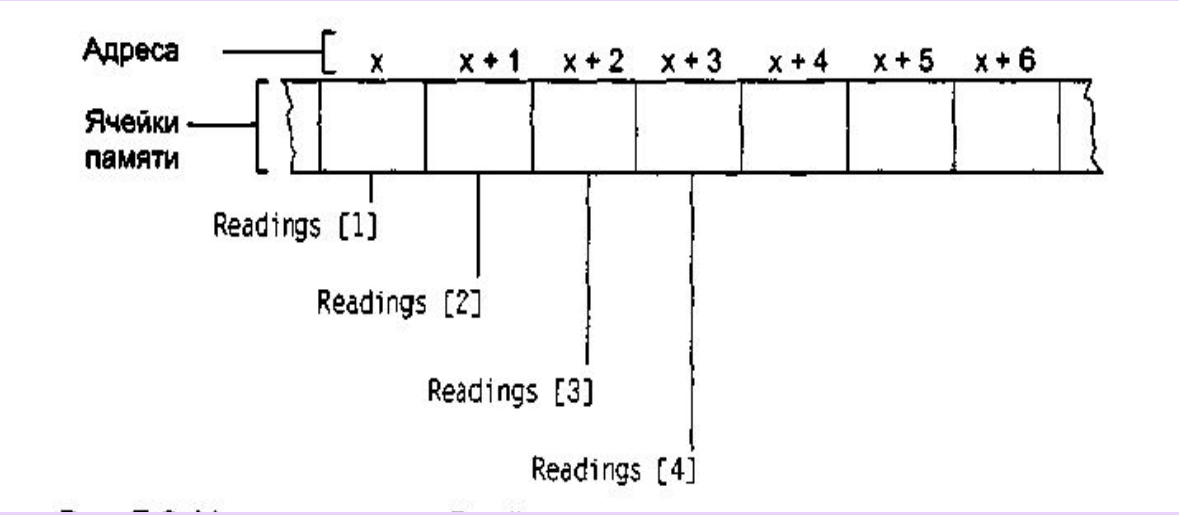
Массив - это упорядоченный набор данных. Каждый элемент массива имеет индекс и ключ. Индекс (ключ) служит для однозначной идентификации элемента внутри массива. В одном массиве не может быть двух элементов с одинаковыми индексами.

Массив можно представить как переменную, содержащую упорядоченный набор данных одного типа. К каждому элементу массива можно получить доступ по его адресу. В языках C/C++ массив не является стандартным типом данных. Напротив, он сам имеет тип: char, int, float, doubleи т.д. Допускается создавать массивы массивов, указателей, структур и др. Принципы построения массивов и работы с ними в основе своей одинаковы в C и C++.



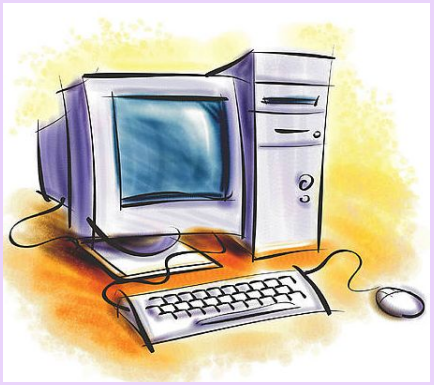
Свойства массивов

• все элементы массива должны быть одного типа;
• все элементы массива сохраняются в памяти последовательно, и первый элемент имеет нулевое смещение адреса, т.е. нулевой индекс;
• имя массива является константой и содержит адрес первого элемента массива.



Объявление массива имеет два формата:

1) спецификатор_типа описатель [константное_выражение];
2) спецификатор_типа описатель [ ];
Описатель - это идентификатор массива.
Спецификатор_типа задает тип элементов объявляемого массива. Элементами массива не могут быть функции и элементы типа void.
Константное_выражение в квадратных скобках задает количество элементов массива.



Инициализация массивов (присваивание начальных значений их элементам)

Можно явно не указывать количество элементов одномерного массива, а только перечислить их начальные значения в списке инициализации:
double d[ ]={1.0, 2.0, 3.0, 4.0, 5.0};
Если в определении массива явно указан его размер, то коли чество начальных значений не может быть больше количества элементов в массиве. Если количество начальных значений меньше, чем объявленная длина массива, то начальные значения получат только первые элементы массива.
Присвоить начальные значения элементам двумерного массива A, состоящего из трех "строк" и двух "столбцов", можно следующим образом:
double A[3][2]={{10,20), {30,40}, {50,60}};
Инициализаци двумерного массива:
double A[3][2]={10,20,30,40,50,60};

Динамические массивы

Динамические массивы создают с помощью операции распределения памяти (new, malloc и т.п.), при этом необходимо указать тип и размерность, например:

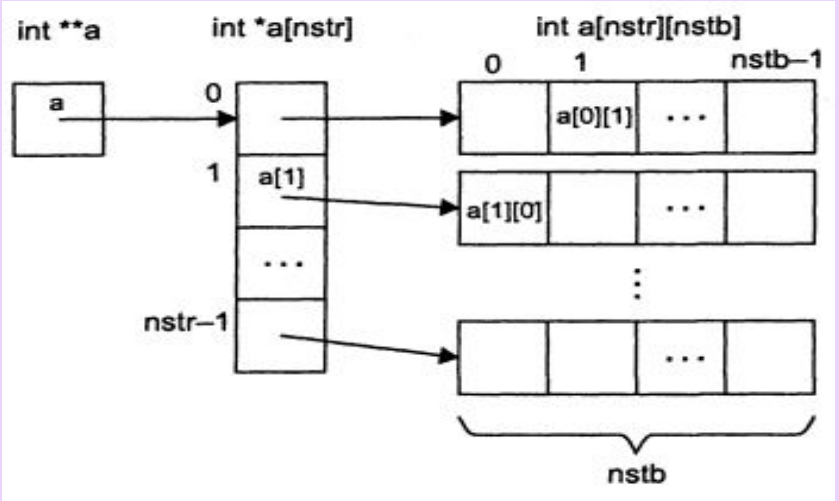
int n = 100;

float *p = new float [n];

Память, зарезервированная под динамический массив с помощью new [], должна освобождаться оператором delete [], а память, выделенная функцией malloc – посредством функции free, например:

delete [ ] p;

free (q);



Выделение памяти под двухмерный массив

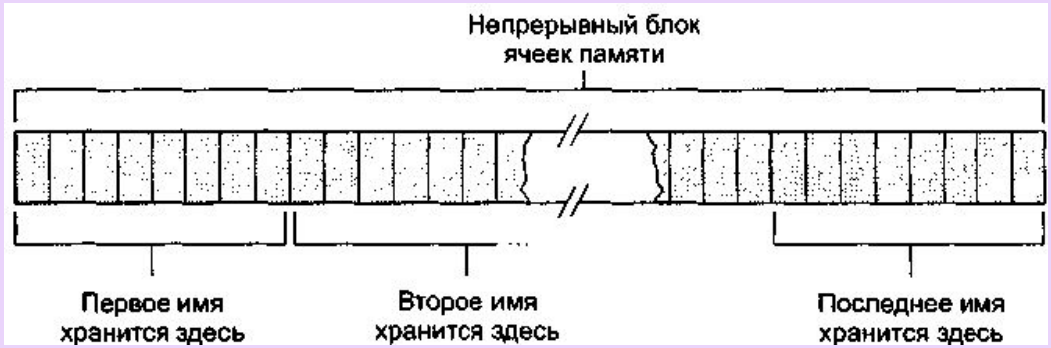
Списки

Список — это набор записей, выстроенных в определенной последовательности. Примерами могут служить список учащихся класса, список дел на день или словарь.

Непрерывные списки

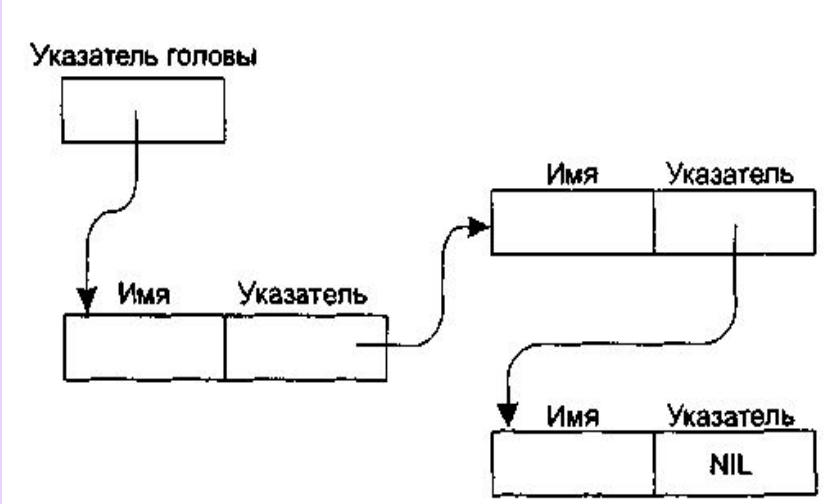
Непрерывный список — это удобная структура хранения для реализации статических списков, но у него есть недостатки, делающие его не слишком подходящим для динамических случаев.

Если все элементы списка принадлежат одному примитивному типу данных, тогда список — это не что иное, как одномерный однородный массив.



Связные списки

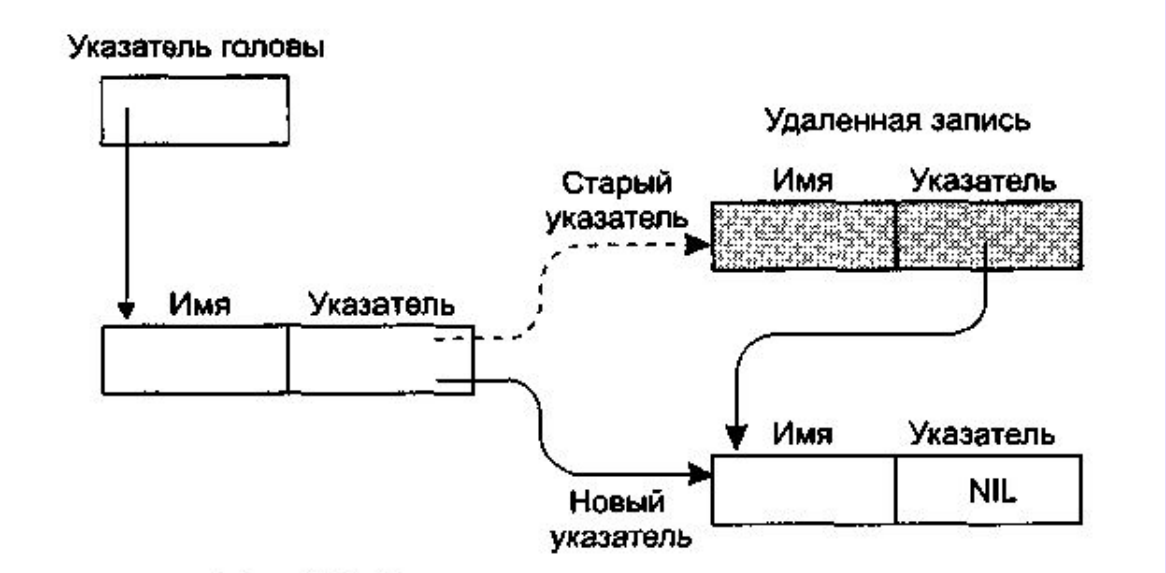
Связные списки - это структура данных, состоящая из узлов, каждый из которых содержит как собственные данные, так и одну или две ссылки («связки») на следующий или предыдущий узел списка.
Для того чтобы определить начало связного списка, отдельно определяется еще один указатель, в котором хранится адрес первой записи. Так как этот указатель указывает на начало, или голову списка, он называется указателем головы (head pointer).
Для определения конца связного списка мы используем пустой указатель (NIL pointer), который является просто определенного вида набором битов и находится в последней записи, указывая, что за ним в списке более нет элементов.



Структура связного списка

Удаление записи из связного списка

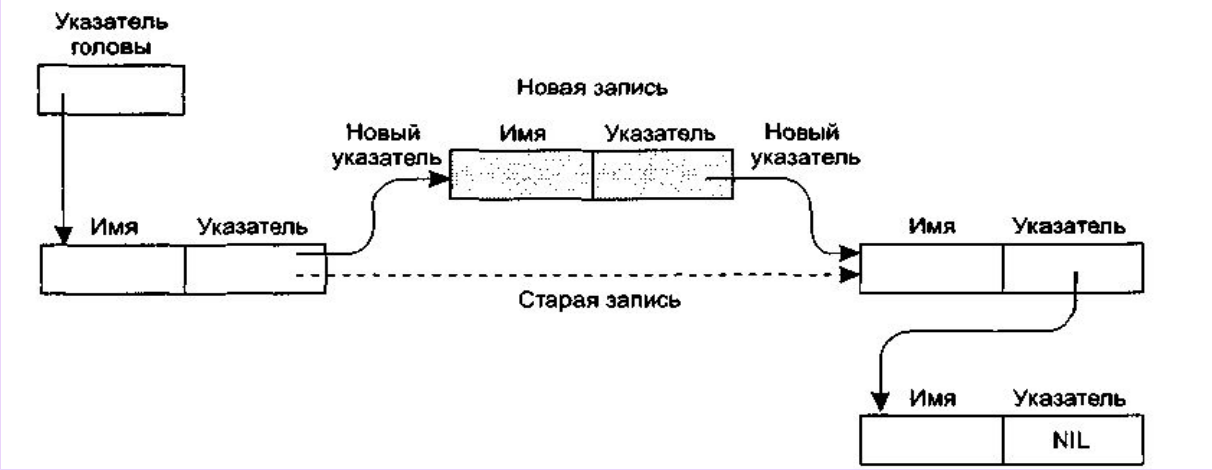
Удалить имя можно, изменив один указатель. Это означает, что указатель, который ранее указывал на удаленное имя, изменяется и указывает теперь на имя, следующее за удаленным. И тогда при прохождении списка удаленное имя пропускается, поскольку не является более элементом цепочки.



Удаление записи из связного списка

Добавление нового имени в связный список

Сначала находим неиспользуемый блок из девяти ячеек памяти, записываем новое имя в первые восемь ячеек, а в девятую записываем адрес имени, которое должно следовать в списке за новым. Затем изменяем указатель, связанный с именем, которое должно предшествовать новому имени в списке так, чтобы он указывал на этот новый блок. После этого новую запись можно будет найти на нужном месте при прохождении списка.



Добавление записи в связный список