Лекция 1.2. Массивы и списки

Добрый день. В этой лекции мы рассмотрим основные способы организации структур данных в памяти компьютера. Они известны всем, кто хоть немного умеет программировать, но мы хотим посмотреть на них с точки зрения эффективности использования в разных ситуациях.

К числу этих способов относятся следующие три:

- структуры примыкания, то есть обычные записи;
- векторные структуры (они же – массивы), в которых однотипные элементы располагаются в ряд, и адрес каждого элемента легко вычислить, зная адрес начала массива, размер элемента и его номер;
- сцепленные структуры: списки, деревья, графы, сети и т.п. В таких структурах с каждым элементом может быть связана одна или несколько ссылок (указателей) на элементы, следующие за ним. Чтобы получить доступ к конкретному элементу, нужно пройти к нему по цепочке ссылок от начального элемента структуры.

Комбинируя эти три типа, можно построить структуры данных произвольной сложности.

Начнем с примыкания.

Примыкающие друг к другу элементы могут быть разных типов и размеров.

В C и C++ конструкция примыкания называется struct, в Паскале – record, то есть «запись». Поля записи имеют имена, с которыми компилятор связывает значения смещения каждого поля от начала записи. Использование записей обычно не вызывает затруднений. Они объединяют в единое целое набор разнородных характеристик некоторого типа объектов, иначе говоря – используются для группировки различной информации об объекте.

Как практически во всех языках программирования, в C можно описать массив переменных.

Массив – это структура, объединяющая однотипные элементы, расположенные в памяти последовательно. Главное и очень существенное достоинство массивов – возможность быстрого доступа к любому его элементу по известному индексу (или нескольким индексам, если массив многомерный), причем значение индекса может вычисляться в ходе работы программы.

При вычислении адреса элемента одномерного массива программа прибавляет к адресу начала массива значение смещения, равное размеру одного элемента, умноженному на индекс элемента. Это одна операция умножения и одно сложение. Для двумерного массива как нетрудно увидеть, вычисление адреса элемента требует двух умножений и двух сложений.

Классический вид массивов, присутствующий во всех версиях C и C++, это статический массив. Размер такого массива должен задаваться константной величиной при написании программы и не может изменяться в ходе работы программы. Константа может записываться либо непосредственно как целое число, либо как именованная константа, определенная либо директивой #define, либо, в более поздних версиях языка,

с помощью модификатора const в описании целочисленной переменной. Второй способ считается более корректным.

Статические массивы не лишены существенных недостатков:

- поскольку размер массива задается при написании программы, его приходится задавать «по максимуму», в расчете на наибольшее возможное значение размера в конкретных задачах;

- если в начале или в середине массива нужно вставить или, наоборот, удалить элемент, то приходится перемещать на одну позицию все элементы с большими индексами. Эта простая операция требует затраты времени, пропорциональной размеру массива, и для массива размером в несколько мегабайт это может быть весьма существенная задержка.

В C++ можно использовать динамические массивы, создаваемые с помощью оператора new. Размер массива при этом определяется в момент его создания, а не при написания программы. По истечении надобности такой массив можно удалить с помощью оператора delete, освободив занимаемую им память.

По сравнению со статическим массивом:

- определение размера массива в ходе работы программы является несомненным достоинством,

- однако остальные недостатки всех массивов сохраняются (нельзя изменить размер после создания, вставка и удаление элементов требуют существенных затрат времени).

Что мы узнали?

Что известны три способа представления структур данных в памяти компьютера: примыкание (записи), векторное размещение (массивы), сцепление (списки).

Что записи используются для группировки различной информации о едином объекте.

Что основным преимуществом массивов является быстрый доступ по индексу, а главными недостатками – необходимость определять размер массива при написании программы и трудоемкость операций вставки и удаления элементов в начале или в середине массива.

На этом лекция закончена, благодарю за внимание.