Лекция 1.2. Типы структур. Списки

Добрый день. В этой лекции мы рассмотрим сцепленные списки – структуры данных, альтернативные массивам.

Простейшими из сцепленных структур данных являются односвязные линейные списки. Каждый элемент такого списка, помимо данных, содержит указатель на следующий элемент. Последний элемент списка (называемый обычно хвостом, или по-английски tail) содержит специальное нулевое значение указателя. Доступ к элементам списка возможен только от его первого элемента (головы, head) путем прохода по цепочке указателей.

Перед созданием списка надо описать структуру его элементов, каждый из которых состоит из указателя на следующий элемент next и поля данных data. Первоначально список пуст, указатель на его голову (list) содержит нулевое значение.

Добавлять элементы проще всего в голову списка. Сначала надо создать новый элемент elem с помощью оператора new и задать значение данных этого элемента (в примере – целое число 1). Затем, поскольку новый элемент должен стать первым в списке, в его поле next заносится указатель на текущий первый элемент list, который теперь становится вторым в списке. А указатель list (голова списка) теперь будет указывать на новый вставленный элемент.

В зависимости от того, какие операции планируется выполнять со списком, используют разные варианты списков.

Список с указателями не только на голову, но и на хвост используется, если требуется часто добавлять элементы в хвост списка. Без указателя на хвост пришлось бы каждый раз проходить весь список от головы, чтобы подойти к хвосту.

Той же цели легкого добавления в хвост списка можно достичь, используя циклический список. Тогда можно хранить единственный указатель, но не на первый, а на последний элемент (хвост). В циклическом списке головной элемент следует сразу на хвостовым, что позволяет без труда вставлять элементы как в хвост, так и в голову списка.

Другой вариант линейных списков – двусвязные (или двунаправленные) списки. В таком списке каждым элементом связаны два указателя: на следующий элемент и на предыдущий. При этом список может быть либо циклический, либо с указателями на голову и на хвост.

В некоторых случаях двусвязный список оказывается более удобным, чем односвязный. По такому списку можно перемещаться и вперед, и назад, а это, в частности, позволяет легко удалять текущий элемент, зная указатель на него, или вставлять новый элемент не только после текущего, но и перед ним.

К недостаткам двусвязного списка относится его бо’льшая громоздкость: с каждым элементов связан не один, а два указателя, и при операциях со списком требуется корректировать и те, и другие указатели.

Так всё-таки, списки лучше или хуже массивов?

У списков есть серьезные преимущества перед массивами:

- размер списка может увеличиваться практически неограниченно в рамках всего объема памяти, выделенного программе;

- структура списка более гибкая:

- легко добавляются и удаляются элементы в голове списка или после указанного элемента;

- легко выполняется слияние двух списков в один, разделение списка на два, изменение порядка элементов.

Но есть и серьезные недостатки списков по сравнению с массивами:

- доступ к элементу с известным номером возможен только проходом от головы до нужного элемента;

- при добавлении и удалении элементов используются не очень быстро работающие операторы new и delete;

- добавляется расход памяти на хранение одного или двух указателей вместе с каждым элементом. Например, если элементы списка – это целые числа, а каждый указатель занимает столько же места в памяти, как число, то получается, что двунаправленный список целых чисел будет занимать в памяти в три раза больше места, чем массив.

Общий вывод такой: списки не лучше и не хуже ,чем массивы. Просто они очень разные.

Во многих задачах возникает выбор, какую структуру данных лучше использовать, массив или линейный список? И если список, то в каком из вариантов?

На эти вопросы не может быть однозначного ответа. Правильный выбор зависит от того, какие операции будут в конкретной задаче чаще выполняться с этими данными:

- частое обращение к элементам по их номерам (индексам) однозначно требует массива;

- операции вставки/удаления элементов, неограниченный рост их числа могут склонить выбор в пользу списков.

Что мы узнали?

Что разные варианты сцепленных списков являются альтернативой представлению данных с помощью массивов.

Что к достоинствам списков можно отнести гибкость структуры и отсутствие жестких ограничений по памяти.

Недостатками же списков являются отсутствие эффективного доступа к элементам по индексу и дополнительные затраты памяти.

Что выбор наиболее эффективного представления данных зависит от того, какие операции предстоит часто выполнять при решении поставленной задачи.

На этом лекция закончена, благодарю за внимание.