План лекции 2

Представление данных

2.1 Представление данных

Переменная в компьютере действительно меняет значение. На самом деле, переменная — это место в памяти, в котором хранится текущее значение этой переменной. В языках программирования ряд языковых конструкций подразумевает *«разыменование»*: например, если переменная упоминается в правой части присваивания, вместо нее подставляется ее значение (это и есть «разыменование»: написано имя переменной, а оказалось — значение), а вот если переменная стоит слева от оператора присваивания := (и кроме нее там больше ничего нет), то она действительно обозначает место в памяти, куда будет записан результат вычисления правой части.

Подпрограммы (процедуры, функции). Это куски программы, которые можно вызвать (передать управление на их начало); по окончании работы они возвращают управление на следующий оператор, идущий за оператором вызова. При вызове можно передать параметры, в ответ подпрограмма может вернуть значение.

Область действия переменной — подпрограмма (главная программа). Одноименные переменные при каждом вызове подпрограммы — это *разные* переменные (так же, как и параметры).

2.2 Структуры данных

Абстрактная структура данных полностью описывается *набором операций*, которые могут быть к ней применены (например, «найти все данные»). В наиболее простом случае структура данных состоит из элементов некоторого базового типа[1] (при реализации может потребоваться, чтобы к элементам этого типа были применимы определенные операции: например, «выяснить, какой из двух элементов больше»).

Абстрактная структура данных может быть реализована по-разному. При разных реализациях время работы операций может быть различным.

[1] В более сложных случаях могут быть элементы разных базовых типов.

2.2.1 Массив

(Одномерный) *массив* — это абстрактная структура данных, к которой применима операция `[·]`. Эта операция для массива a и целого числа i выдает местонахождение элемента номер i из массива a. *Иначе говоря, выражение $a[i]$ может быть использовано как переменная: если оно находится слева от присваивания, в него может быть записано значение; если оно находится справа от присваивания, вместо него подставляется значение упомянутого элемента.*

Реализация: на РАМ можно отвести под массив несколько последовательных регистров; операция `[·]` реализуется при помощи косвенной адресации (например, если индексы нумеруются с нуля, $[i]$ обращается к регистру номер $i_0 + i$, где i_0 — номер первого из отведенных под массив регистров).

В большинстве языков программирования имеется встроенная реализация массива.

2.2.2 Файл

Файл последовательного доступа — это абстрактная структура данных, к которой применимы операции

- READ (с переходом к следующей записи),
- WRITE (с переходом к следующей записи) и
- REWIND.

Его также можно рассматривать как ленту, по которой движется читающая/пишущая головка: каждый раз после чтения или записи она передвигается на одну позицию в направлении конца файла; по команде REWIND она возвращается к самому началу.

2.2.3 Очередь

Очередь (FIFO[2]) — это абстрактная структура данных, к которой применимы операции

- ENQUEUE, позволяющая добавить элемент в конец очереди;
- DEQUEUE, позволяющая удалить элемент из начала очереди;
- HEAD, позволяющая посмотреть элемент из начала очереди.

Реализация: можно реализовать как массив со счетчиком, который действует по модулю числа элементов в массиве, но тогда размер очереди будет ограничен. Можно также реализовать как список (что такое список — см. ниже).

2.2.4 Очередь с приоритетами

$\boxed{\text{ПРОБЕЛ В КОНСПЕКТЕ.}}$

[2] First In, First Out

2.2.5 Стек

Стек (LIFO[3]) — это абстрактная структура данных, к которой применимы операции

- PUSH, позволяющая положить элемент на верхушку стека;
- POP, позволяющая снять (удалить) элемент с верхушки стека и выдать его значение.

Часто реализация стека допускает и применение операции `[·]` (как у массива). Заметим, что операция «посмотреть элемент из верхушки стека» реализуется тривиально (POP, а затем PUSH).

Реализация: можно реализовать как массив со счетчиком. В частности, это очень легко сделать на РАМ (см. реализацию массива выше).

2.2.6 (Однонаправленный) список

ПРОБЕЛ В КОНСПЕКТЕ.

Реализация:

ПРОБЕЛ В КОНСПЕКТЕ.

2.2.7 Двунаправленный список

ПРОБЕЛ В КОНСПЕКТЕ.

Реализация:

ПРОБЕЛ В КОНСПЕКТЕ.

2.2.8 Словарь

ПРОБЕЛ В КОНСПЕКТЕ.

О реализации словарей мы подробно поговорим несколькими лекциями позже.

[3] Last In, First Out