Тесты кафедры анатомии человека МГМСУ им. А.И. Евдокимова

Алгоритмизация

Понятие алгоритма и его свойства

Алгоритм - описанная на некотором языке точная конечная система правил, определяющая содержание и порядок действий над некоторыми объектами, строгое выполнение которых дает решение поставленной задачи. Понятие алгоритма, являющееся фундаментальным в математике и информатике, возникло задолго до появления средств вычислительной техники. Слово «алгоритм» появилось в средние века, когда европейцы познакомились со способами выполнения арифметических действий в десятичной системе счисления, описанными узбекским математиком Муххамедом бен Аль-Хорезми. Слово алгоритм - есть результат европейского произношения слов аль-Хорезми. Первоначально под алгоритмом понимали способ выполнения арифметических действий над десятичными числами. В дальнейшем это понятие стали использовать для обозначения любой последовательности действий, приводящей к решению поставленной задачи.
Любой алгоритм существует не сам по себе, а предназначен для определенного исполнителя (человека, робота, компьютера, языка программирования и т.д.). Свойством, характеризующим любого исполнителя, является то, что он умеет выполнять некоторые команды. Совокупность команд, которые данный исполнитель умеет выполнять, называется системой команд исполнителя. Алгоритм описывается в командах исполнителя, который будет его реализовывать. Объекты, над которыми исполнитель может совершать действия, образуют так называемую среду исполнителя. Исходные данные и результаты любого алгоритма всегда принадлежат среде того исполнителя, для которого предназначен алгоритм.
Значение слова «алгоритм» очень схоже со значениями слов «рецепт», «метод», «процесс», Однако, в отличие от рецепта или процесса, алгоритм характеризуется следующими свойствами: дискретностью, массовостью, определенностью, результативностью, формальность.
Дискретность (разрывность - противоположно непрерывности) - это свойство алгоритма, характеризующее его структуру: каждый алгоритм состоит из отдельных законченных действий, говорят: «Делится на шаги».
Массовость - применимость алгоритма ко всем задачам рассматриваемого типа, при любых исходных данных. Например, алгоритм решения квадратного уравнения в области действительных чисел должен содержать все возможные исходы решения, т.е., рассмотрев значения дискриминанта, алгоритм находит либо два различных корня уравнения, либо два равных, либо делает вывод о том, что действительных корней нет.

Определенность (детерминированность, точность) - свойство алгоритма, указывающее на то, что каждый шаг алгоритма должен быть строго определен и не допускать различных толкований; также строго должен быть определен порядок выполнения отдельных шагов. Помните сказку про Ивана-царевича? «Шел Иван-царевич по дороге, дошел до развилки. Видит большой камень, на нем надпись: «Прямо пойдешь - голову потеряешь, направо пойдешь - жену найдешь, налево пойдешь - разбогатеешь». Стоит Иван и думает, что дальше делать». Таких инструкций алгоритм содержать не может.
Результативность - свойство, состоящее в том, что любой алгоритм должен завершаться за конечное (может быть очень большое) число шагов. Вопрос о рассмотрении бесконечных алгоритмов остается за рамками теории алгоритмов.
Формальность - это свойство указывает на то, что любой исполнитель, способный воспринимать и выполнять инструкции алгоритма, действует формально, т.е. отвлекается от содержания поставленной задачи и лишь строго выполняет инструкции. Рассуждать «что, как и почему?» должен разработчик алгоритма, а исполнитель формально (не думая) поочередно исполняет предложенные команды и получает необходимый результат.
Как правило, для любого заданного алгоритма можно выделить семь характеризующих его независимых параметров:
1. совокупность возможных исходных данных;
2. совокупность возможных промежуточных результатов;
3. совокупность результатов;
4. правило начала;
5. правило непосредственной переработки;
6. правило окончания;
7. правило извлечения результата.
Пример. Алгоритм вычисления корней квадратного уравнения
.
Исходные данные - коэффициенты уравнения: a - при второй степени неизвестного; b - при первой степени неизвестного; c - при нулевой степени неизвестного.
Искомый результат - значения корней уравнения x1 и x2.
Предписание:
1. Вычислить значение дискриминанта по формуле .
2. Если , то вычислить значения корней уравнения по формулам:
3. Если , то уравнение не имеет корней.
Приведенное предписание обладает всеми свойствами алгоритма:
· дискретностью - предписание разбито на этапы (шаги выполнения);
· однозначностью - однозначно указано как обозначаются коэффициенты, дискриминант, корни, приведены формулы для вычисления дискриминанта и корней уравнения;
· результативностью - при выполнении предписания получается результат - значения корней уравнения или заключение, что уравнение не имеет решения;
· массовостью - в предписании составлено для общего случая (указаны не

конкретные значения коэффициентов).
Способы описания алгоритмов

Рассмотрим следующие способы описания алгоритма: словесное описание, псевдокод, блок-схема, программа.
Словесное описание представляет структуру алгоритма на естественном языке. Например, любой прибор бытовой техники (утюг, электропила, дрель и т.п.) имеет инструкцию по эксплуатации, т.е. словесное описания алгоритма, в соответствии которому данный прибор должен использоваться.
Никаких правил составления словесного описания не существует. Запись алгоритма осуществляется в произвольной форме на естественном, например, русском языке. Этот способ описания не имеет широкого распространения, так как строго не формализуем (под «формальным» понимается то, что описание абсолютно полное и учитывает все возможные ситуации, которые могут возникнуть в ходе решения); допускает неоднозначность толкования при описании некоторых действий; страдает многословностью.
Псевдокод - описание структуры алгоритма на естественном, частично формализованном языке, позволяющее выявить основные этапы решения задачи, перед точной его записью на языке программирования. В псевдокоде используются некоторые формальные конструкции и общепринятая математическая символика. Строгих синтаксических правил для записи псевдокода не существует. Это облегчает запись алгоритма при проектировании и позволяет описать алгоритм, используя любой набор команд. Однако в псевдокоде обычно используются некоторые конструкции, присущие формальным языкам, что облегчает переход от псевдокода к записи алгоритма на языке программирования. Единого или формального определения псевдокода не существует, поэтому возможны различные псевдокоды, отличающиеся набором используемых слов и конструкций.
Блок-схема - описание структуры алгоритма с помощью геометрических фигур с линиями-связями, показывающими порядок выполнения отдельных инструкций. Этот способ имеет ряд преимуществ. Благодаря наглядности, он обеспечивает «читаемость» алгоритма и явно отображает порядок выполнения отдельных команд. В блок-схеме каждой формальной конструкции соответствует определенная геометрическая фигура или связанная линиями совокупность фигур.
Рассмотрим некоторые основные конструкции, использующиеся для построения блок-схем.

Блок, характеризующий начало/конец алгоритма (для подпрограмм - вызов/возврат)

Блок - процесс, предназначенный для описания отдельных действий

Блок - предопределенный, процесс, предназначенный для обращения к вспомогательным алгоритмам (подпрограммам)

Блок - ввода/вывода с неопределенного носителя

Блок - ввод с клавиатуры

Блок - решение (проверка условия или условный блок)

Блок, описывающий цикл с параметром

Блок - границы цикла, описывающий циклические процессы типа: «цикл с предусловием», «цикл с постусловием»:

Блок - вывод не монитор

Соединительные блоки

Описания алгоритма в словесной форме, на псевдокоде или в виде блок-схемы допускают некоторый произвол при изображении команд, Вместе с тем она настолько достаточна, что позволяет человеку понять суть дела и исполнить алгоритм. На практике исполнителями алгоритмов выступают компьютеры. Поэтому алгоритм, предназначенный для исполнения на компьютере, должен быть записан на «понятном» ему языке, такой формализованный язык называют языком программирования.
Программа - описание структуры алгоритма на языке алгоритмического программирования.
Основные алгоритмические конструкции

Элементарные шаги алгоритма можно объединить в следующие алгоритмические конструкции: линейные (последовательные), разветвляющиеся, циклические и рекурсивные.
Линейная алгоритмическая конструкция

Линейной называют алгоритмическую конструкцию, реализованную в виде

последовательности действий (шагов), в которой каждое действие (шаг) алгоритма выполняется ровно один раз, причем после каждого i-го действия (шага) выполняется (i +1)-е действие (шаг), если i -е действие - не конец алгоритма.
Пример: опишем алгоритм сложения двух чисел на псевдокоде в виде блок-схемы.
Псевдокод:
Ввод двух чисел a, b.
Вычисляем сумму S = a + b.
Вывод S.
Конец.Рисунок - блок-схема к примеру
Разветвляющаяся алгоритмическая конструкция

Разветвляющейся (или ветвящейся) называется алгоритмическая конструкция, обеспечивающая выбор между двумя альтернативами в зависимости от значения входных данных. При каждом конкретном наборе входных данных разветвляющийся алгоритм сводится к линейному. Различают неполное (если - то) и полное (если - то - иначе) ветвления. Полное ветвление позволяет организовать две ветви в алгоритме (то или иначе), каждая из которых ведет к общей точке их слияния, так что выполнение алгоритма продолжается независимо оттого, какой путь был выбран (рис). Рисунок - Полное ветвление
Неполное ветвление предполагает наличие некоторых действий алгоритма только на одной ветви (то), вторая ветвь отсутствует, т.е. для одного из результатов проверки никаких действий выполнять не надо, управление сразу переходит к точке слияния (рис). Рисунок - неполное ветвление
Пример: вывести значение наибольшего из двух чисел.
Псевдокод:
1. Ввод двух чисел a, b.
2. ЕСЛИ a > b, ТО «выводим a»,
ИНАЧЕ «выводим b»,
3. Конец.
Рисунок - Блок-схема к примеру
В данном примере реализовано полное ветвление. ЕСЛИ значения входных данных таковы, что a >b, ТО выполняется линейный алгоритм:
Ввод двух чисел a, b.
Вывод a,
ИНАЧЕ, когда a<b, выполняется линейный алгоритм:
Ввод двух чисел a, b.
Вывод b.
Вывод: алгоритм является разветвляющимся и состоит из двух ветвей.
Алгоритмическая конструкция «Цикл»

Циклической (или циклом) называют алгоритмическую конструкцию, в которой некая, идущая подряд группа действий (шагов) алгоритма может выполняться несколько раз, в зависимости от входных данных или условия задачи. Группа

повторяющихся действий на каждом шагу цикла называется телом цикла. Любая циклическая конструкция содержит в себе элементы ветвящейся алгоритмической конструкции.
Рассмотрим три типа циклических алгоритмов: цикл с параметром (который называют арифметическим циклом), цикл с предусловием и цикл с постусловием (их называют итерационными).

Арифметический цикл

В арифметическом цикле число его шагов (повторений) однозначно определяется правилом изменения параметра, которое задается с помощью начального (N) и конечного (К) значении параметра и шагом (h) его изменения. Т.е., на первом шаге цикла значение параметра равно N, на втором N+h, на третьем - N+2h и т.д. На последнем шаге цикла значение параметра не больше К, но такое, что дальнейшее его изменение приведет к значению, большему, чем К. алгоритм линейный разветвляющийся цикличный
Пример: Вывести 10 раз слово «Привет!».
Параметр цикла обозначим i, он будет отвечать за количество выведенных слов. При i = 1 будет выведено первое слово, при i = 2 будет выведено второе слова и т.д. Так как требуется вывести 10 слов, то последнее значение параметра i = 10. В заданном примере требуется 10 раз повторить одно и то же действие: вывести слово «Привет!».
Составим алгоритм, используя арифметический цикл, в котором правило изменения параметра i - 1, 10, 1. То есть начальное значение параметра i = 1; конечное значение i =10; шаг изменения h = 1. На рисунке представлена блок-схема алгоритма решения данной задачи.Рисунок - Блок-схема к примеру
Цикл с предусловием

Количество шагов цикла заранее не определено и зависит от входных данных задачи. В данной циклической структуре сначала проверяется значение условного выражения (условие) перед выполнением очередного шага цикла. Если значение условного выражения истинно, исполняется тело цикла. После чего управление вновь передается проверке условия и т.д. Эти действия повторяются до тех пор, пока условное выражение не примет значение ЛОЖЬ. При первом же несоблюдении условия цикл завершается.
а б
Рисунок - Блок-схема цикла с предусловием
Блок-схема данной конструкции представлена на рисунке двумя способами; с помощью условного блока а и с помощью блока границы цикла б.
Особенностью цикла с предусловием является то, что если изначально условное выражение ложно, то тело цикла не выполнится ни разу.
Цикл с постусловием

Как и в цикле с предусловием, в циклической конструкции с постусловием заранее не определено число повторений тела цикла, оно зависит от входных данных задачи. В отличие от цикла с предусловием, тело цикла с постусловием всегда будет выполнено хотя бы один раз, после чего проверяется условие. В этой конструкции тело цикла будет выполняться до тех пор, пока значение условного выражения ложно. Как только оно становится истинным, выполнение команды прекращается. Блок-схема данной конструкции представлена на рисунке двумя способами: с помощью условного блока а и с помощью блока управления б.

а б
Рисунок - Блок-схема цикла с постусловием

Рекурсивный алгоритм

Рекурсивным называется алгоритм, организованный таким образом, что в процессе выполнения команд на каком-либо шаге он прямо или косвенно обращается сам к себе.
Простые типы данных: переменные и константы

Реальные данные, которые обрабатывает программа, - это целые и вещественные числа, символы и логические величины. Эти простые типы данных называют базовыми. Все данные, обрабатываемые компьютером, хранятся в ячейках памяти компьютера, каждая из которых имеет свой адрес. Для того чтобы не следить за тем, по какому адресу будут записаны те или иные данные, в языках программирования используется понятие переменной, позволяющее отвлечься от адреса ячейки памяти и обращаться к ней с помощью имени (идентификатора).
Переменная - именованный объект (ячейка памяти), который может изменять свое значение. Имя переменной указывает на значение, а способ ее хранения и адрес остаются скрытыми от программиста. Кроме имени и значения, переменная имеет тип, определяющий, какая информация находится в памяти. Тип переменной задает:
используемый способ записи информации в ячейки памяти;
необходимый объем памяти для ее хранения.
Объем памяти для каждого типа определяется таким образом, чтобы в него можно было поместить любое значение из допустимого диапазона значений данного типа. Например, тип «байт» может принимать значения от 0 до 255, что в двоичном коде ($25510 = 11111111_2$) соответствует ячейке памяти длиной в 8 бит (или 1 байт).
В описанных выше алгоритмах все данные хранятся в виде переменных. Например, инструкция «Ввод двух чисел a, b» означает введение пользователем значений двух переменных, а инструкция «К=К+1» означает увеличение значения переменной К на единицу.
Если переменные присутствуют в программе, на протяжении всего времени ее работы - их называют статическими. Переменные, создающиеся и уничтожающиеся на разных этапах выполнения программы, называют динамическими.

Все остальные данные в программе, значения которых не изменяются на протяжении ее работы, называют константами или постоянными. Константы, как и переменные, имеют тип. Их можно указывать явно, например, в инструкции «К = К + 1» 1 есть константа, или для удобства обозначать идентификаторами: pi = 3,1415926536. Только значение pi нельзя изменить, так как это константа, а не переменная.

Структурированные данные и алгоритмы их обработки

Для повышения производительности и качества работы необходимо иметь данные, максимально приближенные к реальным аналогам. Тип данных, позволяющий хранить вместе под одним именем несколько переменных, называется структурированным. Каждый язык программирования имеет свои структурированные типы. Рассмотрим структуру, объединяющую элементы одного типа данных - массив.

Массивом называется упорядоченная совокупность однотипных величин, имеющих общее имя, элементы которой адресуются (различаются) порядковыми номерами (индексами). В качестве иллюстрации можно представить шкаф, содержащий множество пронумерованных ящиков (совокупность - «Ящик № 1», «Ящик № 2», «Ящик № 3» и т.д.; «Ящик» - общее имя всех ее элементов). Доступ к содержимому конкретного ящика (элементу массива) осуществляется после выбора ящика по его номеру (индексу). Элементы массива в памяти компьютера хранятся по соседству, одиночные элементы простого типа такого расположения данных в памяти не предполагают. Массивы различаются количеством индексов, определяющих их элементы.

Одномерный массив (шкаф ящиков в один ряд) предполагает наличие у каждого элемента только одного индекса. Примерами одномерных массивов служат арифметическая (ai) и геометрическая (bi) последовательности, определяющие конечные ряды чисел. Количество элементов массива называют размерностью. При определении одномерного массива его размерность записывается в круглых скобках, рядом с его именем. Например, если сказано: «задан массив A(10)», это означает, что даны элементы: a1, a2, ... , a10. Рассмотрим алгоритмы обработки элементов одномерных массивов.

Ввод элементов одномерного массива осуществляется поэлементно, в порядке, необходимом для решения конкретной задачи. Обычно, когда требуется ввести весь массив, порядок ввода элементов не важен, и элементы вводятся в порядке возрастания их индексов.

Рассмотрим двумерный массив (шкаф с множеством ящиков, положение которых определяется двумя координатами - по горизонтали и по вертикали). В математике двумерный массив (таблица чисел) называется матрицей. Каждый ее элемент имеет два индекса aij, первый индекс i определяет номер строки, в которой находится элемент (координата по горизонтали), а второй, j - номер столбца (координата по вертикали). Двумерный массив характеризуется двумя размерностями N и M, определяющими число строк и столбцов соответственно.

Ввод элементов двумерного массива осуществляется построчно, в свою очередь, ввод каждой строки производится поэлементно, тем самым определяется циклическая конструкция, реализующая вложение циклов. Внешний цикл определяет номер вводимой строки (i), внутренний - номер элемента по столбцу (j). На рис. Представлен алгоритм ввода матрицы A(N на M).Разберем примеры алгоритма на псевдокоде:

Дан массив целых чисел {Ai}, где i=1,2,3,…,M. Пусть M равно 15. Программа вычисляет произведение сумм некоторых элементов этого массива. В программе введены следующие константы: G=1; W=12; T=8; L=15.

```
ПРОГРАММА 15;
ФУНКЦИЯ SUMMA(I1,I2);
НАЧАТЬ ФУНКЦИЮ
||S:=0;
||НЦ ДЛЯ I:=I1 ДО I2
||||S:=S + A[I]
||КЦ;
||SUMMA:=S
КОНЕЦ ФУНКЦИИ;
НАЧАТЬ ПРОГРАММУ
||ПИСАТЬ ('ВВЕДИТЕ ЗНАЧЕНИЯ МАССИВА A:' );
||НЦ ДЛЯ J:=1 ДО M
||||ЧИТАТЬ (A[J]);
||КЦ;
||P:=SUMMA (G, W)*SUMMA(T, L);
||ПИСАТЬ ('ПРОИЗВЕДЕНИЕ РАВНО:', P:6)
КОНЕЦ ПРОГРАММЫ.
```

Для удобства дальнейших пояснений, перепишем псевдокод с номерами строк:

```
1 ПРОГРАММА 15;
2 ФУНКЦИЯ SUMMA(I1,I2);
3 НАЧАТЬ ФУНКЦИЮ
4 ||S:=0;
5 ||НЦ ДЛЯ I:=I1 ДО I2
6 ||||S:=S + A[I]
7 ||КЦ;
8 ||SUMMA:=S
9 КОНЕЦ ФУНКЦИИ;
10 НАЧАТЬ ПРОГРАММУ
11 ||ПИСАТЬ ('ВВЕДИТЕ ЗНАЧЕНИЯ МАССИВА A:' );
12 ||НЦ ДЛЯ J:=1 ДО M
13 ||||ЧИТАТЬ (A[J]);
14 ||КЦ;
15 ||P:=SUMMA (G, W)*SUMMA(T, L);
16 ||ПИСАТЬ ('ПРОИЗВЕДЕНИЕ РАВНО:', P)
```

17 КОНЕЦ ПРОГРАММЫ.
В строке 1 записано название программы - «Программа 15».
В строке 2 описан заголовок функции Summa. Как видно из заголовка в функцию передается 2 формальных параметра - I1 и I2. Само тело функции записано в строках с 3 по 9. Проанализируем тело функции.
В строке 4 переменной S присваивается начальное значение равное 0.
В строках с 5 по 7 записан цикл. Количество повторений цикла известно - оно равно I2-I1. В теле цикла (строка 6) производится прибавление к прежнему значению переменной S элемента под номером I массива из A.
После цикла, в строке 8 записан результат, который будет выдавать функция (это называется - возвращать результат) - результат равен значению переменной S.
Например, если I1=1, I2=1, A[1]=1, A[2]=2, A[3]=4, то выполнение тела функции будет происходить следующим образом:
Шаг 1: 4 ||S:=0;
Шаг 2: 5 ||НЦ ДЛЯ I:=1 ДО 3
Шаг 3: 6 ||||S:=0 + A[1]=0+1=1
Шаг 4: увеличение счетчика цикла: I=I+1=1+1=2
Шаг 5: возврат на начало цикла (строка 5)
Шаг 6: 6 ||||S:=1 + A[2]=1+2=3
Шаг 7: увеличение счетчика цикла: I=I+1=2+1=3
Шаг 8: возврат на начало цикла (строка 5)
Шаг 9: 6 ||||S:=3 + A[3]=3+3=6
Шаг 10: увеличение счетчика цикла: I=I+1=3+1=4
Шаг 11: возврат на начало цикла (строка 5)
Шаг 12: поскольку значение счетчика цикла больше чем заданный предел (I2=3) то происходит выход из цикла и переход на сроку за циклом
Шаг 13: 8 ||SUMMA:=S=6
Шаг 14: завершение функции
Итак, функция производит суммирование всех элементов массива A, начиная с элемента с номером I1 и заканчивая номером I2. Математически, это записывается:
В строке 10 начинается выполнение программы.
В строке 11 на экран выводится надпись «ВВЕДИТЕ ЗНАЧЕНИЯ МАССИВА A:».
В строках с 12 по 14 записан цикл, в котором последовательно вводятся элементы массива A.
В строке 15 вызывается функция SUMMA с фактическими параметрами G и W и еще раз с фактическими параметрами T и L. Результат перемножается и присваивается переменной P.
В строке 16 на экран выводится надпись «ПРОИЗВЕДЕНИЕ РАВНО» и после нее значение переменной P.
Итак, программа выполняется следующим образом:
Шаг 1: на экран пишется «ВВЕДИТЕ ЗНАЧЕНИЯ МАССИВА A:» (строка 10)
Шаг 2: в цикле вводятся элементы массива A (строки 12-14, количество повторений цикла M-1)

Шаг 3: выполняется вызов функции SUMMA (строка 15)
Шаг 4: выполняется тело функции SUMMA (строки 2-9. При
этом вместо I1 подставляется G, а вместо I2 подставляется W. Результат:
)
Шаг 5: выполняется тело функции SUMMA (строки 2-9. При
этом вместо I1 подставляется T, а вместо I2 подставляется L. Результат:
)
Шаг 6: переменной P присваивается произведение результатов вызовов функций SUMMA (строка 15. Результат:
)
Шаг 7: на экран выводится «ПРОИЗВЕДЕНИЕ РАВНО:» (строка 16)
Шаг 8: на экран выводится значение переменной P (строка 16)
Шаг 9: выполнение программы завершается (строка 17)...