Лекция 1.
Алгоритм. Свойства алгоритмов. Формы записи алгоритмов. Линейные алгоритмы.

Происхождение термина «алгоритм» связано с математикой. В IX веке в Багдаде жил ученый ал(аль)-Хорезми (полное имя – Мухаммед бен Муса ал-Хорезми), математик, астроном, географ. В одном из своих трудов он описал десятичную систему счисления и впервые сформулировал правила выполнения арифметических действий над целыми числами и обыкновенными дробями. Арабский оригинал этой книги был утерян, но остался латинский перевод XII в., по которому Западная Европа ознакомилась с десятичной системой счисления и правилами выполнения арифметических действий.

Правила в книгах ал-Хорезми в латинском переводе начинались словами «Алгоризми сказал». В других латинских переводах автор именовался как Алгоритмус. Со временем было забыто, что Алгоризми (Алгоритмус) – это автор правил, и эти правила стали называть алгоритмами.

Алгоритм - это понятное и точное предписание исполнителю совершить последовательность действий, направленных на достижение определенной цели или на решение поставленной задачи.

Свойства алгоритмов:

определенность – за конечное число шагов либо должен быть получен результат, либо доказано его отсутствие;

результативность – обязательное получение некоторого результата (числа, таблицы, текста, звука, изображения и т. д.) или сигнала о том, что данный алгоритм неприменим для решения поставленной задачи;

массовость – возможность получения результата при различных исходных данных для некоторого класса сходных задач;

формальность – отвлечение от содержания поставленной задачи и строгое выполнение некоторого правила, инструкции;

дискретность — возможность разбиения алгоритма на отдельные элементарные действия.

Правила построения алгоритмов.

Первое правило – при построении алгоритма необходимо задать множество объектов, с которыми он будет работать. Формализованное (закодированное) представление этих объектов носит название ***данные***. Алгоритм приступает к работе с некоторым набором данных, которые называются входными, и в результате своей работы выдает данные, которые называются выходными.

Второе правило – для работы алгоритма требуется память. В памяти размещаются входные данные, с которыми алгоритм начинает работать, промежуточные данные и выходные данные, которые являются результатом работы алгоритма. Память является дискретной, т.е. состоящей из отдельных ячеек. Поименованная ячейка памяти носит название переменной. В теории алгоритмов размеры памяти не ограничиваются.

Третье правило – дискретность. Алгоритм строится из отдельных шагов (действий, операций, команд). Множество шагов, из которых составлен алгоритм, конечно.

Четвертое правило – детерминированность (определяемость). После каждого шага необходимо указывать, какой шаг выполняется следующим, либо давать команду остановки.

Пятое правило – сходимость (результативность). Алгоритм должен завершать работу после конечного числа шагов. При этом необходимо указать, что считать результатом работы алгоритма.

Существуют следующие ***формы представления алгоритма***:

* словесная (вербальная) на неформальном языке;
* на языках программирования;
* графическая.

Словесная форма представления алгоритма является самой распространенной формой представления алгоритмов, адресованная человеку. Форму словестной записи имеют многие так называемые «бытовые алгоритмы», часто используемые в повседневной практике (например, инструкции).

Пример 1.

Пусть требуется записать последовательность элементарных действий для вычислений по формуле:

$$y = \frac{3x}{\sqrt{8x} + 1}$$

При этом предположении искомый словесный алгоритм может иметь вид:

1. Прочитать заданное значение x.
2. Умножить x на 8.
3. Из результата второго действия извлечь квадратный корень.
4. К результату третьего действия прибавить 1.
5. Умножить x на 3.
6. Результат пятого действия разделить на результат четвертого действия.
7. Записать значение результата y.

Приведенную выше запись можно сделать более компактной, воспользовавшись операцией присваивания :=

Смысл операции присваивания заключаются в следующем:

Пусть имеется предписание вида

$$y := A$$

y – переменная,

A - некоторое выражение.

Предписание означает следующее: выполнить все действия, предусмотренные формулой A и полученный результат (число) считать значением (т.е. присвоить) переменной y.

В левой части команды присваивания всегда должна стоять переменная. Выражение в правой части может быть переменной или числом.

Для того чтобы сделать нашу запись более компактной воспользуемся вспомогательными переменными, которые широко используются в алгоритмизации.

$$y = \frac{3x}{\sqrt{8x} + 1}$$

1. чтение x	2
2. a:= 8x	16
3. в:=$\sqrt{a}$	4
4. c:= в+1	5
5. d:= 3x	6
6. y:=d/c	1,2
7. запись y	
8. конец	

Операция присваивания допускает случаи, когда одна переменная может быть слева и справа от знака присваивания. Например,

$$к := к + 1$$

это значит, что требуется к значению переменной *к*, которое она имела к началу выполнения операции присваивания, присвоить число 1 и считать полученное значение новым значением переменной *к*.

1. чтение x
2. a:= 8x
3. a:=$\sqrt{a}$
4. a:= a+1
5. y:= 3x
6. y:=y/a
7. запись y
8. конец

Алгоритм, записанный на **языке программирования**, называется **программой.**

Графическая форма представления алгоритмов является более наглядной и строгой. Алгоритм изображается в виде последовательности связанных между собой блоков, каждый из которых соответствует выполнению одного или нескольких операторов. Такое графическое представление называется **блок-схемой алгоритма.**

Условные графические обозначения символов, используемых для составления блок-схемы алгоритма, стандартизированы.

Блок-схемой называется наглядное изображение алгоритма, когда отдельные действия (этапы алгоритма) изображаются при помощи различных геометрических фигур (блоков), а связи между этапами (последовательность выполнения этапов) указываются при помощи стрелок, соединяющие эти фигуры.

Выполнение блок-схем осуществляется по ГОСТ 19.701–90.

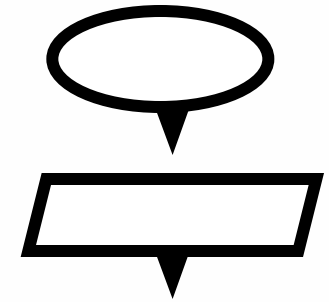
При выполнении блок-схем внутри каждого блока указывается поясняющая информация, которая характеризует действия, выполняемые этим блоком.

Потоки данных в схемах показываются линиями. Направление потока слева направо и сверху вниз считается стандартным. В случаях, когда необходимо внести большую ясность в схему или поток имеет направление отличное от стандартного, на линиях используются стрелки, указывающие это направление.

В схемах следует избегать пересечения линий. Пересекающиеся линии не имеют логической связи между собой, поэтому изменения направления в точках пересечения не допускаются. Если две или более входящих линии объединяются в одну исходящую линию, то место объединения линий смещается.

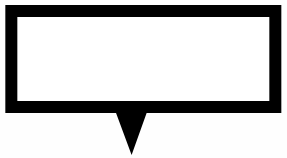
Количество входящих линий не ограничено, выходящая линия из блока должна быть одна, за исключением логического блока.

ОСНОВНЫЕ БЛОКИ БЛОК - СХЕМЫ

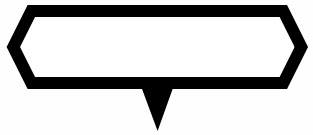


- НАЧАЛО (КОНЕЦ) БЛОК - СХЕМЫ

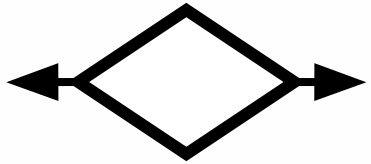
- БЛОК ВВОДА, ВЫВОДА ДАННЫХ



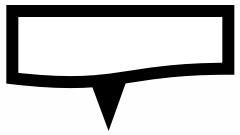
- БЛОК ОПЕРАТОРА ПРИСВАИВАНИЯ (ВЫЧИСЛЕНИЕ)



- БЛОК ЦИКЛА



- ЛОГИЧЕСКИЙ БЛОК



- БЛОК ВЫВОДА РЕЗУЛЬТАТОВ НА ПЕЧАТЬ

Представление алгоритма в виде блок-схемы является промежуточным, так как алгоритм в таком виде не может быть непосредственно выполнен компьютером, но помогает пользователю при создании (написании) программы для ПК.

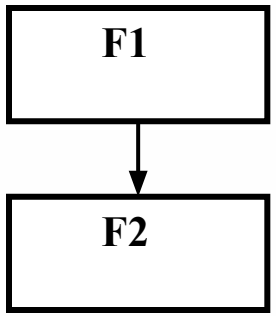
Выделяют три основные структуры алгоритмов:

1. Линейная.
2. Разветвляющаяся (альтернатива «если–то–иначе» или «если–то»).
3. Циклическая (повторение).

ЛИНЕЙНЫЕ СТРУКТУРЫ

Линейная структура – является основной. Она означает, что действия выполняются друг за другом.

Прямоугольник, показанный на рисунке, может представлять как одну единственную команду, так и множество операторов, необходимых для выполнения сложной обработки данных, где ***F1*** и ***F2*** – некоторые команды для соответствующего исполнителя. Команды записываются с помощью операции присваивания.



Пример 1.

Разработать блок-схему алгоритма вычисления площади и периметра прямоугольника по двум заданным сторонам ***а*** и ***в***.