Тема 1.1 **Алгоритм и его свойства**

1. Понятие алгоритма и его свойства

Само слово "алгоритм" происходит от имени персидского математика Аль Хорезми, который в IX веке разработал правила четырех арифметических действий (сегодня мы бы сказали алгоритмы арифметических действий).

В начале XX века алгоритмы стали объектом изучения математиков, появились различные математические уточнения понятия "алгоритм" и возникла целая отрасль математики – теория алгоритмов. Результаты, полученные теорией алгоритмов, служат теоретическим фундаментом всей компьютерной технологии, но в повседневной программистской практике не используются.

Алгоритм – это описание некоторой последовательности действий, приводящее к решению поставленной задачи.

Алгоритм – система четких однозначных указаний, которая определяет последовательность действий над некоторыми объектами и после конечного числа шагов приводит к получению требуемого результата.

Алгоритмы бывают численными и логическими.

Алгоритмы, в соответствии с которыми решение поставленных задач сводится к арифметическим действиям, называются *численными алгоритмами*.

Алгоритмы, в соответствии с которыми решение поставленных задач сводится к логическим действиям, называются *логическими алгоритмами* (алгоритмы поиска минимального числа, поиска пути в лабиринте).

Основными свойствами алгоритма являются:

1) *Дискретность* – разделение выполнения решения задачи на отдельные операции.

Под *дискретностью* понимается то, что алгоритм состоит из описания последовательности шагов обработки, организованных таким образом, что в начальный момент задаётся исходная ситуация, а после каждого следующего шага ситуация преобразуется на основе данных, полученные в предшествующие шаги обработки. Дискретность алгоритма означает, что он исполняется по шагам: каждое действие, предусмотренное алгоритмом, исполняется только после того, как закончилось исполнение предыдущего, то есть преобразование исходных данных в результат происходит во времени дискретно.

2) *Детерминированность* (*определенность*) – каждая команда алгоритма должна однозначно определять действия исполнителя.

Это свойство означает, что на каждом шаге алгоритма однозначно определяется преобразование данных, полученных на предшествующих шагах алгоритма, то есть на одинаковых исходных данных алгоритм должен всегда давать одинаковые результаты.

3) *Результативность* (*конечность*) - завершение работы алгоритма за конечное число шагов (при этом количество шагов может быть заранее не известным и различным для разных исходных данных).

4) *Массовость* (*универсальность*) - алгоритм решения задачи разрабатывается в общем виде, то есть возможность решения класса задач, различающихся лишь исходными данными. При этом исходные данные выбираются из некоторой области, называемой областью применимости алгоритма.

5) *Понятность* – содержание допустимого набора команд, понятного конкретному исполнителю. Каждый шаг алгоритма должен обязательно представлять собой какое-либо допустимое действие, т.е. алгоритм строится для конкретного исполнителя автором и должен быть им обоим понятен. Это облегчает проверку и модификацию алгоритма при необходимости.

2. Формы записи алгоритмов

Процесс составления алгоритмов называют *алгоритмизацией*.

Алгоритм, реализующий решение задачи, можно представить различными способами – с помощью графического или текстового описания.

Графический способ представления алгоритмов имеет ряд преимуществ благодаря визуальности и явному отображению процесса решения задачи. Алгоритмы, представленные графическими средствами, получили название *блок-схем.*

Текстовое описание алгоритма является достаточно компактным и может быть реализовано на *естественном языке* или *специальном* (*алгоритмическом*) *языке* в виде программы.

Все три способа представления алгоритмов можно считать взаимодополняющими друг друга. На этапе проектирования алгоритмов наилучшим способом является графическое представление, а на этапах проверки и применения алгоритма – текстовая запись в виде программы.

Правила выполнения блок-схем:

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.

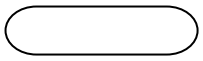
Выполнение блок-схем осуществляется по ГОСТ 19.701–90.

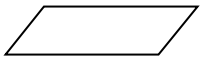
При выполнении блок-схем внутри каждого блока указывается поясняющая информация, которая характеризует действия, выполняемые этим блоком. Потоки данных в схемах показываются линиями. Направление потока слева направо и сверху вниз считается стандартным. В случаях, когда необходимо внести большую ясность в схему или поток имеет направление отличное от стандартного, на линиях используются стрелки, указывающие это направление.

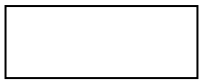
В схемах следует избегать пересечения линий. Пересекающиеся линии не имеют логической связи между собой, поэтому изменения направления в точках пересечения не допускаются. Если две или более входящих линии объединяются в одну исходящую линию, то место объединения линий смещается.

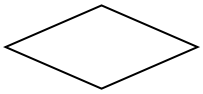
Количество входящих линий не ограничено, выходящая линия из блока должна быть одна, за исключением логического блока.

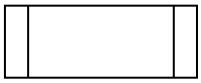
Основными элементами блок-схем являются:

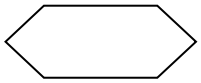

- начало (конец) алгоритма


- блок ввода-вывода данных


- блок вычислений


- логический блок, в котором направление потока информации выбирается в зависимости от некоторого условия


- процесс пользователя (подпрограмма)


- блок модификации, в котором функция выполняет действия, изменяющие пункты (например, заголовок цикла)


- соединитель, используется для указания связи между потоками информации в пределах одного листа

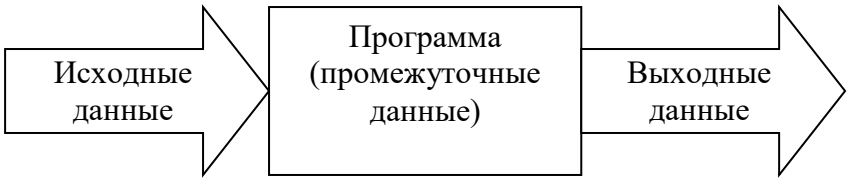

- межстраничный соединитель, т.е. указание связи между информацией на разных листах

3. Данные и их типы

Алгоритм, реализующий решение задачи, всегда работает с данными.

Данные – это любая информация, представленная в формализованном виде и пригодная для обработки алгоритмом.

По отношению к программе данные делятся на исходные, промежуточные и выходные.



Данные, известные перед выполнением алгоритма, являются начальными, *исходными данными*. Данные, используемые в процессе выполнения программы, являются *промежуточными данными*. Результат решения задачи – это конечные, *выходные данные*.

Данные делятся на переменные и константы.

Переменные – это такие данные, значения которых могут изменяться в процессе выполнения алгоритма.

Константы – это данные, значения которых не меняются в процессе выполнения алгоритма.

Любая величина имеет *3 основные свойства*:

- *имя*, которое задается идентификатором, представляющим собой последовательность букв и цифр, начинающихся с буквы;
- *значение*;
- *тип данных* – это такая характеристика данных, которая задает множество допустимых значений и определяет множество операций, которые можно к этим данным применить.

Типы данных делят на 2 группы:

1) *Простые (скалярные) типы* – содержат одно единственное значение. К ним относятся:
   - *целый тип* – определяет подмножество допустимых значений из множества целых чисел (например: 23, -12);
   - *вещественный тип* - определяет подмножество допустимых значений из множества целых и дробных чисел в некотором диапазоне (например: 2,5; -0,01; $3{,}6 \cdot 10^9$);
   - *логический тип* – переменная принимает только два значения: истина (true) и ложь (false);
   - *символьный тип* – любые символы компьютерного алфавита (например: ‘a’, ‘5’, ‘+’).
2) *Структурированные типы* - описывают наборы однотипных или разнотипных данных (т.е. содержат несколько значений), с которыми алгоритм должен работать как с одной именованной переменной. К ним относятся: массивы, строки, множества и т.д.