**Тема 3.2. Алгоритмические конструкции**

К основным алгоритмическим конструкциям относятся:

1. ***Линейный***. Такой вычислительный процесс получил данное название за счет того, что все действия выполняются в линейной последовательности, при этом каждый шаг выполняется не более одного раза. Если рассматривать схему задачи, то блоки в ней размещаются один под другим в зависимости от порядкового номера выполнения. Линейные алгоритмы работают таким образом, что от исходных данных не меняется направление и смысл действий. Такой способ решения подойдет для вычисления суммы или разности, площади фигуры или ее периметра и т. п. Основным типом алгоритмической конструкции является именно он.
2. ***Разветвляющийся***. Этот вычислительный процесс подразумевает наличие логического выражения (далее ЛВ) и выбора условия (ветви «ложь» и «правда»). В каждом случае реализуется лишь одна из двух и более команд. Нет задач и не может быть, в которых будут выполнены еще и другие варианты. Если в алгоритме две ветви, он простой, если больше двух – сложный. Причем последний процесс легко представляется за счет первого. Основным типом алгоритмической конструкции является как первый пункт, так и второй. Следующий вид тоже входит в этот список.
3. ***Циклический***. В таком алгоритме обязательно будет элемент, повторяющийся многократно, при этом используются разные исходные данные. Иными словами, такой процесс называется циклом.

Нужно заметить, что все основные алгоритмические конструкции (следование, ветвление, цикл) взаимосвязаны друг с другом, хотя и могут использоваться отдельно.

**Базовый алгоритм**

Стоит запомнить то, что к основным алгоритмическим конструкциям не относится базовый алгоритм. Что он собой представляет? Данное понятие уже давно не встречается в современной литературе, однако это не значит, что его и вовсе больше не существует. Учитывая, что в решении задач может встретиться несколько ветвлений или повторений, можно выделить следующее заключение. Основные алгоритмические конструкции (линейные, разветвляющие, циклические) являются базовыми. По сути, они представляют собою «структурную единицу» каждой так называемой инструкции.

**Линейные алгоритмы**

Все действия, которые воспроизводятся в линейном алгоритме, имеют четко последовательное выполнение, все шаги выполняются строго друг за другом. Как правило, такие задачи небольшие и имеют низкий уровень сложности.

Примером линейного алгоритма может быть процесс приготовления чая:

1. Налить воды в чайник.
2. Поставить чайник на плиту закипать.
3. Взять чашку.
4. Насыпать в чашку чай.
5. Добавить сахар.
6. После кипения налить в чашку кипяток.
7. Взять ложку.
8. Перемешать сахар.

Программирование основных алгоритмических конструкций - достаточно тяжелое дело, но если речь идет о линейных алгоритмах, то зачастую реализовать их очень легко.

**Разветвляющиеся алгоритмы**

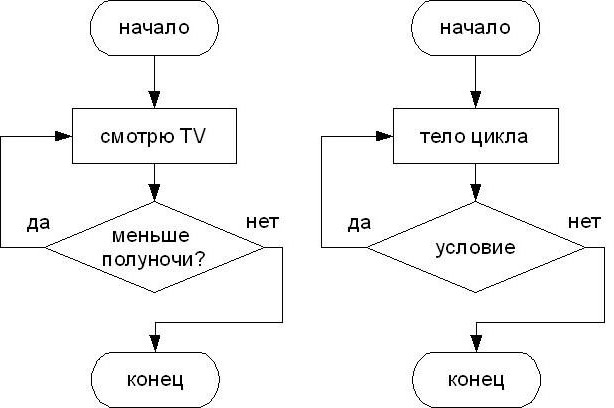
Как понять, что алгоритм является разветвляющимся? Достаточно убедиться в наличии выбора из двух или более вариантов действия, в зависимости от выполнения или невыполнения условия. Каждый путь называется ветвью.

Основным признаком разветвляющегося алгоритма является существование условного перехода. Он происходит во время проверки выражения на истину или ложь.

Как правило, логические выражения представлены знаками "меньше", "больше", "меньше или равно", "больше или равно", "равно", "не равно". Иногда встречаются варианты, где условие связано между собой с помощью команд and (и) и or (или).

Примером такого алгоритма может быть решение следующей задачи: если выражение ((х+3)/1) равняется положительному числу, то вывести результат на экран, если отрицательному – сообщить пользователю об ошибке.

Достаточно просто на практике использовать основные алгоритмические конструкции. Ветвление является одним из самых распространенных методов решения.



**Создание циклов и их виды**

Что же необходимо для создания цикла?

* Счетчик цикла. Это переменная, которая задает начальное значение, и при повторении действия она будет изменяться. Обязательно она должна входить в алгоритм. Основные алгоритмические конструкции циклового типа работать без нее не будут.
* Смена показателя вышеописанных данных перед новым повторением самого цикла.
* Проверка условия, чтобы ЭВМ решила, следует ли снова «прокручивать» цикл или больше в этом нет нужды.

Циклы могут быть детерминированными и итерационными. Первые представляют собой повтор действий с уже известным количеством повторений. Итерационный цикл – тот, который повторяется неопределенное количество раз, пока условие не станет правдой или ложью.

**Детерминированный цикл или цикл со счетчиком**

Цикл со счетчиком – цикл, который включает в себя переменную, изменяющую значение с определенным шагом. Шаг задается пользователем или прописывается программистом во время написания обеспечения. Большая часть языков для такого цикла использует оператор for.

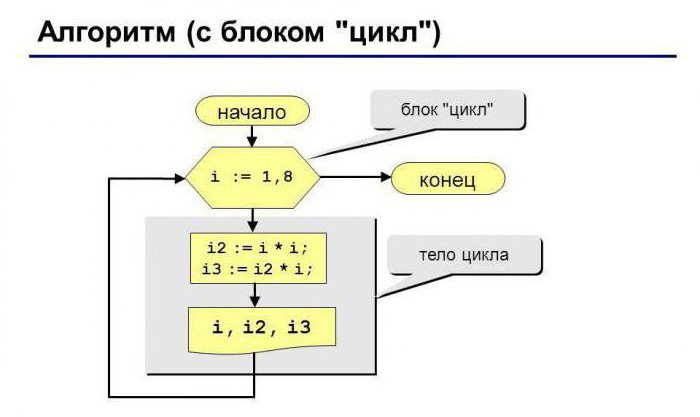
Чтобы программа выводила на экран две строки 4 раза:

1. «Как дела?»
2. «Хорошо, спасибо!»
3. «Как дела?»
4. «Хорошо, спасибо!»

Необходимо создать детерминированный цикл. Как это выглядит? Воспользуемся языком «Паскаль» для лучшего восприятия конструкции.

1. For i:=1 to 2 do: - i является счетчиком цикла, именно он определяет количество повторений в цикле.
2. Begin (открываются операторные скобки для того, чтобы обе фразы являлись телом цикла и повторялись вместе.)
3. Writeln (‘Как дела?’): - слово writeln означает вывод фразы, находящейся в одинарных кавычках.
4. Writeln (‘Хорошо, спасибо').
5. End.
6. i:=i + 1.

Как видно, достаточно легко и даже интересно использовать основные алгоритмические конструкции. Базовые алгоритмы действительно широко известны, без них невозможно писать программы.



**Цикл с постусловием**

Цикл с постусловием может повторять неопределенное количество действий без вставки в них операторных скобок или составных слов. Он обязательно выполнится хотя бы один раз. Работает цикл, пока условие является ложным. Прекращается он при становлении показателей правильными. На этом построен алгоритм. Основные алгоритмические конструкции такого типа работают именно в данном темпе.

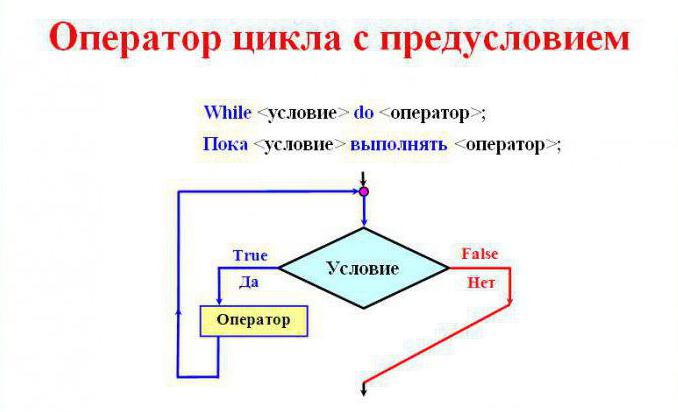
Для реализации этого цикла необходима конструкция Repeat A until B. Дословно она переводится как «повторять действия, пока условие ложно». Соответственно, через А выражен сам процесс повторения, через В – данные, которые в результате должны принять правильное значение.



**Цикл с предусловием**

Цикл с постусловием строится таким образом, что он выполняется минимум один раз в любом случае. Однако имеются такие случаи, когда цикл необходим в случае того или иного условия, а при его отсутствии повторения осуществляться не должны. Иначе результат будет неверен. Именно в таком случае используется цикл с предусловием. Для его создания необходима конструкция «while A do B». Первая команда дословно переводится как «пока». А – условие, а В – действия, которые будут повторяться. Вся конструкция означает: «пока условие является верным, выполнять действия».

Все основные алгоритмические конструкции работают лишь в определенных случаях. Какие же они в цикле с предусловием? Если необходимо, чтобы повторялось не одно действие, а сразу несколько, то стоит использовать или составные операторы, или специальные скобки. Цикл вполне может не выполниться, если при вхождении в него условие не является верным. Соответственно, повторяться действия будут, если оно правильное.



**Вспомогательный алгоритм**

Вспомогательный алгоритм используется в других процессах при помощи указания лишь его имени. Он к основным алгоритмическим конструкциям не относится. В языках программирования такой процесс действий называется подпрограммой. Для облегчения работы с кодом и впоследствии более простого решения задач каждое действие объединяется в один блок, который и является вспомогательным алгоритмом. Каждому из них можно задать свое имя, что позволяет впоследствии неоднократно обращаться к нему.