# 18. Основные моменты.

Прорезюмируем основные аспекты по отдельным главам.

## *18.1. Из чего состоит компьютер.*

- Компьютер состоит из центрального процессора (ЦП), памяти и различных устройств ввода/вывода. Информация между отдельными устройствами передается через системную шину.
- Центральный процессор состоит из контрольного устройства и арифметико-логического устройства (АЛУ). Контрольное устройство обеспечивает взаимодействие и функционирование периферийных устройств и передает информацию между составными элементами компьютера. Арифметико-логическое устройство выполняет вычисления.
- Память компьютера состоит из двух частей: оперативной и внешней.
- Оперативная память (ОП) обладает быстродействием и ограниченной вместимостью. ЦП имеет доступ только к оперативной памяти. ОП не сохраняет информацию после отключения питания. Оперативная память состоит из последовательности пронумерованных ячеек, называемых байтами. Каждый байт, в свою очередь, содержит восемь битов. Номер, присоединенный к байту, является его адресом.
- Внешняя (вспомогательная) память имеет низкое быстродействие и фактически неограниченную вместимость. Она сохраняет информацию после отключения питания. Доступ к информации на внешней памяти имеет только ЦП, если она предварительно перемещена в оперативную память.
- Информация внутри компьютера и на внешней памяти представлена в двоичной системе счисления, в которой используются только нули (0) и единицы (1).
- Программы, запускаемые компьютером, размещаются в оперативной памяти. ЦП вызывает каждую инструкцию по очереди и выполняет ее.
- Каждый ЦП имеет свой собственный машинный язык – набор инструкций, которому он подчиняется.

- Операционная система (ОС) предоставляет интерфейс между пользователем и компьютером, и управляет функционированием компьютера и его периферийными устройствами. ОС также организует файлы пользователей и системные утилиты, такие, как компиляторы, редакторы, прикладные пакеты и др.
- Компьютерные сети позволяют одновременный доступ многих пользователей к компьютерным средствам, общим базам данных, а также использовать такие средства, как электронная почта.
- Файлы хранятся на устройствах внешней памяти в виде иерархической структуры каталогов. Каждый пользователь имеет свой начальный каталог. Средства операционной системы обеспечиваются системной библиотекой.

## *18.2. Языки программирования*

- Каждый центральный процессор обладает собственным внутренним машинным языком. Программирование на машинном уровне осуществляется на языке ассемблера. Он специфичный для каждого компьютера и каждая его инструкция однозначно соответствует инструкции машинного языка.
- Языки высокого уровня транслируются на внутренний машинный язык с помощью специальных компиляторов. Откомпилированный код присоединяется к требуемым системным библиотекам и в таком виде конечная программа загружается в память компьютера, откуда она может быть запущена на выполнение.
- Если язык высокого уровня соответствует определенным принятым стандартам, то соответствующая программа, после компиляции, может быть запущена на любом компьютере. Эта «переносимость» является очень важным преимуществом языков высокого уровня.

## *18.3. Подготовка компьютерной программы, алгоритмы.*

- До составления непосредственно компьютерной программы постановка и цели прикладной задачи должны быть четко

сформулированы.

- Точный набор инструкций для решения задачи называется алгоритмом. Первая стадия составления компьютерной программы заключается в формулировке алгоритма решения.
- Алгоритм должен быть конечным, однозначным и эффективным.
- Последовательность, выбор и повтор составляют основные управляющие структуры алгоритма.
- Составное утверждение позволяет рассмотреть группу утверждений как одно целое.
- Процесс определения значения величины называется присвоением.
- Управляющая структура выбора позволяет прервать выполнение очередного утверждения на основе определенного условия.
- Управляющая структура повтора позволяет выполнить несколько раз одно или группу утверждений.
- Алгоритм должен быть переведен на подходящий язык программирования высокого уровня. Затем программа в таком виде компилируется, присоединяется к системным библиотекам и загружается в память. На этапе компиляции выявляются синтаксические ошибки, которые надо исправлять, чтобы программа смогла заработать.
- После компиляции, присоединения и загрузки программа тестируется с помощью пробных начальных данных. Здесь могут выявиться логические ошибки, ведущие либо к неправильным конечным результатам, либо к приостановке программы из-за ошибок прогона.

## *18.4. Структура программы на Паскале и некоторые простые операторы.*

- Все выполняемые операторы в Паскале разделяются друг от друга символом точка с запятой.
- Комментарии игнорируются компилятором, но играют важную роль для описания назначения программы, ее читабельности. Все символы между фигурными скобками «{ }» воспринимаются компилятором как комментарии.

- Все переменные и константы, которые используются в Турбо-Паскалевской программе, должны быть предварительно объявлены. Объявление для переменной определяет идентификатор и тип, а для константы – идентификатор и значение.
- Тип Integer используется для представления целых чисел. Их значения в компьютере представляются точно, без приближений.
- Тип Real используется для представления действительных чисел (т.е. чисел с десятичной точкой). Их значения в компьютере представляются только приближенно.
- Тип Char используется для представления одного символа. Символьная константа заключается в одинарные апострофы.
- Идентификаторы (имена переменных или констант) могут быть составлены только из букв латинского алфавита, цифр и знаков подчеркивания, и должны начинаться с буквы. Регистр для букв не различается компилятором.
- Выражения являются комбинацией операторов и операнд.
- Порядок выполнения операторов определяется по их приоритету.
- В операторе присваивания сначала вычисляется значение выражения в его правой части, а затем присваивается переменной в его левой части.

## *18.5. Операторы ввода и вывода.*

- Переменным значения присваиваются либо с помощью оператора присвоения «:=», либо оператором Read(идентификатор-переменной).
- Строки символов, заключенные в одинарные апострофы, могут использоваться в операторах вывода.
- Значения переменных или констант могут быть выведены на экран с помощью оператора вывода Write(список-имён) или Writeln(список-имён).

## *18.6. Структура управления с условием.*

- Наилучший способ конструирования решения задачи – принцип сверху-вниз. Сначала определяются подзадачи, которые, в свою очередь, решаются по тому же принципу.
- «Условие» – это логическое выражение, которое может быть либо истинным (True), либо ложным (False).
- Условие (логическое выражение) формируется с помощью символов/операторов логических отношений – больше, меньше, равно. неравно, больше или равно, меньше или равно, в общем случае может содержать арифметические выражения.
- Логическое выражение может принимать только значения «истинно» или «ложно».
- Сложные логические выражения формируются с помощью логических связок: НЕТ (Not), И (And), ИЛИ (Or), исключающее ИЛИ (Xor). Значение сложного логического выражения вычисляется соответственно таблице истинности.
- Условные операторы выполняют одно или другое программное указание в зависимости от значения истинности условия.
- Оператор «If» выполняет послестоящий оператор только в том случае, если условие истинно.
- После зарезервированного слова «Then» может стоять только один оператор – простой или составной, который, в свою очередь, в общем случае, может содержать другие условные операторы «If».
- Оператор «If-Else» используется в случаях, когда нужно выбрать между двумя операторами, которые будут выполняться альтернативно в зависимости от истинности условия.
- Вложенные «If» и «If-Else» операторы используются в случаях, когда нужно сделать выбор между более чем двумя альтернативами.
- Во вложенных «If-Else» операторах каждый «Else» относится к ближайшему предыдущему «If» без «Else».
- Оператор «Case» позволяет сделать выбор из множества альтернатив (больше чем двух), и выбор этот зависит от значения «селектора», который может быть величиной любого скалярного порядкового типа Integer (или другого

целочисленного), Char, Boolean, а также перечисляемого или интервального типа.

- Если значение селектора не совпадает ни с одной меткой вариантов, тогда выполняется оператор после «Else», если он присутствует.
- Метка «Else» – необязательный атрибут. Если она отсутствует, а селектор не совпадает ни с одной определенной меткой, то оператор «Case» будет игнорироваться.

## *18.7. Структура управления с повтором.*

- При формировании алгоритмов важным свойством является способность обеспечить повторное выполнение группы опера – торов.
- Повторное выполнение группы операторов называется циклом, а сами операторы, составляющие группу, – телом цикла.
- Существует три основных вида цикла: For, While и Repeat-Until.

**For**

- Структура повтора и оператор «For» используется для реали – зации циклов, когда количество повторов известно наперед. Это количество зафиксировано перед входом в цикл.
- Параметр цикла последовательно пробегает все значения из заданного диапазона — упорядоченного множества альтерна-тив.
- Если диапазон значений для параметра цикла задан неправильно, цикл не выполнится вообще.
- Тело цикла может быть либо простым, либо составным оператором, следовательно, в последнем случае группа операторов должна быть заключена в программные скобки «Begin End».

**While**

- Структура повтора и оператор «While» являются самым важным. Практически любой цикл можно реализовать с его помощью. Оператор повторно выполняет тело цикла до тех пор, пока логическое условие управления циклом остается истинным.

- Так как выполняемость цикла контролируется вручную, надо внимательно следить, чтобы все параметры, составляющие логическое условие, были определены до начала цикла. Кроме того, хотя бы один из этих параметров должен изменять свое значение в теле, иначе может возникнуть зацикление.
- Тело цикла, как и в случае «For», может быть либо простым, либо составным оператором.
- Если контрольное логическое условие ложно изначально, тело цикла не выполнится вообще.

**Repeat**

- Структура повтора и оператор «Repeat-Until» выполняет тело цикла до тех пор, пока какое-либо условие прерывания цикла не примет значение «истинно». Синтаксическая структура этого цикла такая, что она всегда выполняется по крайней мере один раз.
- Здесь так же, как и в случае оператора «While», выполняемость цикла контролируется вручную, и, следовательно, всё, что было сказано для данной структуры в этом аспекте, остается в силе и для оператора «Repeat».

## *18.8. Дизайн «сверху–вниз», модульное программирование, процедуры и функции.*

- Подход к решению проблемы «сверху-вниз» разбивает ее на несколько подзадач, которые, в свою очередь, могут быть разбиты на меньшие подзадачи. Как только подзадача становится приемлемо простой для решения, она может быть реализована в виде подпрограммы.
- Подпрограммы позволяют повторное использование программного кода.
- Подпрограмма должна быть полностью независимой. Это означает, что при ее вызове она должна соприкасаться с главной программой только через свои параметры. Программист, который использует подпрограмму, должен знать только, что она делает, но не обязан разбираться в деталях, как она работает.

- Входные данные передаются подпрограмме с помощью списка параметров. Каждому параметру назначается тип.
- Когда подпрограмме передается параметр-значение, она работает не с фактическим элементом главной программы, а с его копией, и, следовательно, значение оригинала не может быть изменено.
- Любая переменная, объявленная внутри подпрограммы, является локальной. Она обладает смыслом и значением только в пределах этой подпрограммы. Следовательно, для нее можно использовать любой идентификатор, уже объявленный в главной программе или в других блоках.
- Информация из процедуры возвращается с помощью параметров-значений, т.е. параметров-ссылок. Такой параметр в заголовке описания подпрограммы объявляется как ссылка на фактическую переменную с помощью ключевого слова «Var» перед ней. В этом случае процедура использует адрес фактического параметра и может изменить его значение. Таким образом, информация возвращается обратно в главную программу.
- Подпрограмма-функция возвращает значение и ее тип должен совпадать с типом этого значения. Функция может быть только скалярного или строкового типа.

## *18.9. Больше о модульном программировании.*

### Recursion

- Рекурсивные процедуры и функции обычно прямо или косвенно (через другие подпрограммы) вызывают себя. Они используются для упрощения алгоритма решения и в общем случае носят характер математической индукции.
- При косвенной рекурсии, т.е. кода, рекурсивные подпрограммы взаимно вызывают друг друга, код подпрограммы можно привести в конце описательной части программы. В этом случае ее нужно предварительно объявить с помощью ключевого слова «Forward».

## *18.10. Типы данных.*

- На Паскале есть возможность манипулировать с различными типами данных. Выбор типа зависит от поставленной задачи и, естественно, от программиста.

## *18.11. Строки (Strings).*

- Строка – это структурированный тип данных, который представляет упорядоченную последовательность символов. Нулевой элемент строки хранит информацию о текущей фактической длине строки, которую можно восстановить с помощью функции *Ord*.
- Значения строковых величин можно вводить с клавиатуры и выводить на экран дисплея. Также ввод и вывод можно осуществить с файла/в файл.

## *18.12. Массивы (Arrays).*

- Массивы используются для представления совокупности элементов одинакового типа.
- К элементу массива можно обратиться по индексу, который является величиной перечисляемого типа.
- Количество элементов массива строго зафиксировано, оно должно быть известно к моменту их объявления.
- Обращение к элементу массива с индексом вне диапазона ранга вызывает программную ошибку и может привести к непредсказуемым результатам.

## *8.13. Записи (Records).*

- Записи – это такой тип переменной, которая объединяет в себе множество разнотипных переменных, называемых полями.
- Поля записи могут иметь несколько названий, ассоциированных с различными типами.

## *18.14. Множества (Sets).*

- Множество представляет собой объединение однотипных переменных, поддерживающих порядок.

- На множествах определены операции объединения, пересечения, вычитания.

## *18.15. Поток и внешние файлы (Files).*

- Файлы – используются для связи программных переменных с данными на дисковых устройствах.
- В зависимости от обрабатываемых данных, различают файлы типизированные и нетипизированные.

## *18.16. Указатели (Pointers).*

- Указатели используются для прямого обращения к оперативной памяти, что является единственным способом при обработке большого объема данных.
- Несмотря на не совсем привычный и удобный для Паскаля способ работы с данными, указатели дают возможность обращаться к памяти до точности одного байта.