Информатика и информационно-коммуникационные технологии

Сафарьян Ольга Александровна

Лекция 1. (ч1)

ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКАХ ВЫСОКОГО УРОВНЯ.

1. Основные понятия языков программирования

Основные понятия языков программирования

Запись алгоритма на формальном языке называется *программой*.

- *Языки программирования* — это формальные искусственные языки. Как и естественные языки, они имеют алфавит, словарный запас, грамматику и синтаксис, а также семантику.
- *Алфавит* — разрешенный к использованию набор символов, с помощью которого могут быть образованы слова данного языка, никакие другие символы в тексте не допускаются.
- *Синтаксис* — система правил, определяющих допустимые конструкции (фразы, осмысленные предложения) языка программирования из букв алфавита.

Основные понятия языков программирования

- *Семантика* — система правил однозначного толкования каждой языковой конструкции, позволяющих производить процесс обработки данных. Семантика устанавливает, какие последовательности действий описываются теми или иными фразами языка и, в конечном итоге, какой алгоритм определен данным текстом на алгоритмическом языке.

- Взаимодействие синтаксических и семантических правил определяет основные *понятия языка* (синтаксические единицы, конструкции), такие как операторы, идентификаторы, константы, переменные, функции, процедуры и т.д. В отличие от естественных, язык программирования имеет ограниченный запас слов (операторов) и строгие правила их написания, а правила синтаксиса и семантики, как и для любого формального языка, сформулированы явно, однозначно и четко.

Основные понятия языков программирования

Имена (идентификаторы) — обозначения, присваиваемые объектам программы (переменным, массивам, функциям и др.) и используемые для обращения к ним. Как правило, в качестве имён разрешается использовать последовательности алфавитно-цифровых символов, начинающихся не с цифры. Имена также не должны содержать символов, имеющих специальное назначение (пробелов, скобок, знаков операций). Желательно, чтобы имя отражало назначение объекта.

Зарезервированные имена – уже имеющие определённый смысл слова, которые не могут использоваться в иных целях. В каждом языке (иногда даже в разных версиях одного и того же языка) существуют свои особенности присвоения имён и зарезервированные слова. Примеры допустимых и недопустимых имён (для языков Бейсик, Паскаль, Си++) приведены в таблице 1.

Примеры использования имён

Допустимые имена	Недопустимые имена (пояснение, почему недопустимо)
a2	2a (начинается с цифры)
ab	a b (содержит пробел)
ab_2_cd3	ab-2-cd3 (содержит знак арифметической операции «-»)
Basic	C++ (содержит знак арифметической операции «+»)
sinx	sin (зарезервированное имя – встроенная функция)
True2	True (зарезервированное – логическое значение истина)

Основные понятия языков программирования

- *Операция* (англ. *operator*) – способ записи некоторых вычислительных действий. Зачастую операция обозначается всего одним знаком. Охарактеризуем наиболее часто встречающиеся практически во всех языках типы операций:

-*арифметические* операции, обозначаемые обычно знаками * (умножение), *I* (деление), **+** (сложение), **-** (вычитание) и др.;

-*логические* операции НЕ, И, ИЛИ, синтаксис обозначения которых устанавливается языком программирования;

-операции *отношения* (сравнения) – знаки **<** , **>** , **<=** , **>=** , **=** , **<>** ;

-операция *конкатенации* (сцепки, слияния, соединения, склеивания символьных значений друг с другом с образованием одной длинной строки) обычно изображается знаком **+** или **&**.

Основные понятия языков программирования

Данные – формализованные величины, обрабатываемые программой. Имеется три базовых (основных) типа данных: *числа*, *символы* (одиночные, или *строки* – их последовательности, в том числе и пустые, не содержащие ни одного символа) и *логические величины*. Разным типам соответствуют разные операции, которые возможно производить с данными (операндами):

- с числовыми типами возможны арифметические операции;
- с символьными – сравнение и сцепка;
- с логическими – проверка значения на истинность или ложность и т.п.

Как правило, в любом языке имеется базовый набор типов и несколько конструкций, которые позволяют строить новые типы из уже имеющихся. Наборы

Основные понятия языков программирования

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 её содержимому с помощью имени (идентификатора).

Переменная (англ. variable) – область памяти (ячейка, элемент данных, объект), имеющая имя и предназначенная для хранения значения, которое допускается изменять в процессе выполнения программы. А о реальном адресе и способе хранения можно спокойно позабыть. Кроме имени и значения, переменная обычно имеет *тип*.

Тип переменной задает не только множество допустимых операций, применимых к данному типу, но и способ записи информации в ячейки памяти (формат представления) и, соответственно, необходимый для ее хранения объем памяти, а также диапазон допустимых значений. Например, в ячейке памяти длиной в 8 бит (или 1 байт) может храниться 256 различных значений в двоичном коде. В зависимости от типа переменной это значение может быть интерпретировано и как целое число в диапазоне от 0 до 255 (byte) или от -128 до 127 (shortint), и как один из символов в кодировке ASCII (char).

Переменные с указанием их типа можно вводить в программу с помощью специальных команд *описания (объявления)* и соответствующих ключевых слов. Значения переменных можно преобразовать из одного типа в другой в соответствии с соглашениями языка программирования. Такой процесс называется приведением типов.

Если переменные присутствуют в программе на протяжении всего времени ее работы, то их называют *статическими*. Переменные, создающиеся и уничтожающиеся на разных этапах выполнения программы, называют *динамическими*.

Константами или *постоянными* называют данные, значения которых заданы в тексте программы и не изменяются при ее выполнении. Константы, как и переменные, хранятся в памяти и имеют тип. Их можно указывать в тексте программы явно (такие константы называются также *литералами*), или для удобства обозначать идентификаторами, как и переменные. Например, инструкция `Const pi = 3.14` задаёт значение константы `pi`, и это значение во время работы программы изменить нельзя, так как это не переменная. При этом запись `3.14` является литералом. Значения строковых литералов всегда заключаются в кавычки.

Выражения (англ. *expression*) предназначаются для записи необходимых вычислений, состоят из констант, переменных и функций, объединенных знаками операций – *операторами* (в узком смысле). Каждая операция имеет свой приоритет, то есть очерёдность выполнения. Так, значение выражения `2+2*2` будет равно шести. Изменить порядок вычисления можно с помощью круглых скобок. Выражения записываются в виде линейных последовательностей символов (без подстрочных и надстрочных символов, "многоэтажных" дробей и т.д.), что позволяет вводить их в компьютер одной строкой. В зависимости от типа операций различают арифметические, логические и строковые выражения.

Арифметические выражения служат для определения числового значения. Например, выражение `(1+sin(x))/2` содержит константы `1` и `2`, переменную `x`, функцию `sin()` и знаки операций `+`, `/`. Значение этого выражения при `x=0` будет равно `0.5`, а при `x=pi/2` – единице

Логические выражения описывают некоторые условия, которые могут удовлетворяться или не удовлетворяться. Таким образом, логическое выражение может принимать только два значения – "*истина*" или "*ложь*" ("*да*" или "*нет*"). Рассмотрим в качестве примера логическое выражение `x*x + y*y < r*r`, определяющее принадлежность точки с координатами `(x, y)` внутренней области круга радиусом `r` с центром в начале координат. При `x=1, y=1, r=2` значение этого выражения – "истина", а при `x=2, y=2, r=1` – "ложь".

Примеры использования в выражениях некоторых, наиболее часто используемых операторов (обозначений операций), приведены в таблице 2.

Операторы в арифметических и логических выражениях

Операция	Бейсик	Паскаль	Си++	Пример выражения	
				запись	значение
Арифметические операции					
Возведение в степень	^	отсутствует		2^3	8
Умножение, деление	*, /	*, /	*, /	2*2/4	1
Целочисленное деление	\	Div		17 \ 5	3
Остаток от деления (деление по модулю)	mod	Mod	%	17 mod 5	2
Сложение, вычитание	+, -	+, -	+, -	4+2-1	5
Операции сравнения					
равно	=	=	==	1=2	False
не равно	<>	<>	!=	1<>2	True
меньше	<	<	<	1<2	True
больше	>	>	>	1>2	False
меньше или равно	<=	<=	<=	1<=2	True
больше или равно	>=	>=	>=	1>=2	False
Логические (побитовые) операции					
НЕ	Not	Not	! (~)	Not True	False
И	And	And	&& (&)	2 And 3	2
ИЛИ	Or	Or	|| (|)	2 Or 3	3
Исключающее ИЛИ	Xor	Xor	(^)	2 Xor 3	1

Строковые выражения позволяют описывать преобразования, в результате выполнения которых получается значение типа строка (текст, последовательность символов, литер). В строковые выражения могут входить строковые константы, литералы, переменные и функции, объединённые знаками операции конкатенации. Например, `A + B` означает присоединение строки `B` к концу строки `A`. Так, если задано `A = "сидели на "`, `B = "трубе"`, то значением выражения `A + B` будет `"сидели на трубе"`. При `A = "1"` и `B = "2"` значением выражения `A + B` будет `"12"`.

Операторы (*команды*, *инструкции* – англ. *statement*) – наиболее крупное и содержательное понятие языка: каждый оператор представляет собой законченную фразу языка и определяет некоторый вполне законченный этап обработки данных. В состав операторов входят ключевые (зарезервированные) слова, данные, выражения и т.д.

Операторы подразделяются на *исполняемые и неисполняемые*. Неисполняемые операторы предназначены для описания данных и структуры программы, а исполняемые – для выполнения различных действий (например, оператор присваивания, операторы ввода и вывода, условный оператор, операторы цикла и др.).