10. Базовые конструкции языков программирования: условные и безусловные переходы, операторы выбора, циклы, процедуры и функции.

Ссылки на материалы:

1. http://www.ngpedia.ru/id96229p1.html

2. http://dfe.petrsu.ru/koi/posob/c/c.htm (для примера кода на c++)

Операторы

Оператор – допустимое в языке программирования высокого уровня предложение, задающее целостное законченное действие ЭВМ, или представляющее набор описаний. Типичными операторами в традиционных языках программирования являются операторы ввода/вывода, присваивания, перехода, цикла, процедуры и др. Грамматическая конструкция каждого из них определяется синтаксисом конкретного языка программирования. Операторы условно делятся на исполняемые операторы и невыполняемые операторы. Исполняемый оператор представляет одну или несколько последовательных операций, составляющих алгоритм решения задачи. Невыполняемый оператор непосредственно не задает алгоритм, а содержит описания объектов программы либо другую информацию, необходимую для трансляции и выполнения программы.

Оператор безусловного перехода

Оператор безусловного перехода – оператор, не содержащий никакого условия, а непосредственно указывающий, какой оператор должен быть выполнен следующим. В большинстве языков программирования высокого уровня оператор безусловного перехода состоит из слов «GO TO» (Перейти к), за которыми ставится метка. Например, в Фортране оператор безусловного перехода Go to 25 указывает, что следующим должен быть выполнен оператор, помеченный меткой 25. Аналогичную форму имеет оператор безусловного перехода в Паскале и Си. Помеченный оператор может располагаться в программе до или после оператора безусловного перехода. При трансляции программы оператор безусловного перехода заменяется компилятором на команду безусловной передачи управления.

Оператор ввода

Оператор ввода – оператор в программе, предписывающий передачу данных из устройства ввода или внешней памяти в оперативную память и делающий эти данные доступными программе. В языке программирования может быть несколько конструкций оператора ввода. Вид этих операторов, а также информация, которая должна быть указана транслятору этими операторами, различны и определяются синтаксисом конкретного языка программирования и особенностями внешних запоминающих устройств и устройств ввода. Поэтому в некоторых языках программирования высокого уровня (например, в Си) операторы ввода не входят в состав основных определений языка. Их разработка возложена на программистов, проектирующих компиляторы систем программирования. Однако для обеспечения переносимости программ в различных системах программирования,

ориентированных на один язык, применяется единый стандартный набор процедур или функций ввода. Обычно в операторе ввода задается список ввода – перечень переменных и массивов, значения которым должны быть присвоены в процессе ввода данных, и номер устройства или имя файла, из которого данные читаются.

Оператор возврата

Оператор возврата – оператор, завершающий выполнение процедуры или функции и передающий управление вызывающей программе. Как правило, оператор возврата обозначается ключевым словом Return (возврат) и располагается в теле процедуры или функции. Обычно оператор возврата передает управление либо оператору вызывающей программы, в котором содержался вызов функции (в случае функции), либо оператору, следующему за оператором вызова процедуры (в случае процедуры). В некоторых языках программирования (например, в Фортране) существует конструкция оператора возврата, позволяющая передавать управление любому помеченному оператору вызывающей программы. Для этого его метка должна быть указана как фактический параметр в операторе вызова подпрограммы. В некоторых языках программирования (например, в Си) с помощью оператора возврата имени функции может присваиваться значение, возвращаемое в вызывающую программу как результат вычисления функции.

Оператор выбора

Оператор выбора – оператор в программе, определяющий выбор одной из нескольких ветвей алгоритма. Например, с помощью оператора выбора программируется меню (выбор пользователем последовательности действий в соответствии с пунктом меню). Оператор выбора содержит перечень альтернативных операторов, в которых определены действия, соответствующие различным ситуациям или условиям. Выбор необходимого варианта осуществляется с помощью указанного в операторе выбора выражения, обычно называемого переключателем. Выражение-переключатель конструируется так, чтобы оно принимало значение, определяющее номер или метку оператора, который нужно выполнить в создавшейся ситуации.

Оператор вывода

Оператор вывода – оператор в программе, предписывающий передачу данных из оперативной памяти во внешнюю память или на устройства вывода. В языке программирования может быть несколько конструкций оператора вывода. Вид этих операторов, а также информация, которая должна быть указана транслятору этими операторами, различны и определяются синтаксисом конкретного языка программирования и особенностями внешних запоминающих устройств и устройств вывода. Поэтому в некоторых языках программирования высокого уровня (например, в Си) операторы вывода не входят в состав основных определений языка. Их разработка возложена на программистов, проектирующих компиляторы систем программирования. Однако для обеспечения переносимости программ в различных системах программирования, ориентированных на один и тот же язык, применяется единый стандартный набор процедур или функций вывода. Обычно в операторе вывода задается список вывода – перечень переменных и массивов,

значения которых должны быть выведены, имя или номер устройства, либо имя файла, в который записываются данные. Кроме того, указывается формат выводимых данных.

Оператор присваивания

Оператор присваивания – оператор передающий значение арифметического, логического или другого выражения одной или нескольким переменным, либо имени функции. Действие присваивания обычно обозначается символами = (например, в языках Фортран и Си) и := (в языке Паскаль). Слева от символа присваивания пишется имя переменной (или имена переменных), которой должно быть присвоено значение выражения, стоящего справа. При выполнении оператора присваивания вначале вычисляется значение выражения, а затем оно передается переменной. Тип переменной, стоящей слева в операторе присваивания, может не совпадать с типом выражения, стоящего справа. Соответствующее преобразование типов обычно автоматически закладывается в объектную программу при компиляции. Однако во избежание возможной потери информации при преобразовании типов желательно, чтобы тип переменной и тип выражения были одинаковыми.

Оператор процедуры

В языках программирования – оператор, вызывающий процедуру. Выполнение оператора процедуры эквивалентно такой последовательности действий: передача процедуре фактических параметров, передача управления на вход в подпрограмму, выполнение операторов, запрограммированных в теле процедуры, и возврат управления вызывающей программе. Правила записи оператора процедуры определяются синтаксисом конкретного языка программирования. Наиболее распространенный вид оператора процедуры: Name (x1, x2, ..., xn) или Name, где Name – имя вызываемой процедуры или идентификатор дополнительной точки входа в процедуру; xl, x2, ..., xn – фактические параметры, передаваемые процедуре, согласующиеся по количеству, порядку следования, классу и типу с соответствующими формальными параметрами.

Оператор условного перехода

Оператор условного перехода – оператор, содержащий языковую конструкцию, реализующую проверку условия, при котором изменяется естественная последовательность выполнения операторов в программе. Оператор условного перехода применяется при ветвлении программы, когда возникает необходимость осуществить переход в зависимости от результата проверки некоторого условия. Операторы условного перехода транслируются компилятором с помощью команд передачи управления.

Оператор-функция

В языке Фортран оператор – функция – оператор оператор, имеющий вид оператора присваивания и определяющий внутри программного модуля функцию, заданную выражением. К этой функции можно обращаться по имени так же, как к стандартным функциям, если необходимо многократно в разных точках программного модуля вычислять значение этой функции при различных значениях ее аргументов. Оператор-функция

располагается в программном модуле перед первым выполняемым оператором, но после всех других невыполняемых операторов.

Оператор цикла

Оператор цикла – оператор, упрощающий программирование цикла, формально состоит из заголовка цикла и тела цикла. Заголовок цикла указывает на последовательность повторяемых операторов, образующих тело цикла, и либо определяет множество значений параметров цикла и предписывает многократное выполнение тела цикла при этих значениях параметров, либо указывает условие повторного выполнения тела цикла. Правила написания оператора цикла задаются синтаксисом конкретного языка программирования и типом программируемого цикла.

Операторы перехода

Операторы перехода – операторы, изменяющие естественную последовательность действий в программе, указывая оператор, который должен быть выполнен следующим. В большинстве языков программирования высокого уровня оператор перехода содержит строку «GO TO» (Перейти к). Существуют операторы безусловного перехода, предписывающие переход в заданную точку программы без проверки выполнения каких-либо условий. При ветвлении программы возникает необходимость осуществить переход в зависимости от результата проверки некоторого условия. В этом случае могут применяться операторы условного перехода или другие операторы, явно или неявно включающие в себя языковую конструкцию, реализующую проверку условия. Например, в Паскале оператор if x < 0 then GO TO 25, указывает, что в случае, когда значение переменной x отрицательно, следующим должен выполняться оператор, помеченный меткой 25.

Процедуры и функции

Процедура – конструкция многих языков программирования высокого уровня (например, Паскаля), соответствующая понятию подпрограммы. **Процедура** – поименованная часть программы (блок программы или группа описаний и операторов), которая может выполнять некоторые четко заданные действия над условными данными, определяемыми с помощью формальных параметров. Выполнение процедуры может быть инициировано из любого места программы одним оператором. При вызове процедуры вместо формальных параметров указываются фактические параметры, определяющие конкретные данные, над которыми и выполняются запрограммированные в процедуре действия. Возможны процедуры, в которых нет формальных параметров. В них операторы сразу задают действия над объектами, определенными в главной программе. Процедуры вводятся в программу с помощью описания процедуры, которое обычно располагается в разделе описаний. Описание процедуры состоит из заголовка процедуры и тела процедуры. Заголовок служит для присвоения процедуре некоторого имени и, возможно, указания формальных параметров. В теле программируется выполняемый процедурой алгоритм.

Функция – в языках программирования высокого уровня (например, в Паскале и Си) аналогичная процедуре конструкция, соответствующая понятию подпрограммы. **Функция** – поименованная часть программы (блок программы или группа операторов), результатом

выполнения которой является значение, присваиваемое имени функции, поэтому вызов функции используется в качестве операнда в выражении. Алгоритм функции может быть задан в виде действий над условными данными, определяемыми с помощью формальных параметров. При вызове функции вместо формальных параметров указываются фактические параметры, определяющие конкретные данные, над которыми и выполняются запрограммированные функцией действия. Возможны функции, в которых нет формальных параметров. В них операторы сразу задают действия над объектами программы, определенными в главной программе. В некоторых языках программирования (например, в Си) отсутствует понятие процедуры, а вызов функции может не только употребляться в качестве операнда выражения, но и быть отдельным оператором вызывающей программы. При этом значение, которым в результате вызова обладает имя функции, никуда не передается, а результат выполнения функции может состоять, например, в изменении значений некоторых фактических параметров или глобальных переменных. Аналогичное применение функции допускается и в Паскале. Функции вводятся в программу с помощью описания функции, которое обычно располагается в разделе описаний. Описание функции состоит из заголовка функции и тела функции. Заголовок служит для присвоения функции некоторого имени и, возможно, указания формальных параметров. В теле программируется выполняемый функцией алгоритм. Важную роль в программах играют так называемые встроенные или стандартные функции языка программирования. Они не требуют описаний и автоматически распознаются транслятором.

Функции представляют собой центральный объект программирования на C++. Такие языки программирования, как Паскаль, делают различие между процедурами и функциями, но в C++ функции играют обе роли.

Обычно функции объявляются как прототипы в стандартных или создаваемых пользователем файлах заголовка либо в файлах программы. По умолчанию функции имеют тип extern, и доступ к ним возможен из любого файла программы. Функция может быть ограничена спецификатором класса памяти static. Функции объявляются в исходных файлах либо делаются доступными при компоновке с откомпилированными библиотеками. В C++ всегда пользуются прототипами функции.

Функция может быть объявлена в программе несколько раз при условии, что эти объявления совместимы. Неопределяющие объявления функции, использующие формат прототипа функции, предоставляют компилятору C++ детальную информацию о параметрах, что позволяет лучше управлять числом аргументов, контролем их типа и преобразованиями типов. Объявления, если они имеются, должны соответствовать определению функции. Существенным различием между определением и объявлением является то, что определение содержит собственно тело функции.

Декларатоpы определяют типы каждого параметра функции. Компилятор использует эту информацию для контроля достоверности вызова функции. Компилятор также может приводить аргументы к нужному типу.

Прототипы функций также упрощают документирование кодов программы. Если файл заголовка содержит прототипы функций, его можно распечатать и получить информацию, которая нужна, чтобы написать программу, использующую эти функции.

Определение функции состоит из следующих разделов (грамматика позволяет создание и более сложных конструкций):

– опциональные спецификаторы класса памяти: extern или static. По умолчанию – extern;

– тип возврата, возможно void. Умолчанием является int;

– опциональные модификаторы: pascal, cdecl, interrupt, near, far, huge. Умолчание зависит от модели памяти и установленных опций компилятора;

– имя функции;

– список объявлений параметров (который может быть пустым), заключенный в круглые скобки;

– тело функции, представляющее собой коды, выполняемые при вызове функции.

Список объявления формальных параметров имеет синтаксис, аналогичный синтаксису обычных объявлений идентификаторов.

Функция вызывается с фактическими аргументами, помещенными в той же последовательности, что и соответствующие им формальные аргументы. Преобразование фактических аргументов выполняется, как если бы они инициализировались значениями формальных аргументов при объявлении типов.

Правила, управляющие обработкой в C++ модификаторов языка и формальных параметров при вызове функций, как при наличии прототипа, так и при его отсутствии:

– модификаторы языка для определения функции должны соответствовать модификаторам, используемым в объявлении функции, при всех вызовах функции;

– функция может модифицировать значения своих формальных параметров, но это не влияет на фактические аргументы в вызывающей программе, за исключением аргументов типа «ссылка» в C++;

– при наличии прототипа число аргументов в прототипе и функции должно совпадать (при условии, что в прототипе не задано многоточие);

– типы аргументов должны быть совместимы в такой степени, чтобы операция присвоения выполнялась правильно. Всегда можно использовать явные приведения, чтобы преобразовать аргумент к типу, приемлемому для прототипа функции.