Алгоритмический язык

Понятие алгоритмического языка

- Достаточно распространенным способом представления алгоритма является его запись на **алгоритмическом языке,** представляющем в общем случае систему обозначений и правил для единообразной и точной записи алгоритмов и исполнения их.
- Между понятиями «алгоритмический язык» и «языки программирования» есть различие; прежде всего, под исполнителем в алгоритмическом языке может подразумеваться не только компьютер, но и устройство для работы «в обстановке».
- Программа, записанная на алгоритмическом языке, не обязательно предназначена компьютеру.
- Практическая же реализация алгоритмического языка - отдельный вопрос в каждом конкретном случае.

Понятие алгоритмического языка (2)

- Как и каждый язык, алгоритмический язык имеет свой словарь.
- Основу этого словаря составляют слова, употребляемые для записи команд, входящих в систему команд исполнителя того или иного алгоритма. Такие команды называют **простыми командами**.
- В алгоритмическом языке используют слова, смысл и способ употребления которых задан раз и навсегда. Эти слова называют **служебными**. Использование служебных слов делает запись алгоритма более наглядной, а форму представления различных алгоритмов - **единообразной**.

Свойства алгоритма

- При построении новых алгоритмов могут использоваться алгоритмы, составленные ранее.
- Алгоритмы, целиком используемые в составе других алгоритмов, называют **вспомогательными алгоритмами**.
- Вспомогательным может оказаться любой алгоритм из числа ранее составленных.

Свойства алгоритма

- Очень часто при составлении алгоритмов возникает необходимость использования в качестве вспомогательного одного и того же алгоритма, который к тому же может быть весьма сложным и громоздким.
- Было бы нерационально, начиная работу, каждый раз заново составлять и запоминать такой алгоритм для его последующего использования. Поэтому в практике широко используют, так называемые, встроенные (или стандартные) вспомогательные алгоритмы, т.е. такие алгоритмы, которые постоянно имеются в распоряжении исполнителя.
- Обращение к таким алгоритмам осуществляется так же, как и к «обычным» вспомогательным алгоритмам.

Свойства алгоритма

- Алгоритм может содержать обращение к самому себе как вспомогательному и в этом случае его называют **рекурсивным.**
- Если команда обращения алгоритма к самому себе находится в самом алгоритме, то такую рекурсию называют **прямой**.
- Возможны случаи, когда рекурсивный вызов данного алгоритма происходит из вспомогательного алгоритма, к которому в данном алгоритме имеется обращение. Такая рекурсия называется **косвенной.**

Ветвления алгоритма

- Алгоритмы, при исполнении которых порядок следования команд определяется в зависимости от результатов проверки некоторых условий, называют **разветвляющимися.** Для их описания в алгоритмическом языке используют специальную составную команду - команду **ветвления.** Она соответствует блок-схеме «альтернатива» и также может иметь полную или сокращенную форму.

Итерация

Алгоритмы, при исполнении которых отдельные команды или серии команд выполняются неоднократно, называют **циклическими**. Для организации циклических алгоритмов в алгоритмическом языке используют специальную составную команду цикла. Она соответствует блок-схемам типа «итерация».

История развития языков высокого уровня

- Языки программирования служат разным целям и их выбор определяется удобностью пользователя, пригодностью для данного компьютера и данной задачи.
- Задачи для компьютера бывают самые разнообразные: вычислительные, экономические, графические, экспертные и т.д. Такая разнотипность решаемых компьютером задач и определяет многообразие языков программирования.
- В программировании наилучший результат достигается при индивидуальном подходе, исходящем из класса задачи, уровня и интересов программиста.

Области применения языков высокого уровня

- Бейсик широко употребляется при написании простых программ;
- Фортран является классическим языком программирования при решении на ЭВМ математических и инженерных задач;
- язык Кобол был задуман как основной язык для массовой обработки данных в сферах управления и бизнеса.
- язык ЛОГО создан для обучения программированию школьников
- Пролог разработан как язык программирования для создания систем искусственного интеллекта.
- и т.д., и т.п.

История развития языков высокого уровня

- Классическое операциональное и/или процедурное программирование требует от программиста детального описания того, как решать задачу, т.е. формулировки алгоритма и его специальной записи. При этом ожидаемые свойства результата обычно не указываются. Основные понятия языков этих групп - оператор и данные.
- При процедурном подходе операторы объединяются в группы - процедуры.
- Структурное программирование в целом не выходит за рамки этого направления, оно лишь дополнительно фиксирует некоторые полезные приемы технологии программирования.

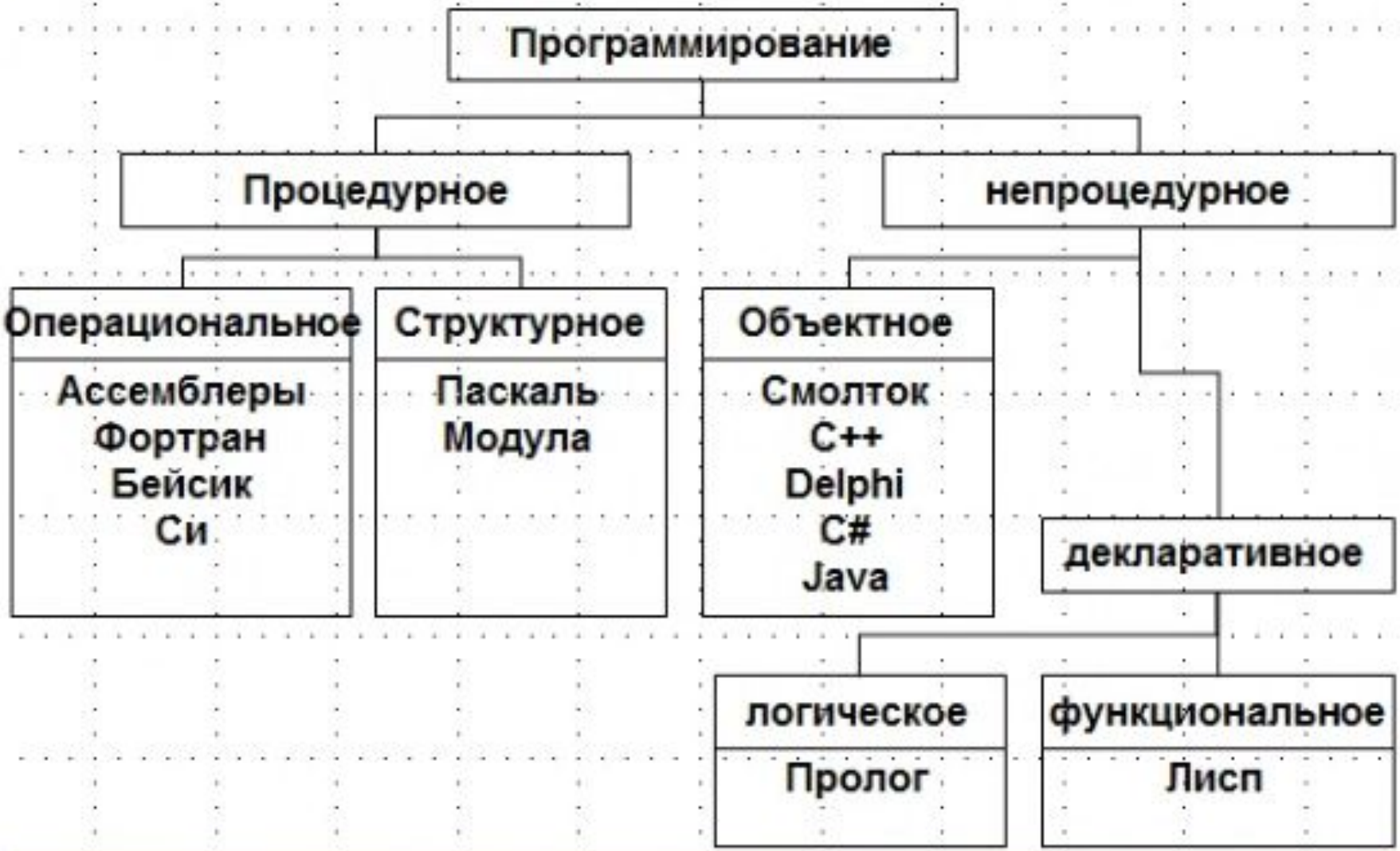
История развития языков высокого уровня

- Другое направление в программировании связано с **методологиями непроцедурного программирования**.
- К ним можно отнести **объектно-ориентированное** и **декларативное** программирование.
- Объектно-ориентированный язык создает окружение в виде множества независимых объектов. Каждый объект ведет себя подобно отдельному компьютеру, их можно использовать для решения задач как «черные ящики», не вникая во внутренние механизмы их функционирования. Из языков объектного программирования, популярных среди профессионалов, следует назвать прежде всего Си++.

Классификация языков программирования

- При использовании декларативного языка программист указывает исходные информационные структуры, взаимосвязи между ними и то, какими свойствами должен обладать результат. При этом процедуру его получения («алгоритм») программист не строит (по крайней мере, в идеале). В этих языках отсутствует понятие «оператор» («команда»).
- Декларативные языки можно подразделить на два семейства - логические (типичный представитель - Пролог) и функциональные (Лисп).

Классификация языков программирования



Языки программирования высокого уровня

Основные понятия

Основные понятия

- **Языки программирования** – это формальные языки специально созданные для «общения» человека с компьютером. Каждый язык программирования, равно как и «естественный» язык (русский, английский и т.д.), имеет алфавит, свои грамматику и синтаксис, а также семантику.
- **Алфавит** – фиксированный для данного языка набор основных символов, допускаемых для составления текста программы на этом языке.
- **Синтаксис** – система правил, определяющих допустимые конструкции языка программирования из букв алфавита.
- **Семантика** – система правил однозначного толкования отдельных языковых конструкций, позволяющих воспроизвести процесс обработки данных.

Основные понятия

- Взаимодействие синтаксических и семантических правил определяют те или иные понятия языка, например, операторы, идентификаторы, переменные, функции и процедуры, модули и т.д.
- В отличие от естественных языков правила грамматики и семантики для языков программирования, как и для всех формальных языков, должны быть явно, однозначно и четко сформулированы.
- Языки программирования, имитирующие естественные языки, обладающие укрупненными командами, ориентированными на решение прикладных содержательных задач, называют **языками «высокого уровня»**.

Достоинства языков программирования высокого уровня

- алфавит языка значительно шире машинного, что делает его гораздо более выразительным и существенно повышает наглядность и понятность текста;
- набор операций, допустимых для использования, не зависит от набора машинных операций, а выбирается из соображений удобства формулирования алгоритмов решения задач определенного класса;
- конструкции команд (операторов) отражают содержательные виды обработки данных и задаются в удобном для человека виде;
- используется аппарат переменных и действия с ними;
- поддерживается широкий набор типов данных.

Метаязыки описания языков программирования

- Интерпретация конструкций языка программирования должна быть абсолютно однозначной, ибо фраза на языке программирования превращается в машинный код автоматически, с помощью программы-транслятора, и любой намек на неоднозначность либо делает эту фразу непереводимой, либо приводит к ошибке.
- В этом отношении языки программирования значительно отличаются от естественных языков, допускающих неоднозначно интерпретируемые фразы.

Метаязыки описания языков программирования

- Для строгого и точного описания синтаксиса языка программирования, как правило, используют специальные **метаязыки** (языки для описания других языков).
- Наиболее распространенными метаязыками являются **металингвистические формулы Бэкуса - Наура** (язык БНФ) и **синтаксические диаграммы Вирта.**

Язык БНФ

- **Язык БНФ** (называемый также языком нормальных форм) представляет компактную форму в виде некоторых формул, похожих на математические.
- Для каждого понятия языка существует единственная метаформула (нормальная форма). Она состоит из левой и правой частей.
- В левой части указывается определяемое понятие, а в правой - задается множество допустимых конструкций языка, которые объединяются в это понятие.
- В формуле используют специальные метасимволы в виде угловых скобок, в которых заключено определяемое понятие (в левой части формулы) или ранее определенное понятие (в ее правой части), а разделение левой и правой частей указывается метасимволом «::=», смысл которого эквивалентен словам «по определению есть».

Язык БНФ (пример метаформул)

<переменная>::=A|B

<выражение>::=<переменная>|<переменная>+<переменная>|
<переменная>-<переменная>

означают, что в том языке, на который эта метаформула распространяется, под термином <переменная> понимается любая из букв A или B, а под термином <выражение> - любая из следующих десяти записей: A; B; A+A; A+B; B+A; B+B; A-A; A-B: B-A; B-B.

Знак | следует читать «или».

Язык БНФ (пример понятия «двоичный код»)

Правая часть метаформулы может содержать правило построения допустимых последовательностей. Допускаются рекурсивные определения терминов и понятий, т.е. когда в правой части формулы участвует понятие, определяемое левой частью.

Например, пусть необходимо ввести понятие <двоичный код>, под которым понимается любая непустая последовательность цифр 0 и 1. Тогда простое и компактное рекурсивное определение с помощью метаформул выглядит так:

<двоичная цифра>::= 0|1

<двоичный код>::=<двоичная цифра>|<двоичный код> <двоичная цифра>

Язык БНФ (пример понятия «двоичный код»)

Для задания синтаксических конструкций произвольной длины часто используют фигурные скобки как метасимволы. Фигурные скобки означают, что конструкция может повторяться нуль или более раз. В частности, термин <двоичный код> можно определить по другому, а именно:

<двоичный код>::=<двоичная цифра>{<двоичная цифра>}

Для полноты множества синтаксических конструкций, необходимо определить конструкцию <пусто>:

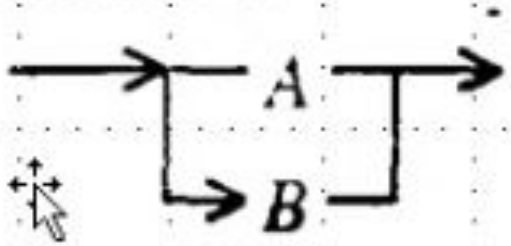
<пусто>::=.

Синтаксическая диаграмма

- **Синтаксическая диаграмма** является графическим представлением значения метапеременной метаязыка.
- Диаграмма состоит из основных символов или понятий языка.
- Каждая диаграмма имеет входящую и выходящую стрелки, означающие начало и конец синтаксической конструкции и отражающие процесс ее чтения и анализа.
- Из каждого элемента выходит одна или несколько стрелок, указывающих на те элементы, которые могут следовать непосредственно за данным элементом.

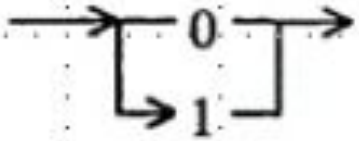
Синтаксическая диаграмма

- <переменная>:: =

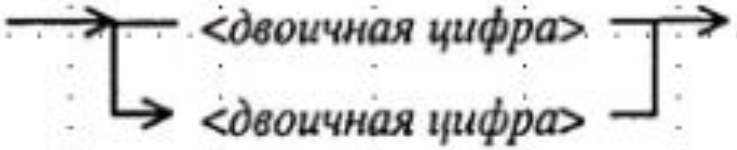


Запись эквивалентна метаформуле <переменная>::= A|B.

<двоичная цифра>::=



<двоичный код>::=



Грамматика языка программирования

- Описанию грамматики языка предшествует описание его алфавита.
- Алфавит любого языка состоит из фиксированного набора символов, однозначно трактуемых.
- Алфавит языков программирования, как правило, связан с литерами клавиатуры печатной машинки. Клавиатуры персональных компьютеров близки к ним по наличию литер.
- Алфавиты большинства языков программирования близки друг другу и основываются на буквах латинского алфавита, арабских цифрах и общепринятых спецсимволах, таких как знаки препинания, математических операций, сравнений и обозначений.

Элементы алфавита

<буква>::= AaBbCcDdEeFf и т.д.

<цифра>::= 0123456789

<знак арифметической операции >::= */+-

<разделитель>::=.,;:()[]{}':=

<служебное слово>:: = begin end if then else for next и т.д.

<спецсимвол>::= <знак арифметической операции> | <разделитель> | <служебное слово>

<основной символ>::= <буква> | <цифра> | <спецсимвол>

<комментарий>::= <любая последовательность символов>

Фундаментальных понятия языка

Оператор - одно из ведущих понятий всех языков программирования.

Каждый оператор представляет собой законченную фразу языка и определяет однозначно трактуемый этап обработки данных.

В соответствии с теорией алгоритмов выделяют **основные операторы** языка: присвоения, условный и безусловный переход, пустой оператор.

К **производным**, не основным, относят составной оператор, оператор выбора, оператор цикла и оператор присоединения.

Фундаментальных понятия языка

- **Величины** могут быть **постоянными** и **переменными.** Значения постоянных величин не изменяются в ходе выполнения программы. Величина характеризуется **типом, именем и значением.** Наиболее распространенные типы величин - числовые (целые и вещественные), символьные, логические. Тип величины определяется ее значением.
- Другая важная классификация величин - ***простые*** и ***структурированные***.
- **Простая** величина в каждый момент может иметь не более одного значения.
- **Структурированная** величина, имея одно имя, может иметь разом несколько значений. Эти значения представляют собой элементы величины.

Структурированная величина

- Важнейшие характеристики структурированной величины:
- **упорядоченность** (да или нет),
- **однородность** (да или нет),
- **способ доступа** к элементам,
- **фиксированность числа элементов** (да или нет).

Так, массив является упорядоченной однородной структурой с прямым доступом к элементам и фиксированным их количеством.

Идентификатор

Всем программным объектам в языках даются индивидуальные **имена.** Имя программного объекта называют **идентификатором** (от слова «идентифицировать»). Чаще всего идентификатором является любая конечная последовательность букв к цифр, начинающаяся с буквы:

<идентификатор>::=<буква> | <идентификатор> | <буква> <идентификатор><цифра>

Многим слово «идентификатор» не нравится, и в настоящее время чаще употребляют слово «имя», поскольку

<имя>::=<идентификатор>.

Объявление переменных

- **Описания** или объявления программных объектов связаны с правилами обработки данных. Данные бывают разные и необходимо для каждого из них определить его свойства. Например, если в качестве данных выступает массив, то необходимо задать его размерность, границы индексов, тип элементов массива. Описательная часть языка программирования является необходимой как для системных программистов - разработчиков трансляторов, которые должны, в частности, проводить синтаксическую и семантическую диагностику программ, - так и для «прикладного» программиста, которому объявления программных объектов часто облегчают процесс разработки и отладки программ.

Список наиболее употребительных обозначений типов данных, используемых в описаниях

- *Целый - Integer*
- *Вещественный - Real*
- *Логический - Boolean*
- *Символьный - Char*
- *Строковый - String*
- *Массив - Array*
- *Множество -Set*
- *Файл - File*
- *Запись - Record*
- *Объект - Object*

Переменная

Понятие «переменная» в языках программирования отличается от общепринятого в математике.

Переменная - это программный объект, способный принимать некоторое значение с помощью оператора присваивания. В ходе выполнения программы значения переменной могут неоднократно изменяться. Каждая переменная после ее описания отождествляется с некоторой ячейкой памяти, содержимое которой является ее значением.

Функция

- **Функция** - это программный объект, задающий вычислительную процедуру определения значения, зависимого от некоторых аргументов. Вводится в языки программирования для задания программистом необходимых ему функциональных зависимостей. В каждом языке высокого уровня имеется в наличии библиотека стандартных функций: арифметических, логических, символьных, файловых и т.п.

Процедура

- **Процедура** - это программный объект, представляющий некоторый самостоятельный этап обработки данных.
- Процедуры явились преемниками подпрограмм, которые были введены для облегчения разработки программ еще на самых ранних стадиях формирования алгоритмических языков.
- Процедура имеет входные и выходные параметры, называемые **формальными**.
- При использовании процедуры формальные параметры заменяются на **фактические**.

Модуль

- **Модуль** (Unit) - это специальная программная единица, предназначенная для создания библиотек и разделения больших программ на логически связанные блоки.
- Модуль - это набор констант, типов данных, переменных, процедур и функций.
- В состав модуля входят разделы: заголовок, интерфейс, реализация, инициализация.

Языки высокого уровня

- Паскаль
- Бейсик
- Фортран
- Си и Си++
- Пролог
- Лисп
- Delphi
- Java
- C#