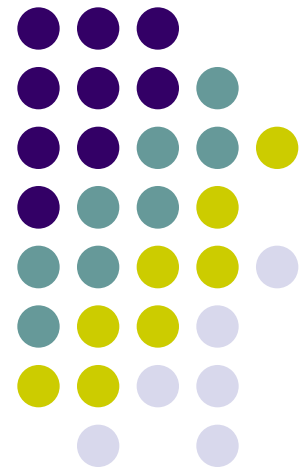
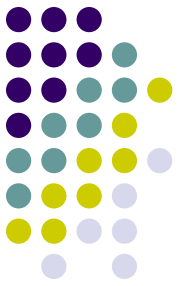


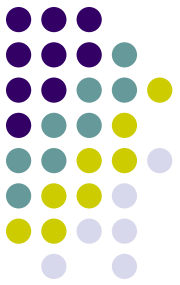
Программирование



Язык программирования



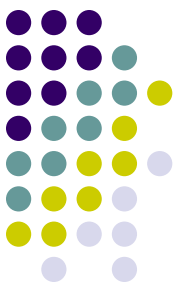
Язык программирования — формальная знаковая система, предназначенная для записи программ. Программа обычно представляет собой некоторый алгоритм в форме, понятной для исполнителя (например, компьютера). Язык программирования определяет набор лексических, синтаксических и семантических правил, используемых при составлении компьютерной программы. Он позволяет программисту точно определить то, на какие события будет реагировать компьютер, как будут храниться и передаваться данные, а также какие именно действия следует выполнять над этими данными при различных обстоятельствах.



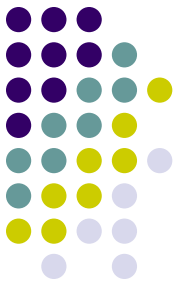
Язык программирования

- *Функция:* язык программирования предназначен для написания компьютерных программ, которые применяются для передачи компьютеру инструкций по выполнению того или иного вычислительного процесса и организации управления отдельными устройствами.
- *Задача:* язык программирования отличается от естественных языков тем, что предназначен для передачи команд и данных от человека компьютеру, в то время, как естественные языки используются для общения людей между собой. В принципе, можно обобщить определение «языков программирования» — это способ передачи команд, приказов, четкого руководства к действию; тогда как человеческие языки служат также для обмена информацией.
- *Исполнение:* язык программирования может использовать специальные конструкции для определения и манипулирования структурами данных и управления процессом вычислений.

Компилируемые и интерпретируемые языки



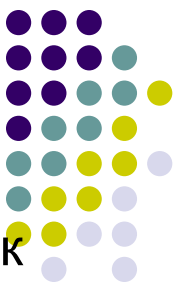
- Программа на компилируемом языке при помощи специальной программы компилятора преобразуется (компилируется) в набор инструкций для данного типа процессора (машинный код) и далее записывается в исполняемый файл, который может быть запущен на выполнение как отдельная программа. Другими словами, компилятор переводит программу с языка высокого уровня на низкоуровневый язык, понятный процессору.
- Интерпретатор непосредственно выполняет (интерпретирует) ее текст без предварительного перевода. При этом программа остается на исходном языке и не может быть запущена без интерпретатора. Можно сказать, что процессор компьютера — это интерпретатор машинного кода.



Языки программирования

- Низкого уровня
- Высокого уровня

Языки программирования низкого уровня



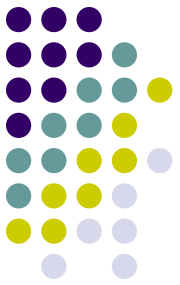
- Низкоуровневый язык программирования – язык программирования, близкий к программированию непосредственно в машинных кодах. Низкоуровневые языки, как правило, используют особенности конкретного семейства процессоров. Низкоуровневым языком является язык ассемблера (от английского assembler - сборщик) - названия транслятора (компилятора) с языка ассемблера. Язык ассемблера, часто для краткости неверно называют "ассемблером". Команды языка ассемблера один в один соответствуют командам процессора и фактически, представляют собой удобную символьную форму записи команд и аргументов. Обычно программы или участки кода пишутся на низкоуровневом языке ассемблера в случаях, когда разработчику критически важно оптимизировать такие параметры, как быстродействие (например, при создании драйверов устройств) и размер кода (загрузочные сектора, программное обеспечение различных устройств, вирусы, навесные защиты и т.д.).



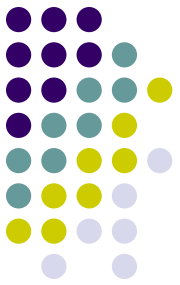
Высокоуровневый язык программирования



Высокоуровневый язык программирования — язык программирования, разработанный для быстроты и удобства использования программистом. Основная черта высокоуровневых языков — это абстракция, то есть введение смысловых конструкций, кратко описывающих такие структуры данных и операции над ними, описания которых на машинном коде (или другом низкоуровневом языке программирования) очень длинны и сложны для понимания.

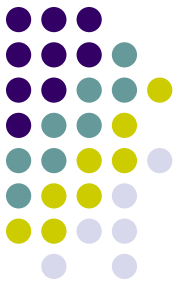


Использование разнообразных трансляторов и интерпретаторов обеспечивает связь программ, написанных при помощи языков высокого уровня, с различными операционными системами и оборудованием, в то время как их исходный код остаётся, в идеале, неизменным.



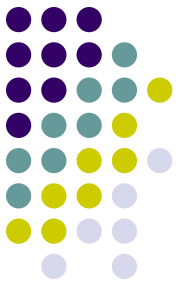
Минусы:

- не позволяет создавать простые и точные инструкции к используемому оборудованию.
- Программы, написанные на языках высокого уровня, проще для понимания программистом, но менее эффективны, чем их аналоги, создаваемые при помощи низкоуровневых языков.



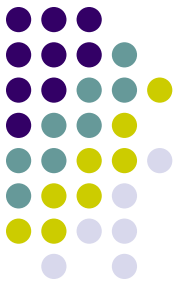
Следствие

Добавление поддержки того или иного языка низкого уровня (язык ассемблера) в ряд современных профессиональных высокоуровневых языков программирования.



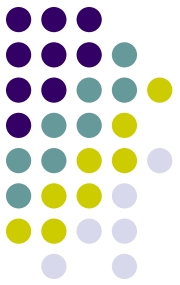
Первым языком программирования высокого уровня считается компьютерный язык Plankalkül разработанный немецким инженером Конрадом Цузе ещё в период 1942—1946 гг. Однако, широкое применение высокоуровневых языков началось с возникновением Фортрана и созданием компилятора для этого языка (1957 г.)

Классы языков программирования



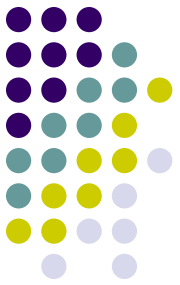
- Функциональные
- Императивные
- Стековые
- Процедурные
- Векторного программирования
- Аспектно-ориентированные
- Декларативные
- Динамические
- Учебные
- Описания интерфейсов
- Прототипные
- Объектно-ориентированные
- Рефлексивные
- Логического программирования
- Параллельного программирования
- Сценарные (скриптовые)
- Эзотерические

Функциональные



В языках функционального программирования основным конструктивным элементом является математическое понятие функции.

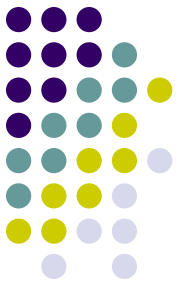
Первым, спроектированным функциональным языком стал Лисп. Вариант данного языка широко используется в системе автоматизированного проектирования AutoCAD и называется AutoLISP



Основные свойства

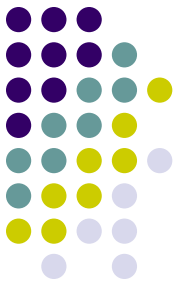
- краткость и простота;
- строгая типизация;
- модульность;
- функции — объекты вычисления;
- чистота (отсутствие побочных эффектов);
- отложенные (ленивые) вычисления.

Языки функционального программирования



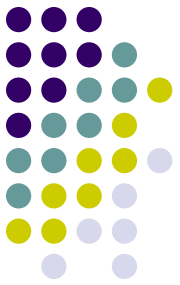
- Лисп
- Haskell
- ML
- Flang
- Erlang
- Prolog (функционально-логический)

Императивные

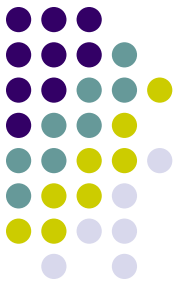


Процедурное (императивное)

программирование является отражением архитектуры традиционных ЭВМ, которая была предложена фон Нейманом в 40-х годах. Теоретической моделью процедурного программирования служит алгоритмическая система под названием «машина Тьюринга».

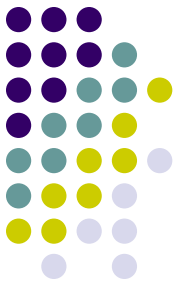


Программа на процедурном языке программирования состоит из последовательности операторов (инструкций), задающих процедуру решения задачи. Основным является оператор присваивания, служащий для изменения содержимого областей памяти. Концепция памяти как хранилища значений, содержимое которого может обновляться операторами программы, является фундаментальной в императивном программировании



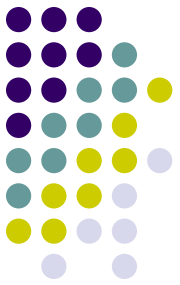
Выполнение программы сводится к последовательному выполнению операторов с целью преобразования исходного состояния памяти, то есть значений исходных данных, в заключительное, то есть в результаты. Таким образом, с точки зрения программиста имеются программа и память, причем первая последовательно обновляет содержимое последней.

Процедурные языки программирования



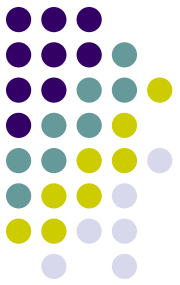
- Ада (язык общего назначения)
- Бейсик (версии начиная с Quick Basic до появления Visual Basic)
- Си
- КОБОЛ
- Фортран
- Модула-2
- Паскаль
- ПЛ/1
- Рапира
- REXX

Стековый язык

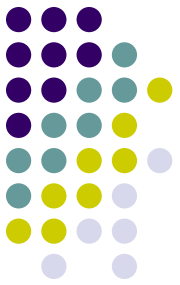


Стековый язык программирования (англ. *stack-oriented programming language*) — это язык программирования, в котором для передачи параметров используется машинная модель стека. Этому описанию соответствует несколько языков, в первую очередь Forth и PostScript, а также многие ассемблерные языки (использующие эту модель на низком уровне — Java, C#). При использовании стека, в качестве основного канала передачи параметров между словами, элементы языка, естественным образом, образуют фразы (последовательное сцепление). Это свойство сближает данные языки с естественными языками.

Аспектно-ориентированное программирование



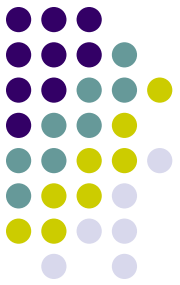
Аспектно-ориентированное программирование (АОП) — парадигма программирования, основанная на идее разделения функциональности, особенно сквозной функциональности, для улучшения разбиения программы на модули.



Примеры

Трассировка — типичный пример сквозной функциональности. Другие примеры: контрактное программирование, в частности проверка пред- и пост-условий, обработка ошибок, реализация систем безопасности.

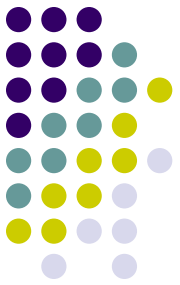
Структурное программирование



Структурное программирование предполагает точно обозначенные управляющие структуры, программные блоки, отсутствие инструкций безусловного перехода (GOTO), автономные подпрограммы, поддержка рекурсии и локальных переменных.

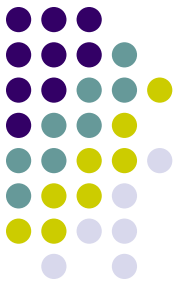
Суть такого подхода заключается в возможности разбиения программы на составляющие элементы.

Языки структурного программирования



- Алгол(1958), Паскаль(1970), Си(1972)

Декларативный язык программирования

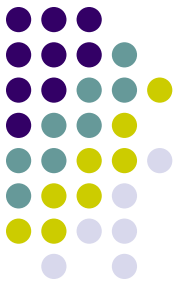


Декларативные языки программирования —

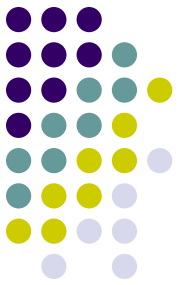
это языки программирования высокого уровня, в которых программистом не задается пошаговый алгоритм решения задачи ("как" решить задачу), а некоторым образом описывается, "что" требуется получить в качестве результата.

Механизм обработки сопоставление по образцу декларативных утверждений уже реализован в устройстве языка. Типичным примером таких языков являются *языки логического программирования* (языки, основанные на системе правил).

Динамические языки программирования

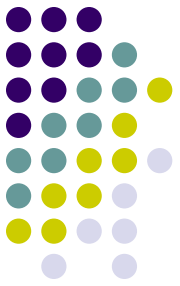


Динамический язык позволяет определять типы данных и осуществлять синтаксический анализ и компиляцию «на лету», непосредственно на этапе выполнения. Динамические языки больше подходят для быстрой разработки приложений.



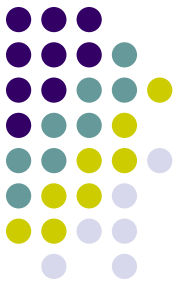
- К динамическим языкам относятся: Python, PHP, Ruby, JavaScript.
- Visual Basic можно рассматривать как некую разновидность динамического языка, ему всегда были присущи определённые динамические черты.

Учебный язык программирования



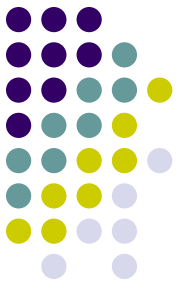
Учебный язык программирования — язык программирования, предназначенный для обучения специалистов программированию. Такой язык должен отвечать главному требованию: простота.

Примеры



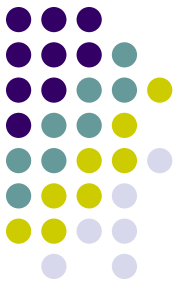
- Алгоритмический язык
- Бейсик
- Е-практикум
- КуМир
- Лого
- Робик
- Рапира
- Форт

Язык описания интерфейсов



- **IDL**, или **язык описания интерфейсов** (англ. *Interface Description Language* или *Interface Definition Language*) — язык спецификаций для описания интерфейсов, синтаксически похожий на C++.
- **CORBA IDL** — Разработан OMG для описания интерфейсов распределенных объектов — названий методов и типов переменных-аргументов. Создан в рамках обобщенной архитектуры CORBA.
- **COM IDL** — Аналогичная CORBA IDL разработка Microsoft, созданная для описания интерфейсов между модулями COM. В общем случае может рассматриваться как подмножество CORBA IDL.

Объектно-ориентированный язык программирования



В основе концепции объектно-ориентированного программирования лежит понятие **объекта** - некоей субстанции, которая объединяет в себе поля (данные) и методы (выполняемые объектом действия).

Особенности



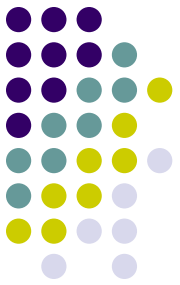
- Наследование. Создание нового класса объектов путем добавления новых элементов (методов). В данный момент ОО языки позволяют выполнять множественное наследование, т. е. объединять в одном классе возможности нескольких других классов.
- Инкапсуляция. Соккрытие деталей реализации, которое (при грамотном использовании) позволяет вносить изменения в части программы безболезненно для других её частей, что существенно упрощает сопровождение и модификацию ПО.
- Полиморфизм. При полиморфизме некоторые части (методы) родительского класса заменяются новыми, реализующими специфические для данного потомка действия. Таким образом, интерфейс классов остаётся прежним, а реализация методов с одинаковым названием и набором параметров различается. С понятием «Полиморфизм» тесно связано понятие «Позднего связывания».
- Типизация. Позволяет устранить многие ошибки на момент компиляции, операции проводятся только над объектами подходящего типа.

Список языков

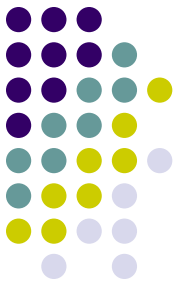


- Eiffel
- Simula
- Java
- C#
- C++
- Objective-C
- Object Pascal (Delphi)
- VB.NET
- Visual DataFlex
- Perl
- PHP
- Python
- Scala
- ActionScript
- JavaScript
- JScript.NET
- Ruby
- Smalltalk
- Ada
- Xbase++
- X++
- Vala

Прототипное программирование



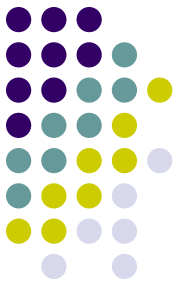
Прототипное программирование — стиль объектно-ориентированного программирования, при котором отсутствует понятие класса, а повторное использование (наследование) производится путём клонирования существующего экземпляра объекта — **прототипа**.



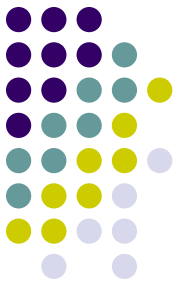
Примеры:

- Self
- JavaScript
- Cecil
- NewtonScript
- Io
- Slate
- MOO
- REBOL
- Kevo

Логическое программирование



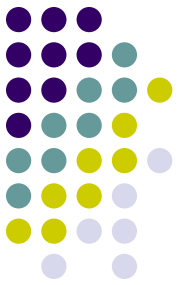
Логическое программирование — парадигма программирования, а также раздел дискретной математики изучающий методы и возможности этой парадигмы, основанная на выводе новых фактов из данных фактов согласно заданным логическим правилам. Логическое программирование основано на теории математической логики. Самым известным языком логического программирования является Prolog



Примеры:

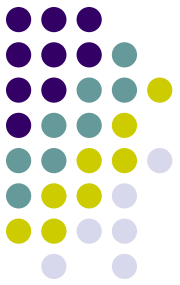
- Prolog
 - Mercury, Visual Prolog, Oz и Fril
- Planner
 - QA-4, Popler, Conniver, и QLISP

Язык параллельного программирования



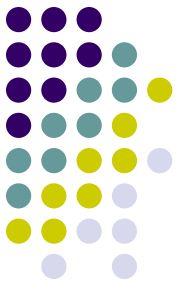
МС# (читается как «эм си шарп», mcsharp),
Multiprocessor C#)

Скриптовый язык

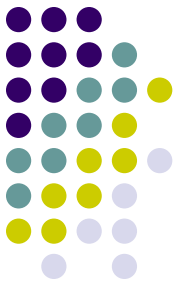


Скриптовый язык (англ. *scripting language*, в русскоязычной литературе принято название **язык сценариев**) — язык программирования, разработанный для записи «сценариев», последовательностей операций, которые пользователь может выполнять на компьютере. Простые скриптовые языки раньше часто называли *языками пакетной обработки* (*batch languages* или *job control languages*). Сценарии обычно интерпретируются, а не компилируются (хотя часто сценарии компилируются каждый раз перед запуском).

Эзотерические языки программирования



- **Эзотерические языки программирования** — вид языков программирования, не предназначенных для практического применения. Образец компьютерного юмора.
- Эзотерические языки придумываются для развлечения, часто они пародируют «настоящие» или являются абсурдным воплощением «серьёзных» концепций программирования.



Языки программирования

- Системное программирование
- Прикладное программирование – приложения.