CS350 PROGRAMMING LANGUAGE DESIGN

Ch1 : Preliminaries

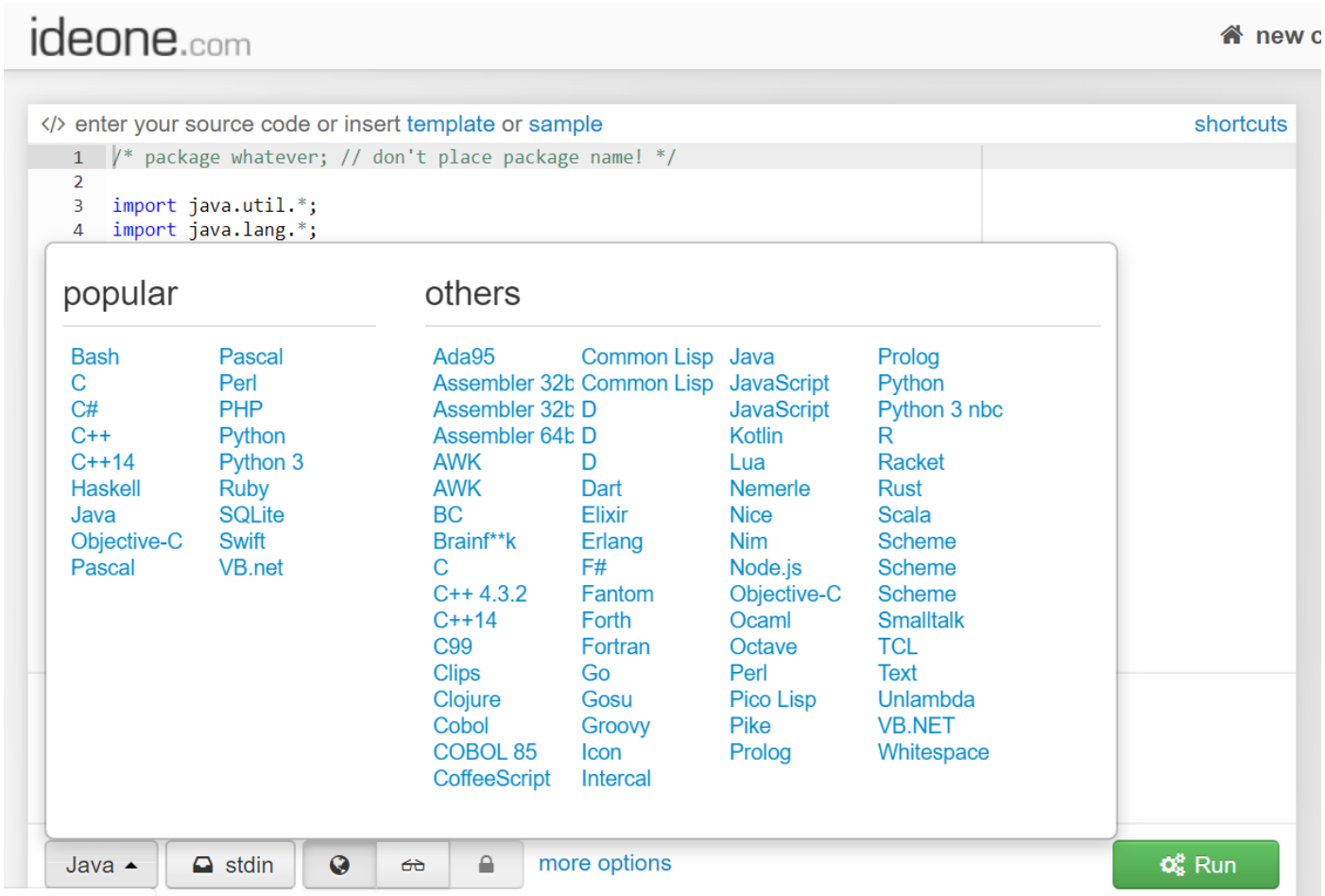
COURSE GOALS

- Survey of language design issues and their implications for translation and run-time support.
- Overview of modern programming languages and their features including abstract data and control structures, binding and scope rules, subprograms, parameter passing mechanisms, Exception Handling, as well as support for concurrency.
- Describe different paradigms of programming languages such as: Object-oriented, functional, and Logic programming languages.

SOFTWARE

- C, C++, Fortran, and Ada gcc.gnu.org
- C# and F# microsoft.com Java java.sun.com
- Scheme www.plt-scheme.org/software/drscheme
- Python www.python.org
- Ruby www.ruby-lang.org
- JavaScript/PHP is included in virtually all browsers;

ALL IN ONE!


ideone.com
new c
</> enter your source code or insert template or sample
shortcuts
1 /* package whatever; // don't place package name! */
2
3 import java.util.*;
4 import java.lang.*;
popular
Bash
C
C#
C++
C++14
Haskell
Java
Objective-C
Pascal
Pascal
Perl
PHP
Python
Python 3
Ruby
SQLite
Swift
VB.net
others
Ada95
Assembler 32b
Assembler 32b
Assembler 64b
AWK
AWK
BC
Brainf**k
C
C++ 4.3.2
C++14
C99
Clips
Clojure
Cobol
COBOL 85
CoffeeScript
Common Lisp
Common Lisp
D
D
D
Dart
Elixir
Erlang
F#
Fantom
Forth
Fortran
Go
Gosu
Groovy
Icon
Intercal
Java
JavaScript
JavaScript
Kotlin
Lua
Nemerle
Nice
Nim
Node.js
Objective-C
Ocaml
Octave
Perl
Pico Lisp
Pike
Prolog
Prolog
Python
Python 3 nbc
R
Racket
Rust
Scala
Scheme
Scheme
Scheme
Smalltalk
TCL
Text
Unlambda
VB.NET
Whitespace
Java
stdin
more options
Run

TOPICS

1.1 Reasons for Studying Concepts of Programming Languages

1.2 Programming Domains

1.3 Language Evaluation Criteria

1.4 Influences on Language Design

1.5 Language Categories

1.6 Language Design Trade-Offs

1.7 language implementation methods

WHY CONCEPTS OF PROGRAMMING LANGUAGES??

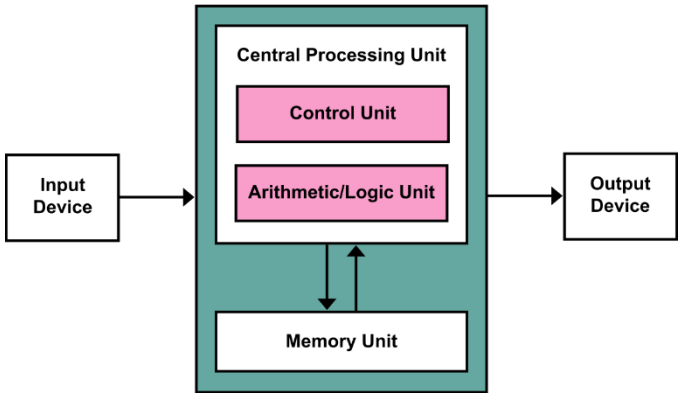
- Better use of languages that are already known
  - New features and unknown constructs
  - Increased ability to express ideas
- Increased ability to learn new languages
  - See how concepts are incorporated into the design of a language
- Improved background for choosing appropriate languages
  - Choose based on features rather than familiarity
- Better understanding of significance of implementation
  - Understanding design issues leads to intelligent use

JUST WONDERING!

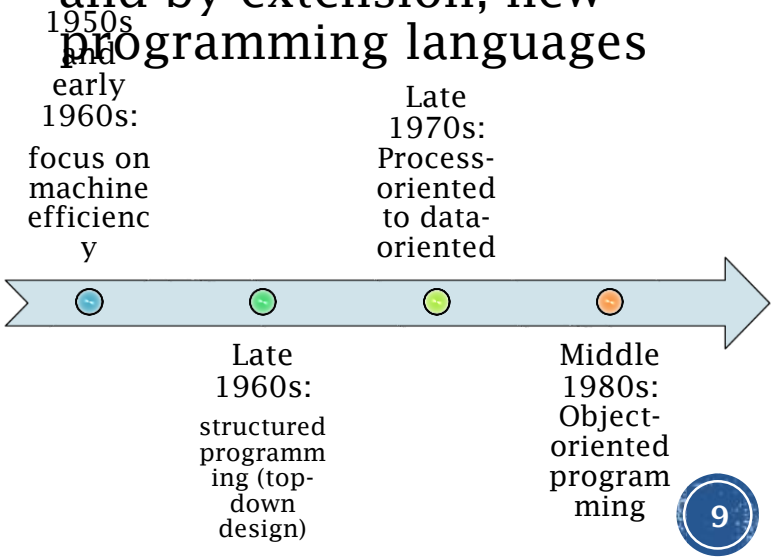
- How Many Computer Programming Languages Are There?
  - According to Wikipedia, there are about 700 programming languages
  - Other sources that only list notable languages still count up to an impressive 245 languages.
  - Another list called HOPL, that claims to include every programming language to ever exist, puts the total number of programming languages at 8,945.

INFLUENCES ON LANGUAGE DESIGN

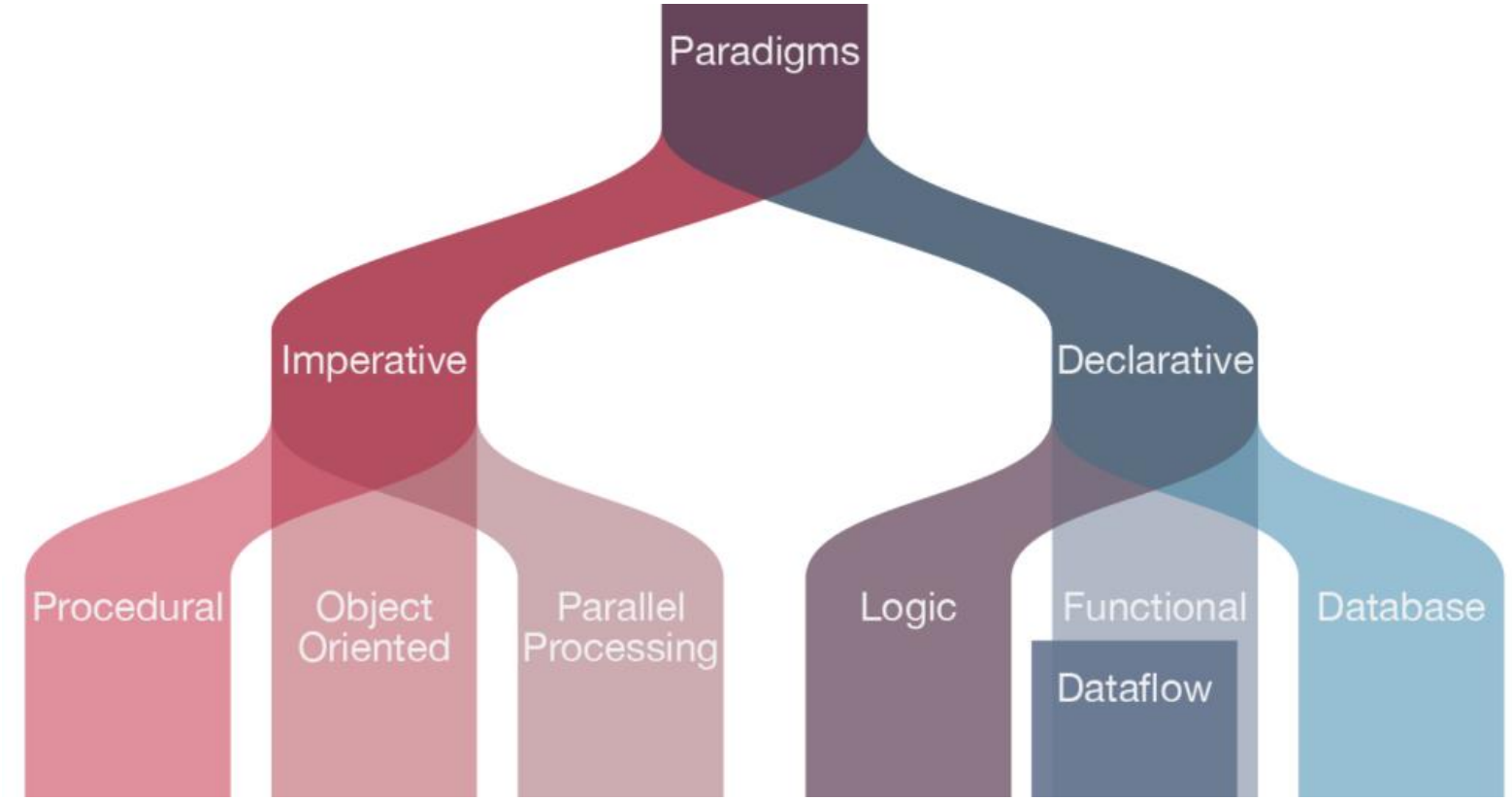
- *Computer Architecture*
  - Languages are developed around the prevalent computer architecture, known as the von Neumann architecture



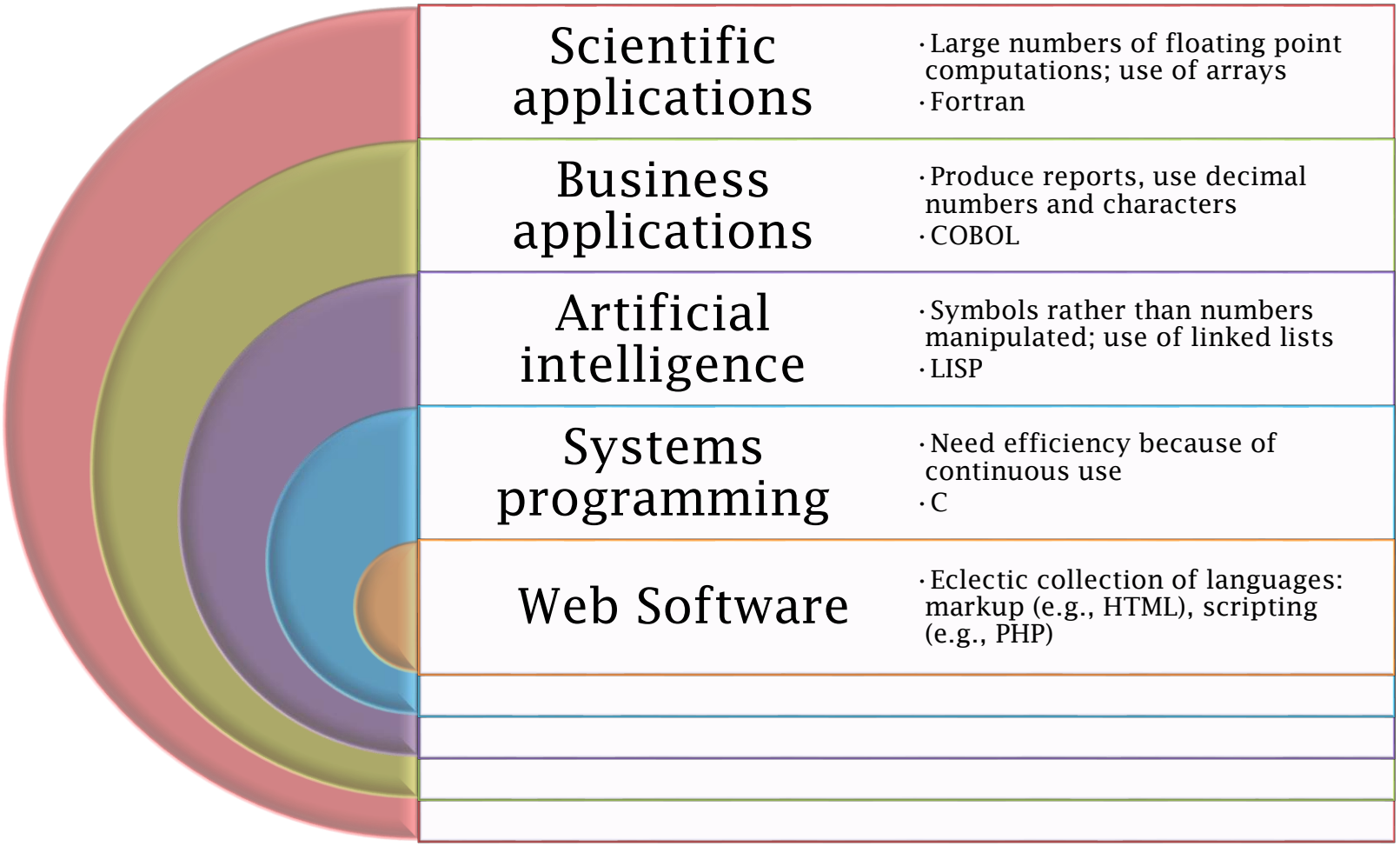
- *Program Design Methodologies*
  - New software development methodologies (e.g., object-oriented software development) led to new programming paradigms and by extension, new programming languages



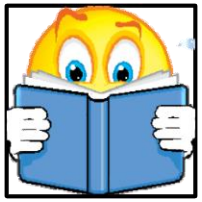
LANGUAGE CATEGORIES



PROGRAMMING DOMAINS



LANGUAGE EVALUATION CRITERIA



Readability — the ease with which programs can be read and understood

the ease with which a language can be used to create programs — **Writability**

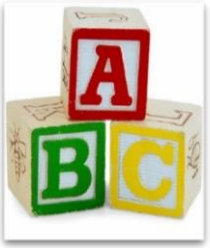
Reliability — conformance to specifications

the ultimate total cost — **Cost**

READABILITY

- Ease of maintenance is determined in large part by the readability of programs
- Characteristics that affect readability:
  - Syntax design
    - meaningful keywords to indicate its purpose
    - Special words and methods of forming compound statements (endif in Ada)
  - Simplicity
    - A manageable set of features and constructs
    - Minimal feature multiplicity
  - Orthogonality
    - A relatively small set of primitive constructs can be combined in a relatively small number of ways where every possible combination is legal → less exceptions

WRITABILITY



- Writability must be considered in the context of the target problem domain of a language
  - VBasic vs. C for GUI application
- Characteristics that affect writability:
  - *Simplicity and orthogonality*
    - Few constructs, a small number of primitives, a small set of rules for combining them
  - *Expressivity*
    - A set of relatively convenient ways of specifying operations
    - Eg. Using for loops simplified counting loops

RELIABILITY

- A program is said to be reliable if it performs to its specification under all conditions
- Related characteristics:
  - Type checking
    - Testing for type errors (eg. Function parameters)
  - Exception handling
    - Intercept run-time errors and take corrective measures
  - Aliasing:
    - Different names to the same memory cell.

COST

- Training programmers to use the language
  - Function of simplicity and orthognality
- Writing programs
  - closeness to particular applications
- Reliability: poor reliability leads to high costs
  - Critical apps → very high
  - Non-critical → lost future business or lawsuits
- Maintaining programs:
  - Usually done by different programmers → readability is an issue!
    - large software systems with relatively long lifetimes, maintenance costs can be as high as two to four times as much as development costs

EVALUATION CRITERIA: OTHER

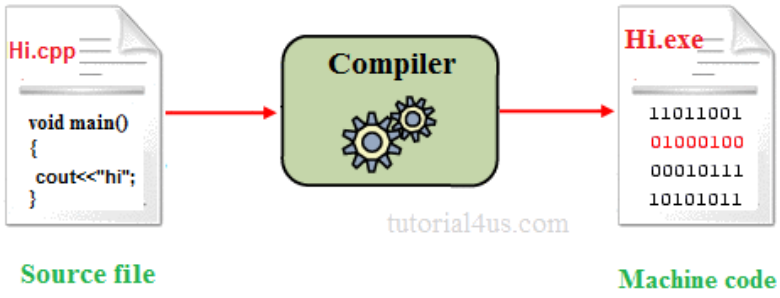
- Portability
  - The ease with which programs can be moved from one implementation to another
- Generality
  - The applicability to a wide range of applications
- Well-defineness
  - The completeness and precision of the language's official definition

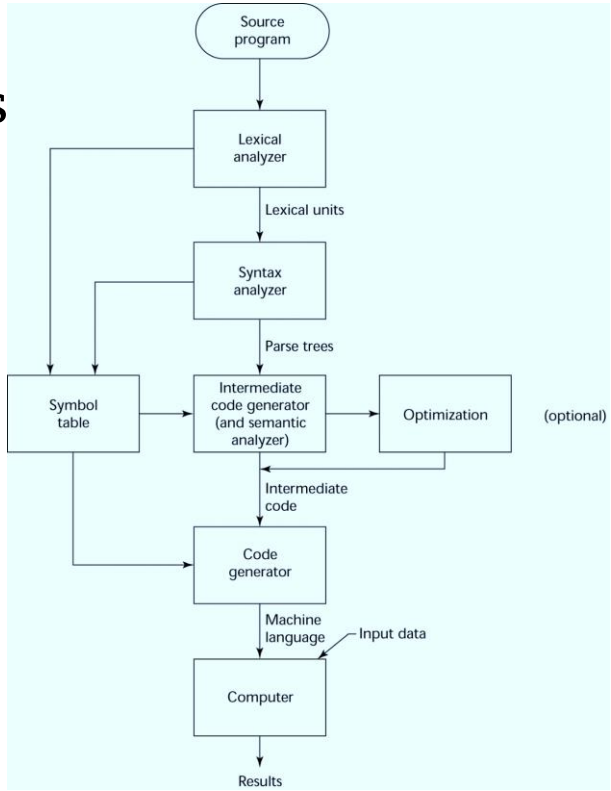
LANGUAGE DESIGN TRADE-OFFS

- Readability vs. writability
  - Example: APL provides many powerful operators (and a large number of new symbols), allowing complex computations to be written in a compact program but at the cost of poor readability
- Reliability vs. cost of execution
  - Example: Java demands all references to array elements be checked for proper indexing, which leads to increased execution cost
- Writability (flexibility) vs. reliability
  - Example: C++ pointers are powerful and very flexible but are unreliable
    - The easier a program to write, the more likely it is to be correct!

IMPLEMENTATION METHODS

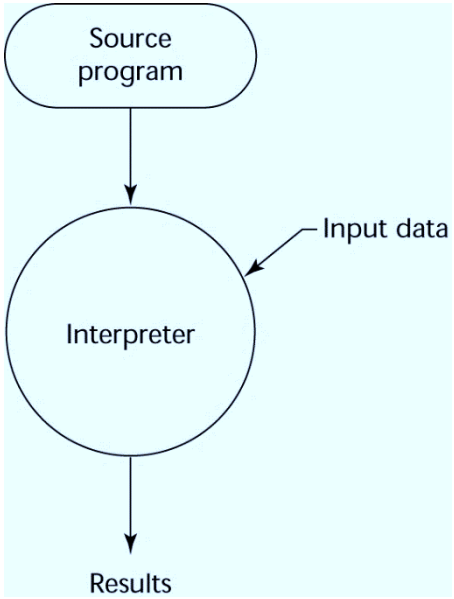
- Translate high-level program (source language) into machine code (machine language)
- Slow translation, fast execution
- Compilation process has several phas





IMPLEMENTATION METHODS

- Programs are interpreted by another program known as an interpreter
- No translation
  - Interpreter is a virtual machine with fetch-decode-execute cycle
  - produces a result from a program statement
- Easier implementation of programs (run-time errors can easily and immediately be displayed)
- Slower execution (10 to 100 times)
  - Due to statement decoding
- Now rare for traditional high-level languages
  - Significant comeback with some Web scripting languages (e.g., JavaScript, PHP)



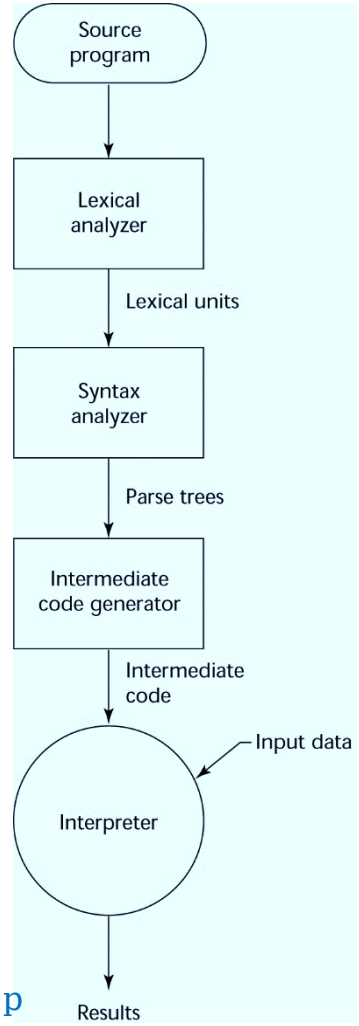
COMPARISON

Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.
Interpreters usually take less amount of time to analyze the source code. However, the overall execution time is comparatively slower than compilers.	Compilers usually take a large amount of time to analyze the source code. However, the overall execution time is comparatively faster than interpreters.
No Object Code is generated, hence are memory efficient.	Generates Object Code which further requires linking, hence requires more memory.
Programming languages like JavaScript, Python, Ruby use interpreters.	Programming languages like C, C++, Java use compilers.

https://www.programiz.com/article/difference-compiler-interpreter

IMPLEMENTATION METHODS

- Hybrid Implementation Systems
  - A compromise between compilers and pure interpreters
- A high-level language program is translated to an intermediate language that allows easy interpretation
- Faster than pure interpretation
  - Use: Small and medium systems when efficiency is not the first concern



https://www.tutorialspoint.com/execute_ruby_online.php

SUMMARY

- The study of programming languages is valuable for a number of reasons:
  - Increase our capacity to use different constructs
  - Enable us to choose languages more intelligently
  - Makes learning new languages easier
- Most important criteria for evaluating programming languages include:
  - Readability, writability, reliability, cost
- Major influences on language design have been machine architecture and software development methodologies