Programming Languages
and Program Development

I want pizza!

Check for money.

Order pizza.

Have money

Pay for pizza.

No money

No pizza

Pizza algorithm done

Box leftovers and put in fridge.

Full

If still hungry

Still hungry

Eat a piece of pizza.
Programming Languages and How They Work

• **Programming**
  - Process used to create software programs

• **Programmers**
  - People who use programming languages to create software applications

• **Programming languages**
  - Consist of a vocabulary and a set of rules called *syntax*
Programming Languages and How They Work

• **Interface**
  
  - Point of interaction between components, such as the interaction between a user’s screen and the computer code, which results when running a program
Development of Programming Languages
Development of Programming Languages

• **Compilers and interpreters**
  o **Source code**—programming instructions in the original form needed to translate a form that the computer can understand
  o **Code**—programming instructions created by the programmers
  o **High-level language**—language that mimics English; does not require a programmer to understand the intimate details of how hardware, especially the processor, handles data

```html
<!DOCTYPE html>
<html id="home-layout">
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8">
  <title>Source Code Pro</title>
  <!-- made with <3 and AFDK0 -->
  <meta name="keywords" content="sans, monospace, open source, coding, for">
  <link rel="stylesheet" type="text/css" src="style.css">
</head>
<body>
  <div id="main">
```
Development of Programming Languages

Compiler implementation:
- Source code → Compiler → Object code
  - Locate errors. Correct source code.
  - Executable program → Program output

Interpreter implementation:
- Source code → Interpreter → Program output
  - Data
Development of Programming Languages

- **Compilers and interpreters**
  - **Compiler**—utility program translates source code into **object code**
  - **Executable program**—code transformed from object code ready to run programs that do not need to be altered
  - **Interpreter**—translation program that does not produce object code—translates one line of source code at a time; executes the translated instruction
Development of Programming Languages

- Five generations of programming languages
  - Machine
  - Assembly
  - Procedural
  - Nonprocedural
  - Natural

```
i = 1; j = 1;
while (true){
  *val++ = i+j;
  j = i+(i=j);
}
```

- Assembly Code
- Binary Code
- High Level Code
Development of Programming Languages

• First-generation languages
  o Machine language
    • Based on binary numbers
    • Only programming language that a computer understands directly
    • Machine dependent
Development of Programming Languages

- **Second-generation language**
  - **Assembly language**
    - **Low-level language**
    - Programs use:
      - Mnemonics—brief abbreviations for program instructions make assembly language easier to use
      - Base-10 (decimal) numbers
    - Must be translated into machine language by an **assembler**
    - Occasionally used to create **device drivers**
      - Programs to control devices attached to a computer and game console programs
Development of Programming Languages

- **Third-generation languages**
  - **High-level languages**—do not require programmers to know details relating to the processing of data
  - Easier to read, write, and maintain than assembly and machine languages
  - Source code must be translated by a language translator
Development of Programming Languages

- **Third-generation languages (con’t.)**
  - *Spaghetti code*—difficult to follow, messy in design, prone to errors due to numerous GOTO statements
  - *Structured programming*—set of quality standards; programs more verbose but more readable, reliable, and maintainable
    - GOTO statements forbidden
    - Examples:
      - Ada
      - Pascal
Development of Programming Languages

• Third-generation languages (con’t.)
  o **Modular programming**—dividing larger programs into separate modules, each takes care of a specific function
  o **Information hiding**—also known as encapsulation, modular programming makes it possible to hide details in sensitive applications
  o Programming languages include:
    • Fortran
    • C
Development of Programming Languages

- **Procedural languages**
  - Provide detailed instructions that are designed to carry out a specific action such as printing a formatted report

- **Nonprocedural languages**
  - Do not require programmers to use step-by-step instructions
Development of Programming Languages

• **Fourth-generation languages**
  o **Nonprocedural languages**
  o Do not require step-by-step procedures
  o Examples
    • **Report generators** (database reports)
    • **Query languages**
      o **SQL (Structured query language)**—enables users to phrase simple or complex requests for data
Development of Programming Languages

- **Fifth-generation languages**
  - Natural language
  - Still being perfected
  - Nonprocedural
  - Use everyday language to program
Development of Programming Languages

- **Object-oriented programming (OOP)**
  - Programming technique based on data being conceptualized as objects
    - **Object**—unit of computer information that defines a data element that is used to model real-world objects
    - **Attributes** define the data
    - Procedures or operations are called **methods**
Development of Programming Languages

- **Object-oriented programming (OOP) (con’t.)**
  - *Class*—blueprint or prototype from which objects are made
  - *Inheritance*—ability to pass on characteristics from a class to subclasses
Development of Programming Languages

- **Program development methods**
  - **Rapid application development (RAD)**
    - Reuses prebuilt objects
    - Possible because of OOP
  - **Joint application development (JAD)**
    - Uses a team approach
    - Involves end users throughout development
  - **Agile software development** techniques—use collaboration between teams to develop solutions to meet customer needs and company goals
Development of Programming Languages

- **Middleware**
  - Makes connections between applications on multiple networks
  - Integral to modern information technology based on XML, Web services, and service-oriented architecture
A Guide to Programming Languages:
One Size Doesn’t Fit All

• **Early high-level languages**
  - **COBOL (Common Business-Oriented Language)**
    - Used for business applications
    - Current focus is on editing aged code on mainframe computers
  - **Fortran (formula translator)**
    - Used for scientific/math/engineering applications
    - Being replaced by object-oriented and formula-solving programs
  - **Mathematica**
    - Used to handle all aspects of technical computing in a coherent and unified way
    - Can manipulate a wide range of objects, using only a small number of basic elements
A Guide to Programming Languages: One Size Doesn’t Fit All

• **Structured and modular languages**
  - Required for large-scale program development
  - Languages in widespread use:
    - **Ada**
    - **BASIC**
    - **Visual Basic**
      - Event-driven programming language that executes only in response to user actions
    - **C**
A Guide to Programming Languages: One Size Doesn’t Fit All

• **Object-oriented languages**
  - **Visual Studio.NET**—Microsoft’s answer to Java and JavaScript
    - Visual Basic.Net
    - Visual C++
    - Visual C#
    - F#—language that combines object-oriented features with the assets of a functional language
      - **Functional language** reflects the way people think mathematically—useful in programs that express findings in mathematical form
  - **Integrated development environment (IDE)**