Technology in Action

Alan Evans • Kendall Martin

Mary Anne Poatsy

Ninth Edition

Technology in Action

Chapter 10

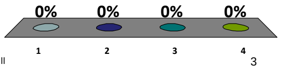
Behind the Scenes: Software Programming

Chapter Topics

- Understanding software programming
- Life cycle of an information system
- Life cycle of a program
- Programming languages

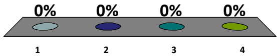
Information Systems include all *EXCEPT* the following:

1. People
2. Data
3. Procedures
4. Locations



Variable declaration tells the operating system that the program needs to allocate storage space in the:

1. Hard drive
2. ROM
3. RAM
4. CPU

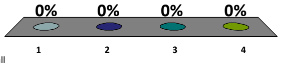


Scope creep occurs when:

1. The program requires hundreds of additional lines of code due to programming errors.
2. Clients request additional functionality and features which extends the project completion time.

Programming languages are classified in several major groupings referred to as:

1. Generations
2. Classes
3. Objects
4. Designs



Understanding Software Programming

- Types of tasks that are candidates for automation:
  - – Routine
  - – Repetitive
  - – Work with electronic data
  - – Follow a series of clear steps
- When existing software cannot be found, programming is mandatory

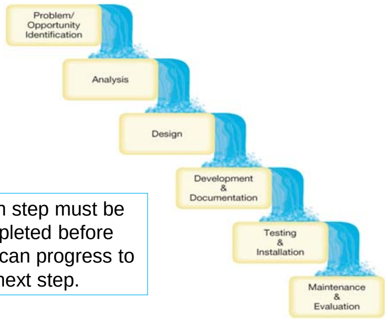
Life Cycle of an Information System

- System
  - – A collection of pieces working together to achieve a common goal
- An information system includes:
  - – Data
  - – People
  - – Procedures
  - – Hardware
  - – Software

System Development Life Cycle

- To create modern software, an entire team is needed
- Programs require many phases to complete
- Must be available for multiple operating systems and work over networks
- Must be free of errors and well supported

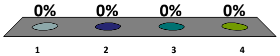
Six Steps in the SDLC


Problem/ Opportunity Identification
Analysis
Design
Development & Documentation
Testing & Installation
Maintenance & Evaluation

Each step must be completed before you can progress to the next step.

Which of the following is *NOT* a phase in the SDLC?

1. Design
2. Analysis
3. Development and Documentation
4. Revisions



Problem & Opportunity Identification

- Development steering committee formed to evaluate systems development proposals
- Reviews ideas
- Decides which projects to take forward based on available resources
  - Personnel and funding

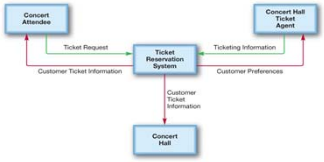
Analysis

- Analysts explore problem to be solved
- Develop program specifications
  - Clear statement of goals and objectives of project
- Feasibility assessment is performed
- User requirements are defined
- Analysts recommend a plan of action

Design

- A detailed plan for programmers is developed
- Flowcharts and data-flow diagrams are used for the current and proposed system

Data-flow diagram


Concert Attendee
Concert Hall Ticket Agent
Ticket Request
Ticketing Information
Ticket Reservation System
Customer Ticket Information
Customer Preferences
Customer Ticket Information
Concert Hall

Development and Documentation

- Actual programming takes place
- First phase of the program development life cycle (PDLC)

Testing and Installation

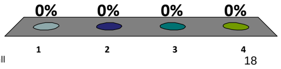
- Program is tested to ensure it works properly
- Program is installed for use

Maintenance and Evaluation

- Performance of the system is monitored
- Corrections and modifications to the program are made
- Additional enhancements that users request are evaluated
- Appropriate program modifications are made

Flowcharts and Data-flow diagrams are created during which stage of the SDLC?

1. Problem Identification
2. Analysis
3. Design
4. Testing



Joint Application Development

- JAD helps designers adapt to changes in program specifications
- Includes customer involvement
- No communication delays
- Also referred to as:
  - Accelerated design
  - Facilitated team technique

Ethics in IT

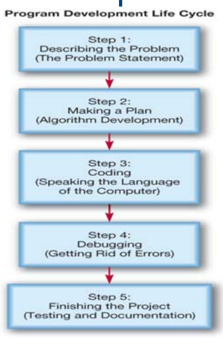
The Association of Computing Machinery (ACM) and the Institute of Electrical and Electronic Engineers (IEEE) have established eight principles for ethical software engineering practices:

1. Public
2. Client and Employer
3. Product
4. Judgment
5. Management
6. Profession
7. Colleagues
8. Self

The Life Cycle of a Program

- Programming is the process of translating a task into a series of commands a computer will use to perform that task
- Programming involves:
  - Identifying the parts of a task the computer can perform
  - Describing tasks in a highly specific and complete manner
  - Translating this description into a language understood by the computer’s CPU

Program Development Life Cycle


Program Development Life Cycle

Step 1: Describing the Problem (The Problem Statement)

Step 2: Making a Plan (Algorithm Development)

Step 3: Coding (Speaking the Language of the Computer)

Step 4: Debugging (Getting Rid of Errors)

Step 5: Finishing the Project (Testing and Documentation)

Step 1: Describing the Problem

- Programmers develop a complete description of problem
- Problem statement identifies task to be automated
- Statement describes how software will behave

Step 2: Making a Plan

- Problem statement is translated into a set of specific, sequential steps known as an algorithm
- Algorithm is written in natural ordinary language such as English

Step 3: Coding

- Algorithm is translated into programming code
- Programmers must think in terms of operations that a CPU can perform

Step 4: Debugging

- Code goes through process of debugging
- Programmers repair any errors found in code

Step 5: Finishing the Project

- Software is tested
  - Programming team
  - People who will use program
- Results of entire project are documented
- Users are trained to use program efficiently

Describing the Problem

- The Problem Statement
  - Staring point of programming work
  - Clear description of tasks the computer program must accomplish
  - How the program will execute these tasks
  - How the program will respond to unusual situations

Creating Problem Statements

- Programmers interact with users to describe three relevant things:
  1. Data – raw input users have at the start
  2. Information – result users require
  3. Method – process of how program converts the inputs to correct outputs

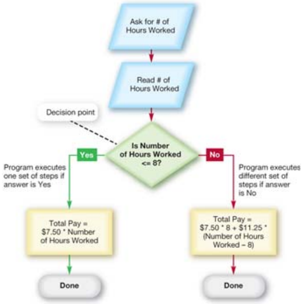
Parking Garage Example

Program Goal:	To compute the total pay for a fixed number of hours worked at a parking garage.		
Inputs:	Number of Hours Worked........................ a positive number		
Outputs:	Total Pay Earned .... a positive number		
Process:	The Total Pay Earned is computed as $7.50 per hour for the first eight hours worked each day. Any hours worked beyond the first eight are billed at $11.25 per hour.		
Error Handling:	The input (Number of Hours Worked) must be a positive real number. If it is a negative number or other non-acceptable character, the program will force the user to re-enter the information.		
Testing Plan:	**INPUT**	**OUTPUT**	**NOTES**
	8	8*7.50	Testing positive input
	3	3*7.50	Testing positive input
	12	8*7.50 + 4*11.25	Testing overtime input
	–6	Error message/ask user to re-enter value	Handling error

Making a Plan

- Algorithm Development
  - Set of specific sequential steps
  - Describe exactly what computer program must do to complete task
  - Use natural language

Flowchart


Ask for # of Hours Worked
Read # of Hours Worked
Decision point
Is Number of Hours Worked <= 8?
Yes
No
Program executes one set of steps if answer is Yes
Program executes different set of steps if answer is No
Total Pay = $7.50 * Number of Hours Worked
Total Pay = $7.50 * 8 + $11.25 * (Number of Hours Worked – 8)
Done
Done

Pseudocode

Bold terms show actions that are common in programming, such as reading data, making decisions, printing, and so on.

```
1. Ask the user how many hours they worked
today
2. If the number of hours worked < = 8,
   compute total pay without overtime
   otherwise,
   compute total pay with overtime pay
3. Print total pay
```

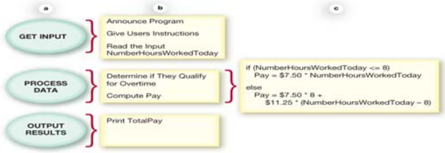
Underlined words are information items that appear repeatedly in the algorithm.

Developing the Algorithm

- Decision Making and Design
  - Convert problem statement into list of steps or actions
  - Only simplest algorithms execute same series of actions every time they run
  - Complex problems involve choices and include decision points

Top-Down Design

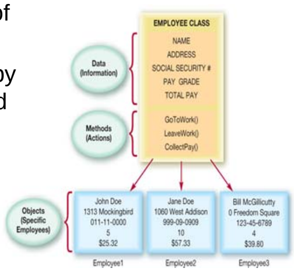
- Problem is divided into a series of high-level tasks
- Detailed subtasks are created from high-level tasks



```
if (NumberHoursWorkedToday <= 8)
   Pay = $7.50 * NumberHoursWorkedToday
else
   Pay = $7.50 * 8 +
      $11.25 * (NumberHoursWorkedToday – 8)
```

Object-Oriented Analysis

- Classes (categories of inputs) are identified
- Classes are defined by information (data) and actions (methods or behaviors)
- Reusability is key



Writing Program Code

- Programmers select best programming language for the problem
- Translate the algorithm into that language
- Translation is act of coding

Generations of Programming Languages

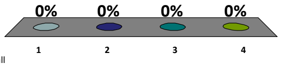
Level	Generation	Example
Low	1GL	Machine
	2GL	Assembly
High	3GL	FORTRAN, BASIC, C, Java
	4GL	SQL
Natural	5GL	PROLOG

Coding

- Speaking the Language of the Computer
  - Syntax
    - Agreed-upon set of rules of language used
  - Keywords
    - Set of words with predefined meanings
  - Data types
    - Describe the kind of data being stored in memory
  - Operators
    - Coding symbols that represent fundamental actions

Which of the following translates source code, line by line?

1. Compliers
2. Interpreters
3. Translators
4. Both 2 & 3



Compilation

- Compilation is the process of converting code into machine language
- A compiler reads the source code and translates it into machine language
- After compilation, programmers have an executable program

Interpreter

- Some programming languages do not have a compiler, but use an interpreter instead
  - The interpreter translates source code into a line-by-line intermediate form
  - Each line is executed before the next line is compiled
  - Programmers do not have to wait for the entire program to be recompiled each time they make a change
  - Programmers can immediately see the results of changes as they are making them

Coding Tools

- Integrated Development Environment
  - Developmental tool that helps programmers write, compile, and test programs
- Every language has its own specific IDE

Debugging

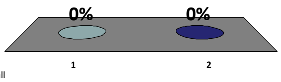
- Getting Rid of Errors
  - Process of running program over and over
    - To find errors
    - To make sure the program behaves the way it should

Finishing the Project

- Testing and Documentation
  - Internal testing – a group with the software company uses program in every way possible
  - External testing – people like those who will eventually purchase the program work with it

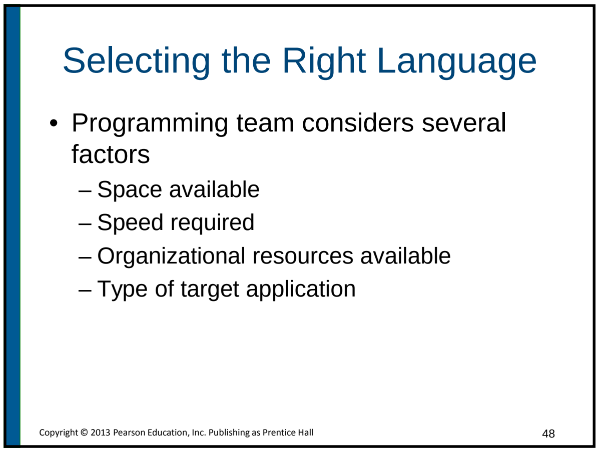
I have used a beta version of a software program.

1. Yes
2. No



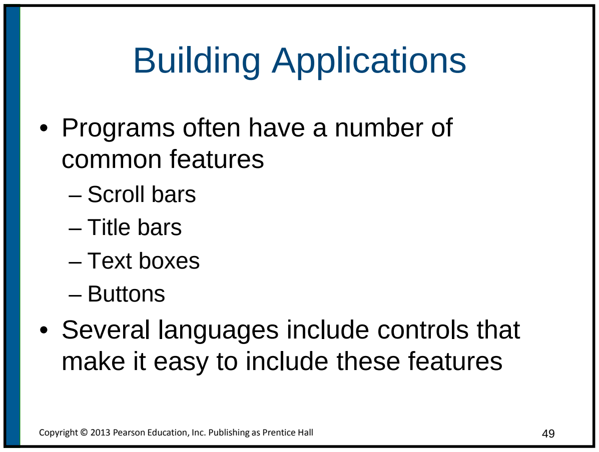
Programming Languages

- Many languages for many projects
- Create a solution to meet several competing objectives
  - Software must run quickly and reliably
  - Simple to expand when demands change
  - Completed on time for minimal cost
  - Use smallest amount of system resources



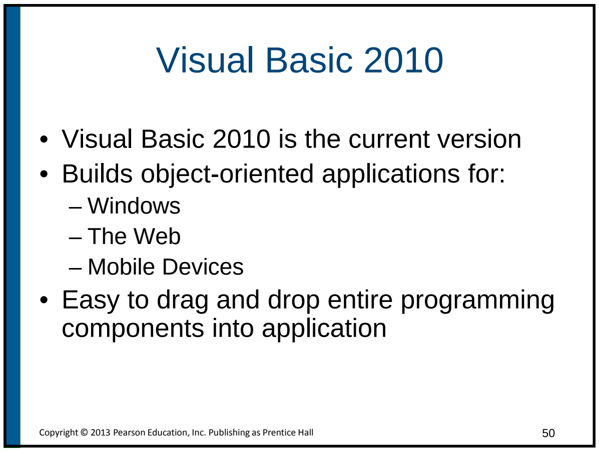
Selecting the Right Language

- Programming team considers several factors
  - Space available
  - Speed required
  - Organizational resources available
  - Type of target application



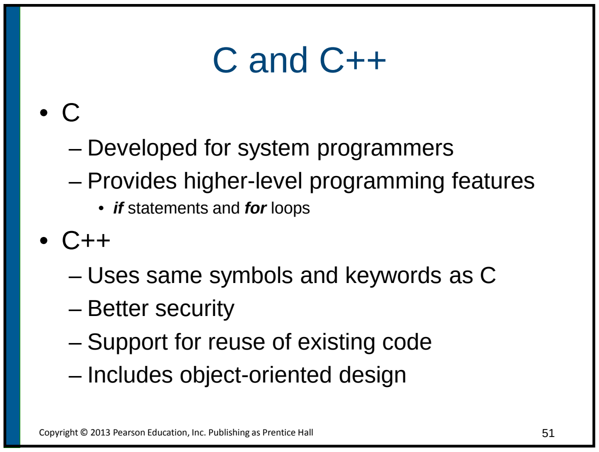
Building Applications

- Programs often have a number of common features
  - Scroll bars
  - Title bars
  - Text boxes
  - Buttons
- Several languages include controls that make it easy to include these features



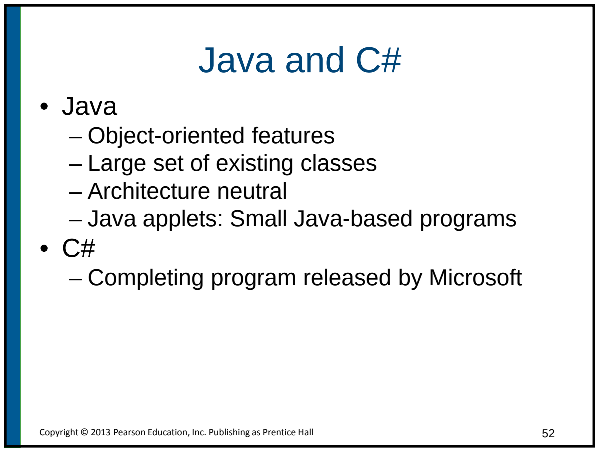
Visual Basic 2010

- Visual Basic 2010 is the current version
- Builds object-oriented applications for:
  - Windows
  - The Web
  - Mobile Devices
- Easy to drag and drop entire programming components into application



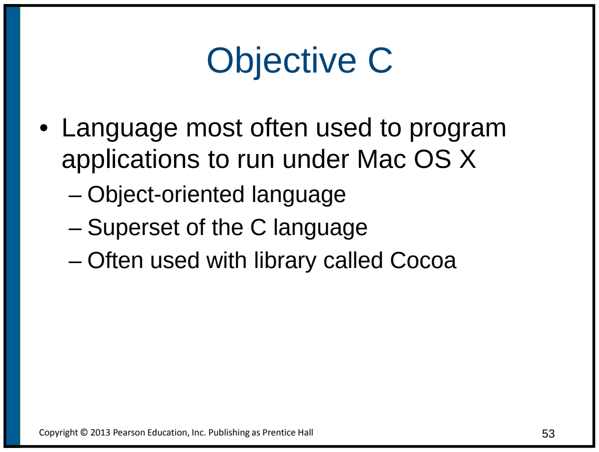
C and C++

- C
  - Developed for system programmers
  - Provides higher-level programming features
    - ***if*** statements and ***for*** loops
- C++
  - Uses same symbols and keywords as C
  - Better security
  - Support for reuse of existing code
  - Includes object-oriented design



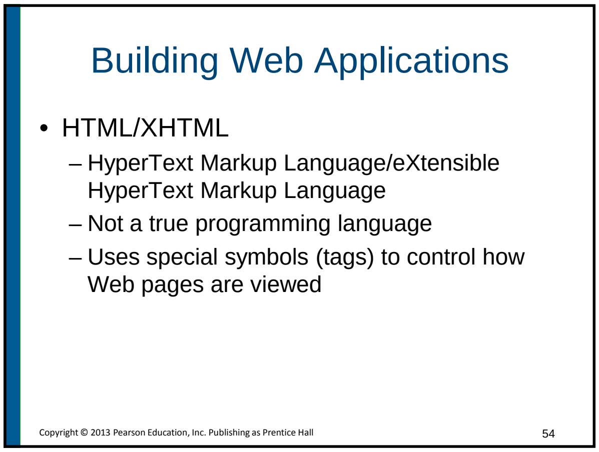
Java and C#

- Java
  - Object-oriented features
  - Large set of existing classes
  - Architecture neutral
  - Java applets: Small Java-based programs
- C#
  - Completing program released by Microsoft



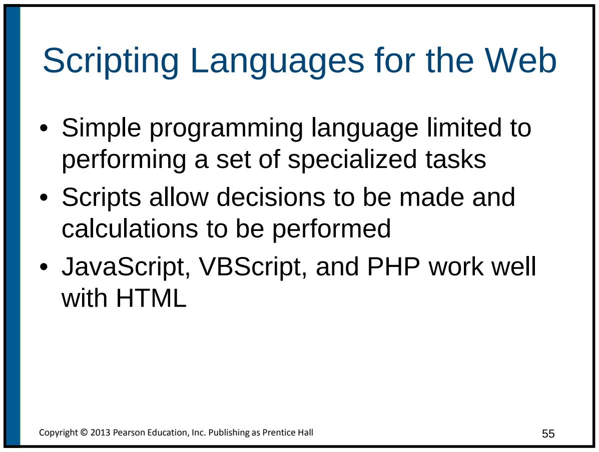
Objective C

- Language most often used to program applications to run under Mac OS X
  - Object-oriented language
  - Superset of the C language
  - Often used with library called Cocoa



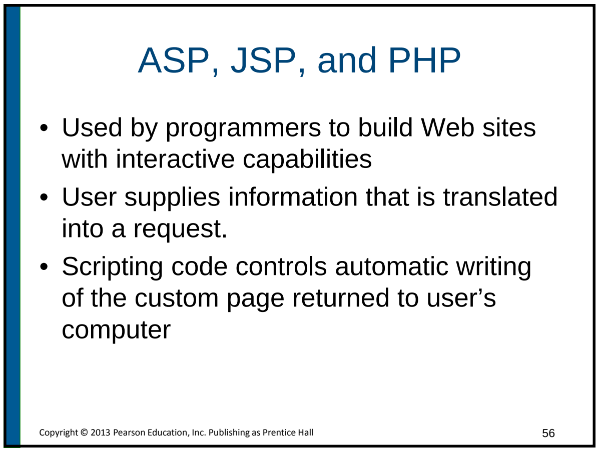
Building Web Applications

- HTML/XHTML
  - HyperText Markup Language/eXtensible HyperText Markup Language
  - Not a true programming language
  - Uses special symbols (tags) to control how Web pages are viewed



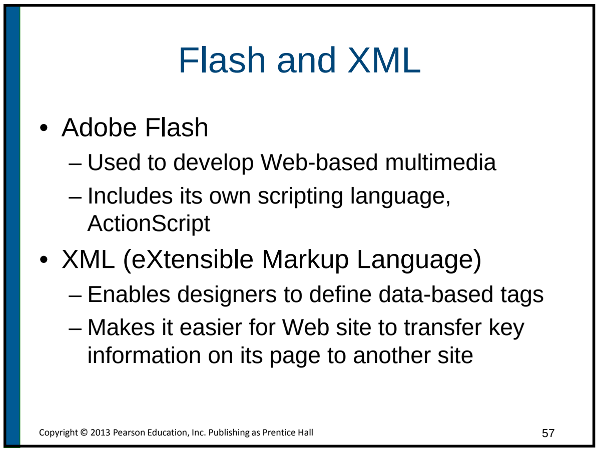
Scripting Languages for the Web

- Simple programming language limited to performing a set of specialized tasks
- Scripts allow decisions to be made and calculations to be performed
- JavaScript, VBScript, and PHP work well with HTML



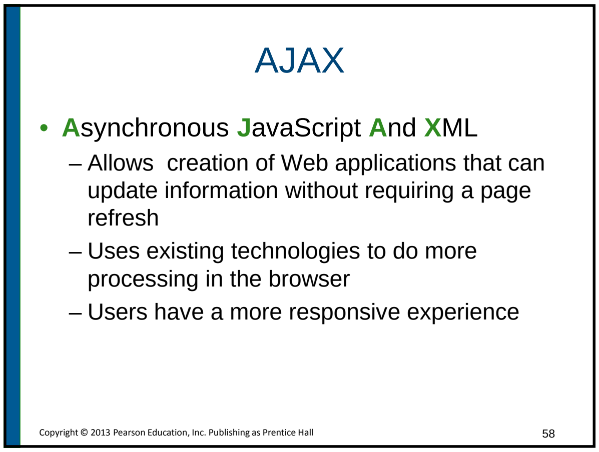
ASP, JSP, and PHP

- Used by programmers to build Web sites with interactive capabilities
- User supplies information that is translated into a request.
- Scripting code controls automatic writing of the custom page returned to user’s computer



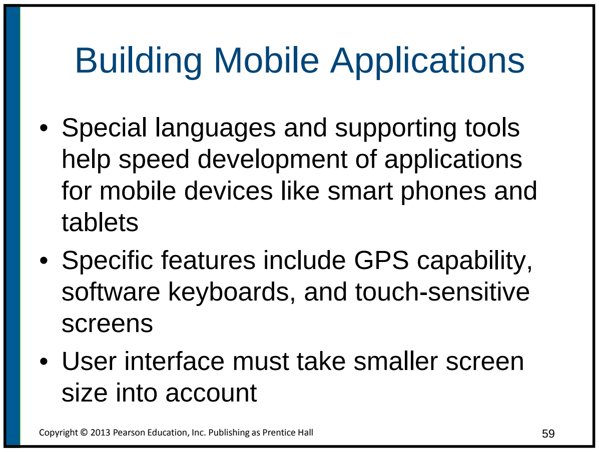
Flash and XML

- Adobe Flash
  - Used to develop Web-based multimedia
  - Includes its own scripting language, ActionScript
- XML (eXtensible Markup Language)
  - Enables designers to define data-based tags
  - Makes it easier for Web site to transfer key information on its page to another site



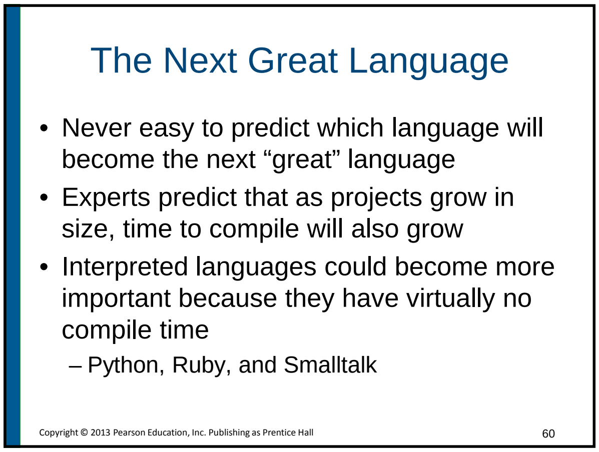
AJAX

- **A**synchronous **J**avaScript **A**nd **X**ML
  - Allows creation of Web applications that can update information without requiring a page refresh
  - Uses existing technologies to do more processing in the browser
  - Users have a more responsive experience



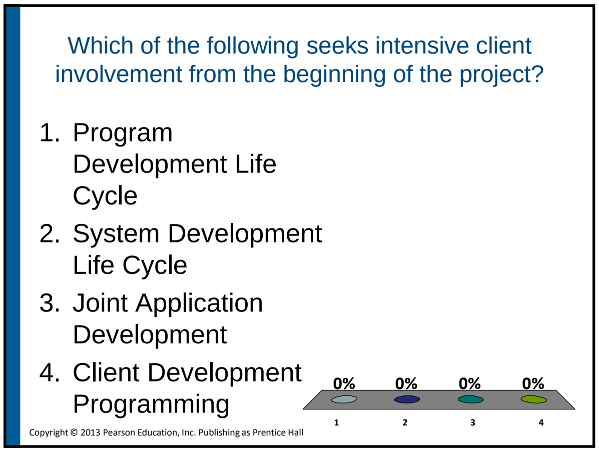
Building Mobile Applications

- Special languages and supporting tools help speed development of applications for mobile devices like smart phones and tablets
- Specific features include GPS capability, software keyboards, and touch-sensitive screens
- User interface must take smaller screen size into account



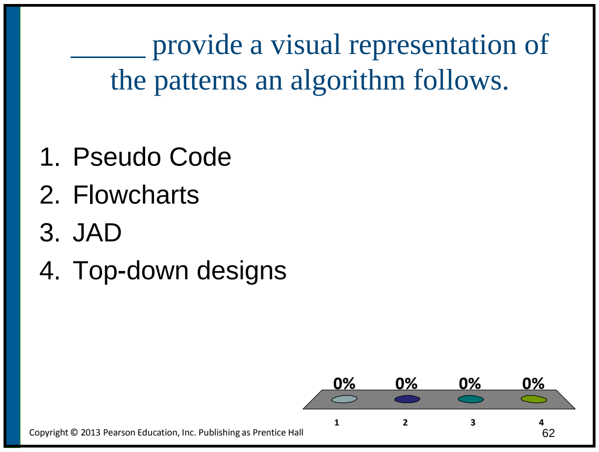
The Next Great Language

- Never easy to predict which language will become the next “great” language
- Experts predict that as projects grow in size, time to compile will also grow
- Interpreted languages could become more important because they have virtually no compile time
  - – Python, Ruby, and Smalltalk



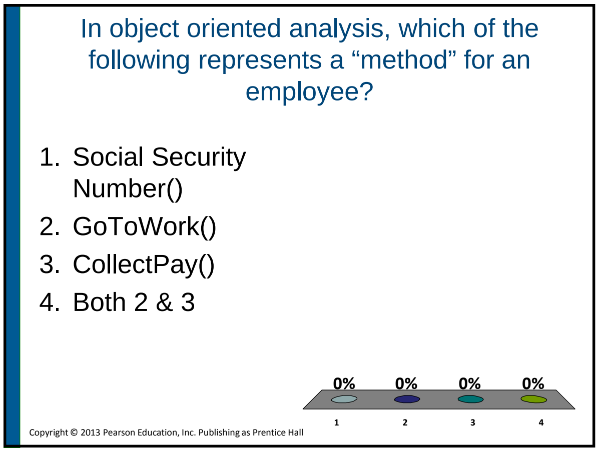
Which of the following seeks intensive client involvement from the beginning of the project?

1. Program Development Life Cycle
2. System Development Life Cycle
3. Joint Application Development
4. Client Development Programming



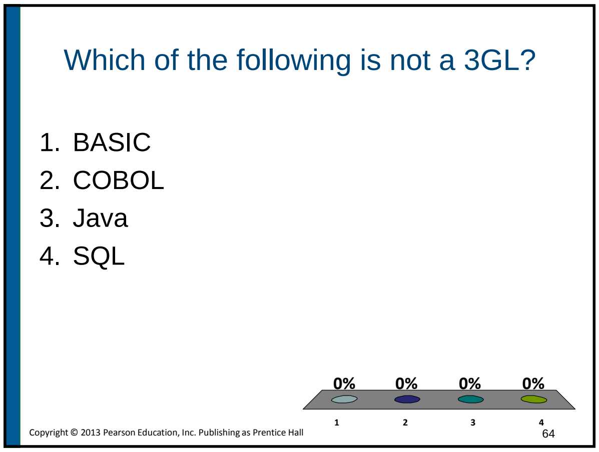
______ provide a visual representation of the patterns an algorithm follows.

1. Pseudo Code
2. Flowcharts
3. JAD
4. Top-down designs



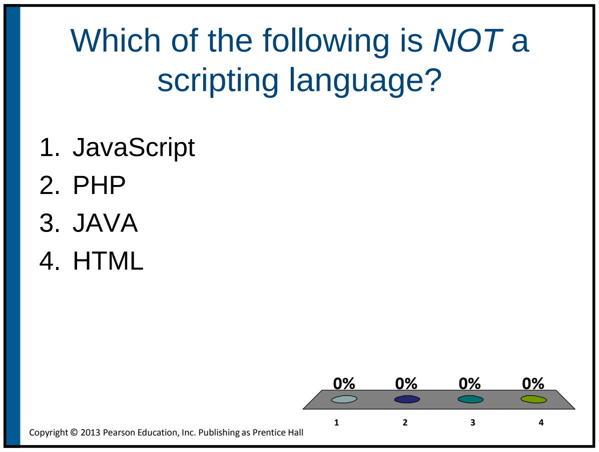
In object oriented analysis, which of the following represents a “method” for an employee?

1. Social Security Number()
2. GoToWork()
3. CollectPay()
4. Both 2 & 3



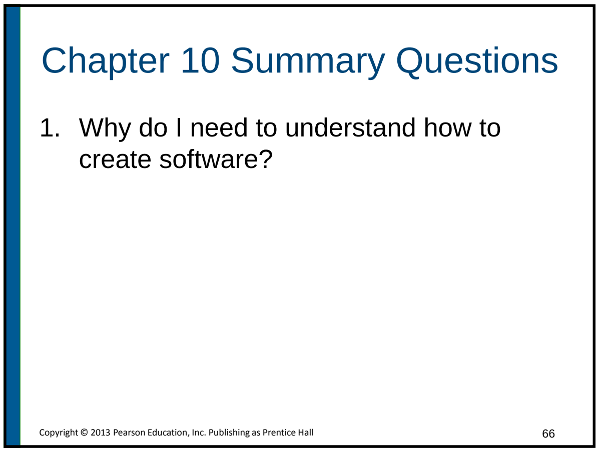
Which of the following is not a 3GL?

1. BASIC
2. COBOL
3. Java
4. SQL



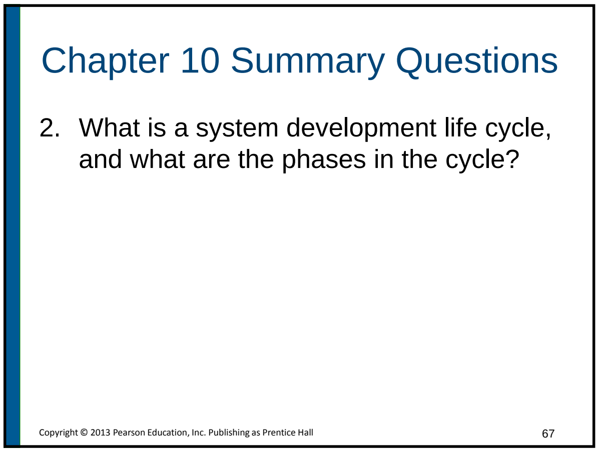
Which of the following is *NOT* a scripting language?

1. JavaScript
2. PHP
3. JAVA
4. HTML



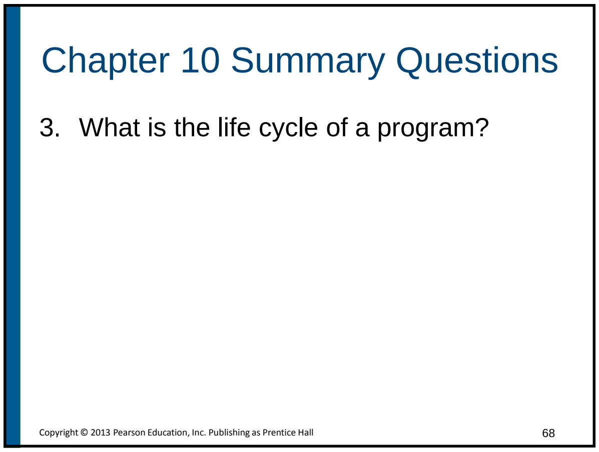
Chapter 10 Summary Questions

1. Why do I need to understand how to create software?



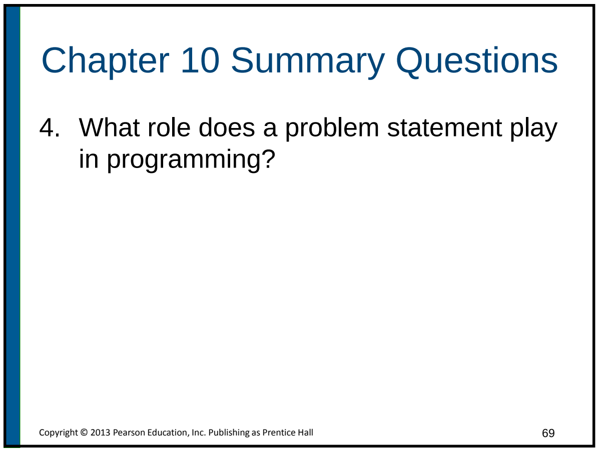
Chapter 10 Summary Questions

2. What is a system development life cycle, and what are the phases in the cycle?



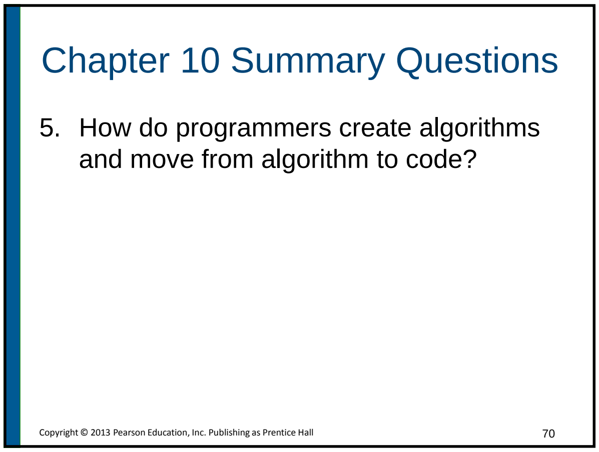
Chapter 10 Summary Questions

3. What is the life cycle of a program?



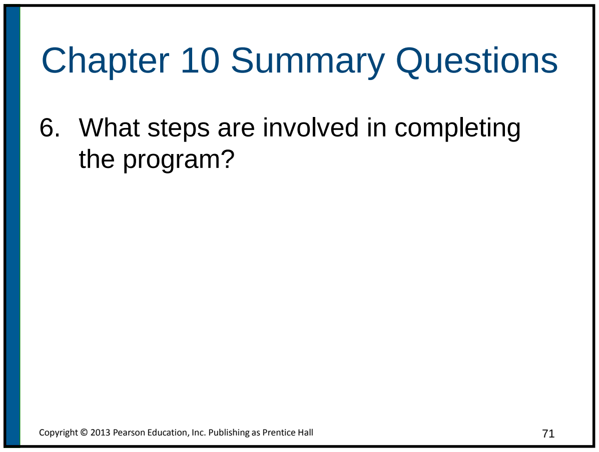
Chapter 10 Summary Questions

4. What role does a problem statement play in programming?



Chapter 10 Summary Questions

5. How do programmers create algorithms and move from algorithm to code?



Chapter 10 Summary Questions

6. What steps are involved in completing the program?

Chapter 10 Summary Questions

7. How do programmers select the right programming language for a specific task?

Chapter 10 Summary Questions

8. What are the most popular programming languages for different types of application development?