

# Chapter 12

## Computer Programming

# Computer Concepts 2014



# Chapter Contents

- Section A: Programming Basics
- Section B: Procedural Programming
- Section C: Object-Oriented Programming
- ~~Section D: Declarative Programming~~
- Section E: Secure Programming

# Computer Programming and Software Engineering

A program is a sequence of instructions that the computer understands

- In a language the computer understand
- Programs are often millions of lines of code
  - Developed by teams of computer programmers
  - Over many years

# Call of Duty



Call of Duty: Ghosts (recently released)

- Costs: roughly \$30-\$50 million to make
- Guess: \$150? million to promote
- Shipped \$1 billion dollars of product to retailers for launch
- Call of Duty: Black Ops II (released in Nov 2012)
- 24.2 million copies sold – over \$1 billion in sales

# 12

## Pacman (1980)

- grossed ~ \$3.5 billion
- Cost ~ \$100,000 to make
- Spawned of other games
  - Ms Pacman, Pacman Jr
  - Cartoon show, top 10 song, etc
  - Merchandising
- Google Pacman:  
<https://www.google.com/doodles/30th-anniversary-of-pac-man>



# Programming Languages

Every CPU has a set of instructions (language) it understands.

- We create programs for a specific CPU
- We often don't write programs directly in the language it understands
- **Low-level languages** include commands specific to a particular CPU or microprocessor family
- **High-level languages** use command words and grammar rules based on human languages

# Programming Languages and Paradigms

- First-generation languages
  - Machine language
  - Written in binary: 0's and 1's
- Second-generation languages
  - Assembly Language
  - English form of Machine language
  - Very simple instructions that the CPU understands

# Programming Languages

- Third-generation languages
  - This is normally what we program in
  - More English like
  - Examples: C, C++, Java, Fortran, etc

Like all languages, there are “words” that have meaning

- Keywords or command words

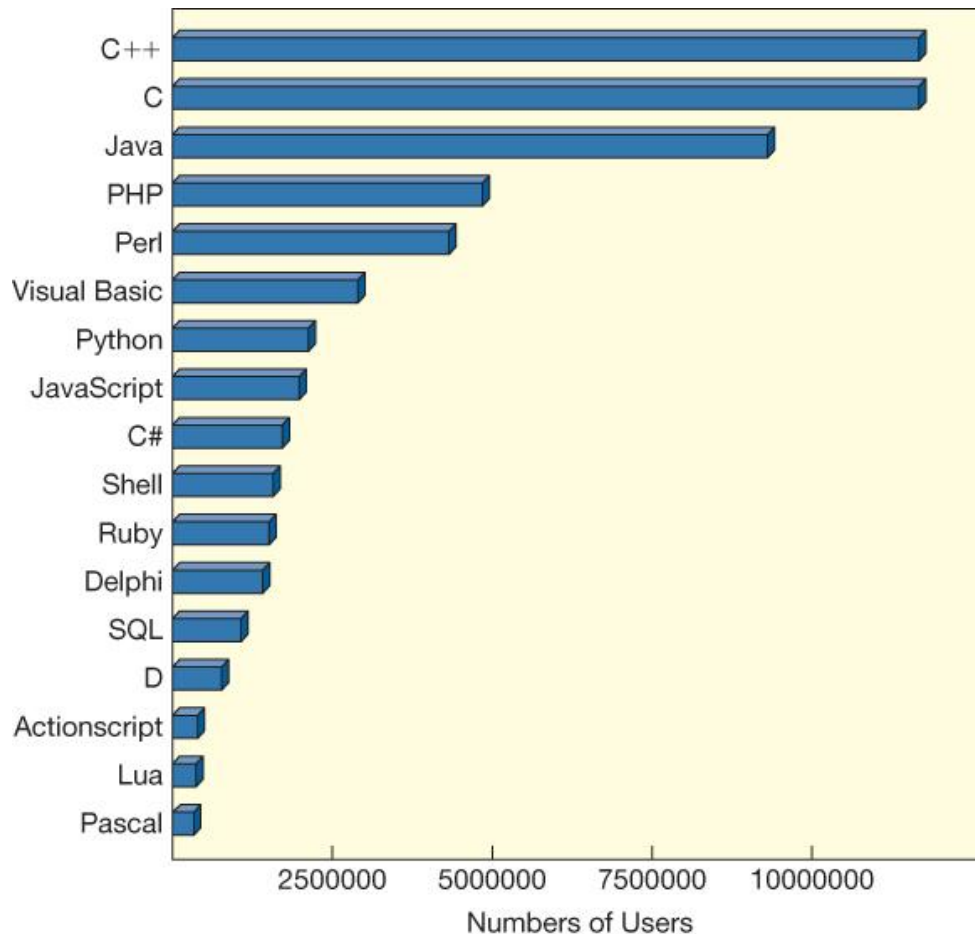
Rules that tell us proper form

- Grammar Rules that specify format

**Source code:** the “English-like” “human readable” code a programmer writes in these languages



# Popularity of Programming Languages



- C/C++ and Java are among the most popular programming languages.

# Programming Languages

- Fourth-generation languages
  - More closely resembles human language
- Fifth-generation languages
  - Command words do more complicated tasks that follow an algorithm

# Programming Languages

With successive generations, the tasks each command word performs is more complicated (higher level of thinking).

- Machine Language: Op Code: 000 0010 0000
- Assembly: Add (two operands)
- 3<sup>rd</sup> Gen: (define PMT (\* AMT (/ (\* r (expt (+ 1 r) t))  
(- (expt (+ 1 r) t) 1))))
- 4<sup>th</sup>/5<sup>th</sup> Gen: (sort (list 3 7 5 2 6 445 3 53 2 4 1 99 85 42))

## 12

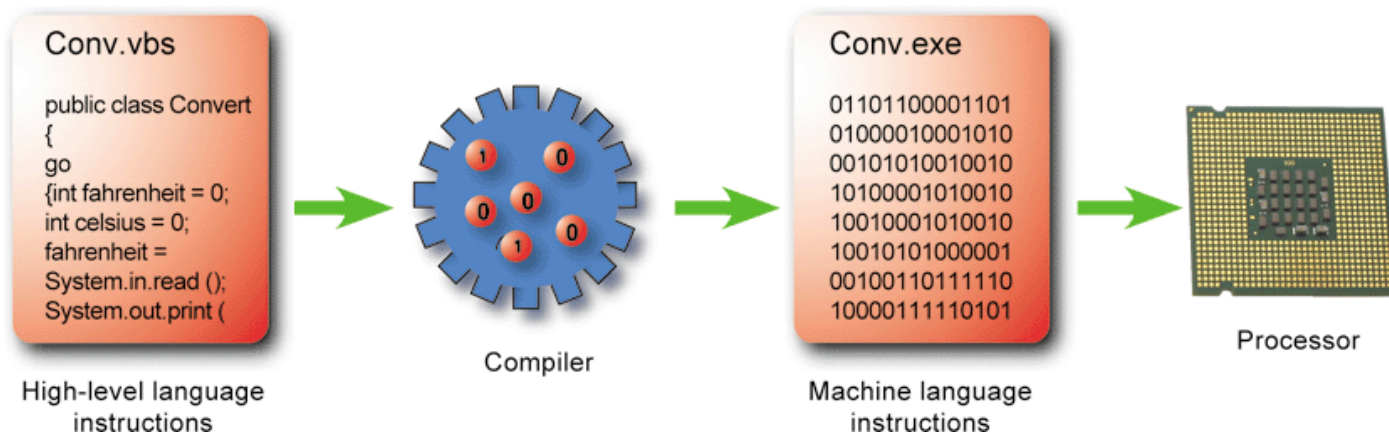
# Programming Languages

- The CPU only understands Machine Language
- Ultimately source code is compiled into machine language
- Stored as a file, loaded into RAM when executed and CPU reads the instructions from RAM

**FIGURE 1-34**

A compiler converts statements written in a high-level programming language into object code that the processor can execute.

▶ Watch a compiler in action.



# 12 Mark's "Bug" Interlude

- 1<sup>st</sup> computer bug (1947) really was a bug!
- Most computer programs contain bugs
  - That's what patches are for right?

Some favourites:

- Millenium Bug 1999 to 2000 changeover
- Many US rockets and space probes
- Intel Pentium CPU rounding error~0.006%
- Toyota Prius – recall 160,000 cars
- [http://en.wikipedia.org/wiki/List\\_of\\_software\\_bugs](http://en.wikipedia.org/wiki/List_of_software_bugs)

## 12

# Programming Languages and Paradigms

- A programming paradigm refers to a way of conceptualizing and structuring the tasks a computer performs

**FIGURE 12-8**

Programming Paradigms

Paradigm	Languages	Description
Event-driven	Visual Basic, C#	Focuses on selecting user interface elements and defining event-handling routines that are triggered by various mouse or keyboard activities
Procedural	BASIC, Pascal, COBOL, Fortran, Ada	Emphasizes linear steps that provide the computer with instructions on how to solve a problem or carry out a task
Object-oriented	Smalltalk, C++, Java, Scratch	Formulates programs as a series of objects and methods that interact to perform a specific task
Declarative	Prolog	Focuses on the use of facts and rules to describe a problem

# Programming Paradigms

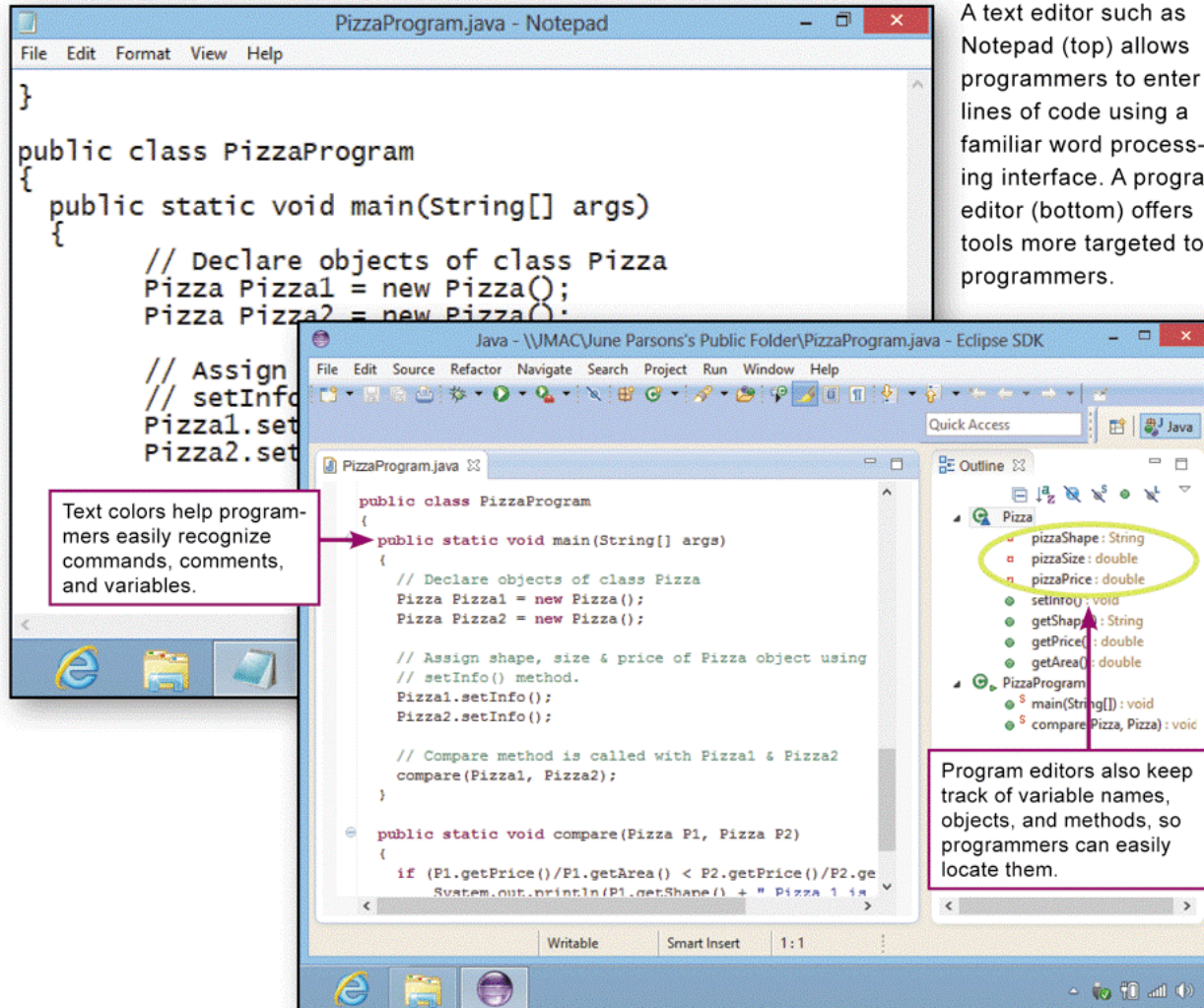
Programming has evolved to be more productive.

- Newer Paradigms include Object Oriented and Event Driven
- Emphasize reuse of code and human-computer interaction

Programs are very long so anything that can be reused saves time.



# Program Coding





# Program Coding

➤ A VDE (visual development environment) provides programmers with tools to build substantial sections of a program

- Form design grid
- Control
- Properties
- Event
- Event-handling code

A text box collects users' typed input.

An option button can be selected by clicking the circle.

The screenshot shows a Windows-style window titled "Pizza" containing a form titled "The Best Pizza Deal". The form has a "Best Deal" button at the top right. Below the title, there are two columns for "Pizza 1" and "Pizza 2". Each column has a "Pizza Price" text box and a "Pizza Size" text box. Underneath, there are two "Pizza Shape" sections, each containing two radio buttons and a corresponding pizza image. A callout box points to the first radio button, stating "An option button can be selected by clicking the circle." Another callout box points to the "Best Deal" button, stating "A button waits for a user's mouse click." A third callout box points to one of the pizza images, stating "A picture box holds a graphic." At the bottom of the form is a text box. The copyright notice "© MediaTechnics" is visible at the bottom right of the window.

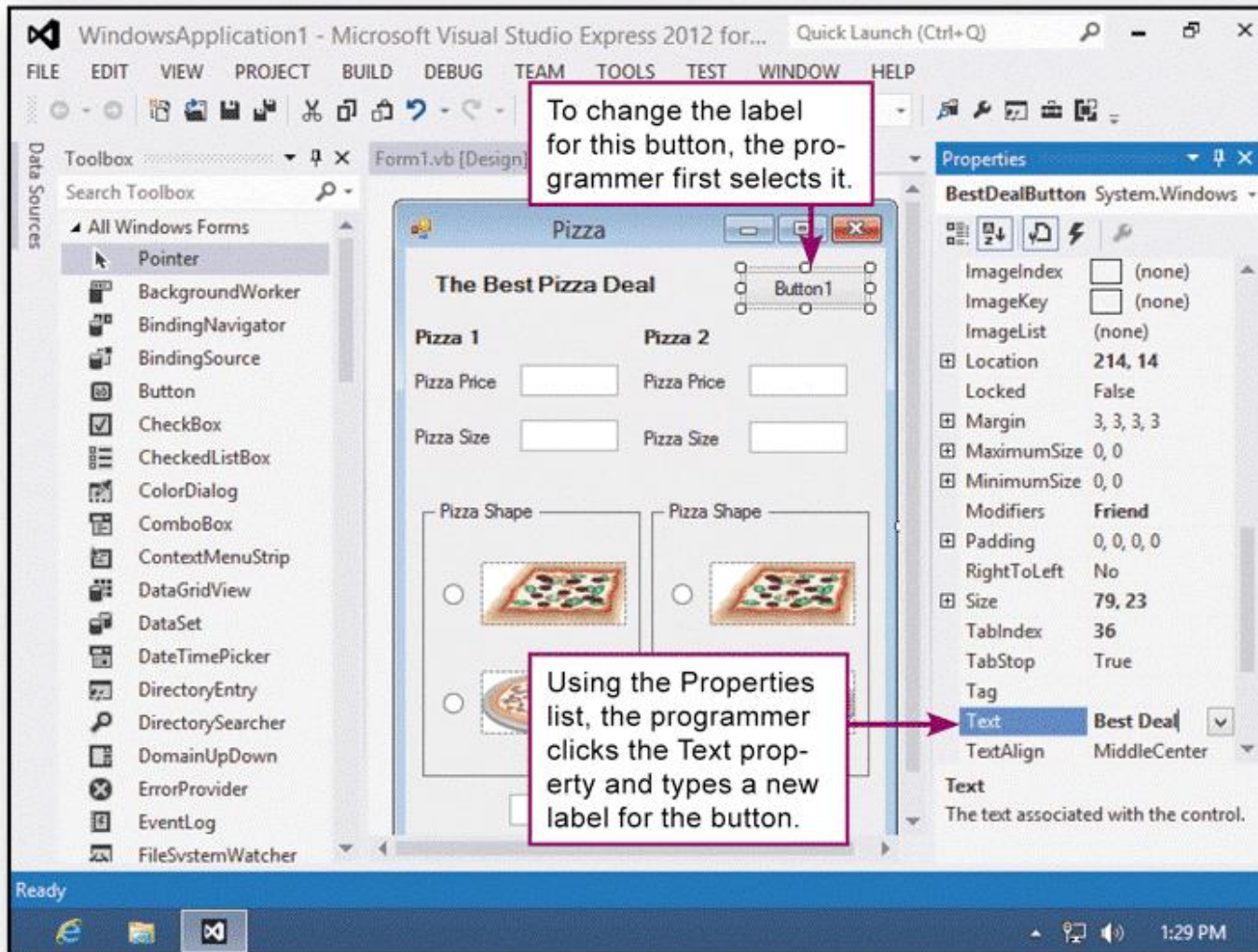
A button waits for a user's mouse click.

A picture box holds a graphic.

**FIGURE 12-12**

A form design grid is an important part of a VDE. This form was designed for the pizza program using Visual Basic.

# Program Coding



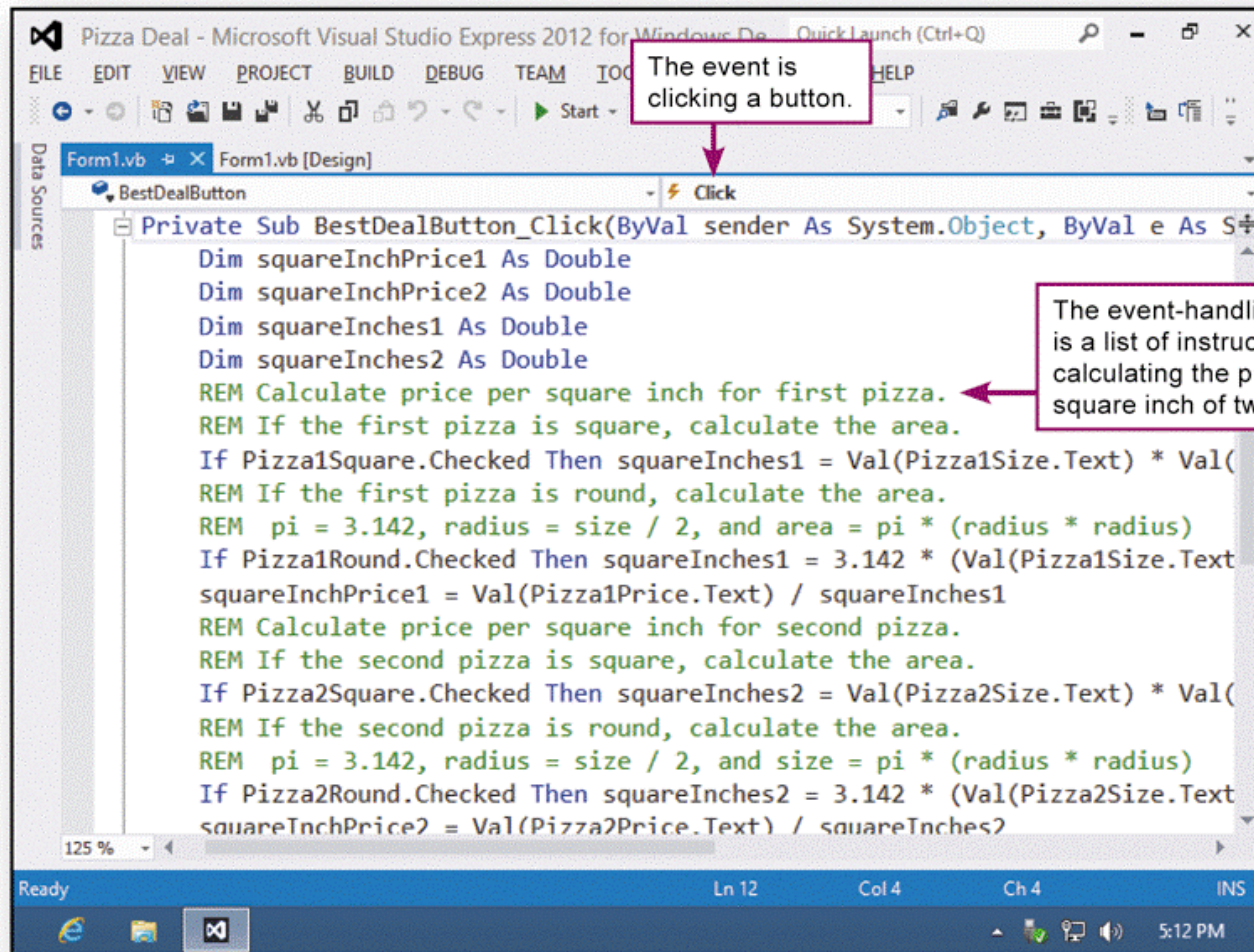
**FIGURE 12-13**

Controls, such as buttons, can be selected by a programmer from a properties list. Here, a programmer is changing the text property of a button so that its label will be "Best Deal". ▶ Learn how to work with properties in a VDE.

# Program Coding

**FIGURE 12-14**

The “Best Deal” event-handling code tells the computer what to do when users click the Best Deal button.



```
Private Sub BestDealButton_Click(ByVal sender As System.Object, ByVal e As S
    Dim squareInchPrice1 As Double
    Dim squareInchPrice2 As Double
    Dim squareInches1 As Double
    Dim squareInches2 As Double
    REM Calculate price per square inch for first pizza.
    REM If the first pizza is square, calculate the area.
    If Pizza1Square.Checked Then squareInches1 = Val(Pizza1Size.Text) * Val(
    REM If the first pizza is round, calculate the area.
    REM pi = 3.142, radius = size / 2, and area = pi * (radius * radius)
    If Pizza1Round.Checked Then squareInches1 = 3.142 * (Val(Pizza1Size.Text
    squareInchPrice1 = Val(Pizza1Price.Text) / squareInches1
    REM Calculate price per square inch for second pizza.
    REM If the second pizza is square, calculate the area.
    If Pizza2Square.Checked Then squareInches2 = Val(Pizza2Size.Text) * Val(
    REM If the second pizza is round, calculate the area.
    REM pi = 3.142, radius = size / 2, and size = pi * (radius * radius)
    If Pizza2Round.Checked Then squareInches2 = 3.142 * (Val(Pizza2Size.Text
    squareInchPrice2 = Val(Pizza2Price.Text) / squareInches2
```

The event is clicking a button.

The event-handling code is a list of instructions for calculating the price per square inch of two pizzas.

# Program Testing

- A computer program must be tested to ensure that it works correctly
- Program errors include:
  - Syntax errors
  - Runtime errors
  - Logic errors
- A debugger can help a programmer read through lines of code and solve problems



# Program Documentation

- Remarks or “comments” are a form of documentation that programmers insert into the program code

**FIGURE 12-16**

A series of remarks in a BASIC program can explain to programmers the method used to calculate the square inches in a round pizza.

```
Rem The program calculates the number of square inches  
Rem in a round pizza using the formula pi r squared  
Rem pi = 3.142, size / 2 = radius,  
Rem and (size / 2) ^2 = radius squared  
Rem SquareInches = 3.142 * (size / 2) ^2
```

# Programming Tools

- An SDK (software development kit) is a collection of language-specific programming tools that enables a programmer to develop applications for a specific computer platform
  - Java SDK
- An IDE (integrated development environment) is a type of SDK that packages a set of development tools into a sleek programming application
  - Eclipse – provides debugger, keyword highlighting, etc

# Programming Tools

- A component is a prewritten module, typically designed to accomplish a specific task
- An API is a set of application program or operating system functions that programmers can access from within the programs they create
- C, Java, and C++ are the most popular programming languages
- Microsoft's XNA framework is a set of tools for creating Xbox 360 games
- Objective-C is popular for creating apps for iPhones and iPads

# Section B: Procedural Programming

- Algorithms
- Expressing an Algorithm
- Sequence, Selection, and Repetition Controls
- Procedural Languages and Applications



# Algorithms

- Set of steps for carrying out a task that can be written down and implemented
- Start by recording the steps you take to solve the problem manually
- Specify how to manipulate information
- Specify what the algorithm should display as a solution

# Algorithms

**FIGURE 12-21**

The algorithm for the pizza problem, written in structured English, has five main sections.

1. Get initial information for the first pizza.

Ask the user for the shape of the first pizza and hold it in RAM as Shape1.  
Ask the user for the price of the first pizza and hold it in RAM as Price1.  
Ask the user for the size of the first pizza and hold it in RAM as Size1.

2. Calculate the price per square inch for the first pizza.

If Shape1 is square then  
    calculate the square inches using the formula:  
     $\text{SquareInches1} = \text{Size1} * \text{Size1}$   
If Shape1 is round then  
    calculate the square inches using the formula:  
     $\text{SquareInches1} = 3.142 * (\text{Size1} / 2) ^2$   
 $\text{SquareInchPrice1} = \text{Price1} / \text{SquareInches1}$

3. Get initial information for the second pizza.

Ask the user for the shape of the second pizza and hold it in RAM as Shape2.  
Ask the user for the price of the second pizza and hold it in RAM as Price2.  
Ask the user for the size of the second pizza and hold it in RAM as Size2.

4. Calculate the price per square inch for the second pizza.

If Shape2 is square then  
    calculate the square inches using the formula:  
     $\text{SquareInches2} = \text{Size2} * \text{Size2}$   
If Shape2 is round then  
    calculate the square inches using the formula:  
     $\text{SquareInches2} = 3.142 * (\text{Size2} / 2) ^2$   
 $\text{SquareInchPrice2} = \text{Price2} / \text{SquareInches2}$

5. Compare the prices per square inch, then output the results.

If  $\text{SquareInchPrice1} < \text{SquareInchPrice2}$  then  
    display the message "Pizza 1 is the best deal."  
If  $\text{SquareInchPrice2} < \text{SquareInchPrice1}$  then  
    display the message "Pizza 2 is the best deal."  
If  $\text{SquareInchPrice1} = \text{SquareInchPrice2}$  then  
    display the message "Both pizzas are the same deal."

# Expressing an Algorithm

- Structured English
- Pseudocode

```
display prompts for entering shape, price, and size
input Shape1, Price1, Size1
if Shape1 = square then
    SquareInches1 ← Size1 * Size1
if Shape1 = round then
    SquareInches1 ← 3.142 * (Size1 / 2) ^2
SquareInchPrice1 ← Price1 / SquareInches1
display prompts for entering shape, price, and size
input Shape2, Price2, Size2
if Shape2 = square then
    SquareInches2 ← Size2 * Size2
if Shape2 = round then
    SquareInches2 ← 3.142 * (Size2 / 2) ^2
SquareInchPrice2 ← Price2 / SquareInches2
if SquareInchPrice1 < SquareInchPrice2 then
    output "Pizza 1 is the best deal."
if SquareInchPrice2 < SquareInchPrice1 then
    output "Pizza 2 is the best deal."
if SquareInchPrice1 = SquareInchPrice2 then
    output "Both pizzas are the same deal."
```

**FIGURE 12-22**

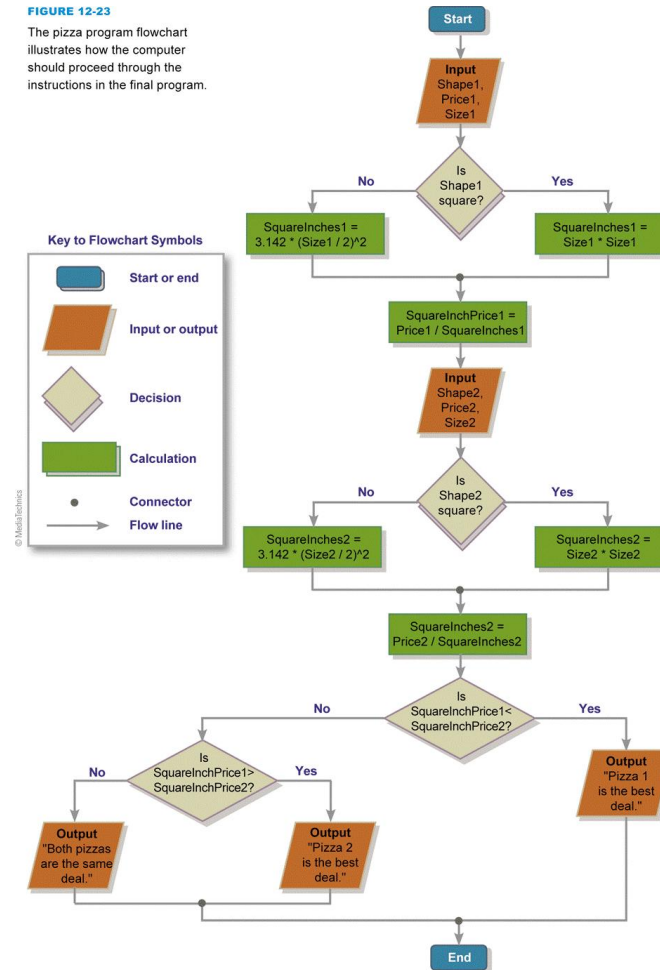
Pseudocode for the pizza program mixes some English-like instructions, such as display prompts, with programming commands, such as INPUT.

# Expressing an Algorithm

## ➤ Flowchart

FIGURE 12-23

The pizza program flowchart illustrates how the computer should proceed through the instructions in the final program.



© MediaInches

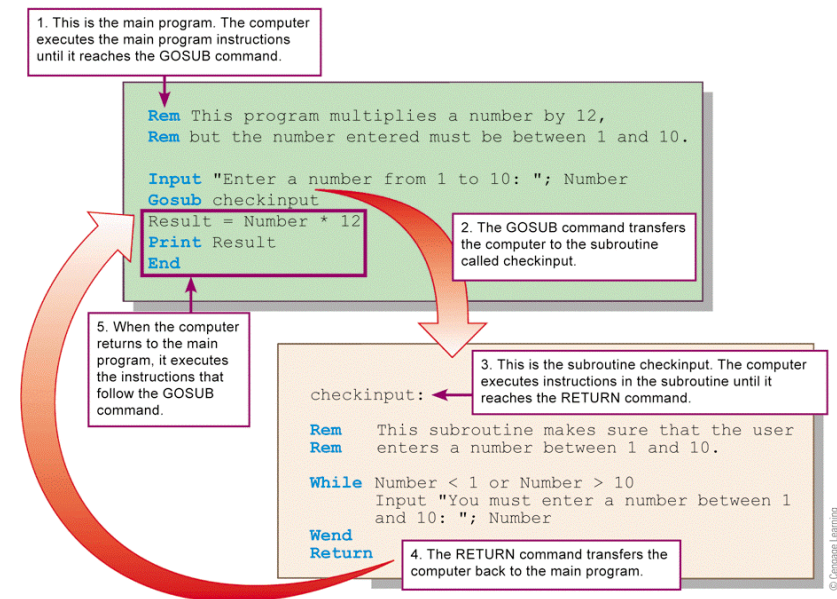


# Sequence, Selection, and Repetition Controls

- Subroutines, procedures, and functions are sections of code that are part of the program, but not included in the main sequential execution path

FIGURE 12-26

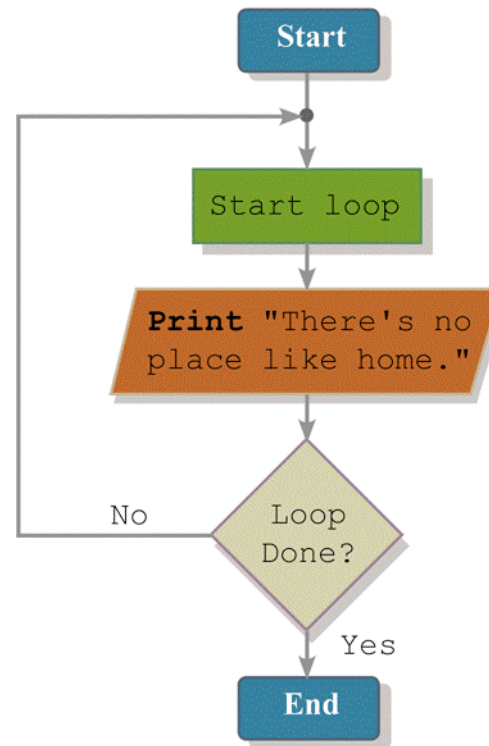
Executing a GOSUB command directs the computer to a different section of the program.



## 12

# Sequence, Selection, and Repetition Controls

## ➤ Repetition control structure



**FIGURE 12-28**

To execute a loop, the computer repeats one or more commands until some condition indicates that the looping should stop.

▶ Remember Dorothy in *The Wizard of Oz*? Watch how a computer would execute her "There's no place like home." loop.

© MediaTechnics

# Procedural Languages and Applications

- Popular procedural languages: COBOL, FORTH, APL, ALGOL, PL/1, Pascal, C, Ada, and BASIC
- The procedural approach is best for problems that can be solved by following a step-by-step algorithm

# Section C: Object-Oriented Programming

- Objects and Classes
- Inheritance
- Methods and Messages
- Object-oriented Program Structure
- Object-oriented Languages and Applications



# Object-Oriented Programming

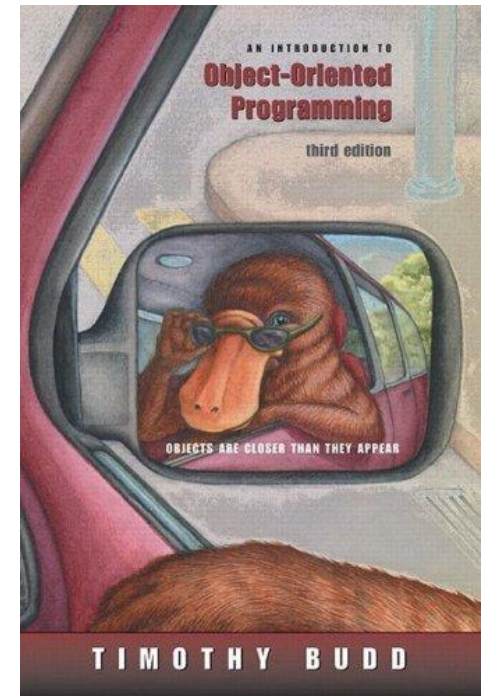
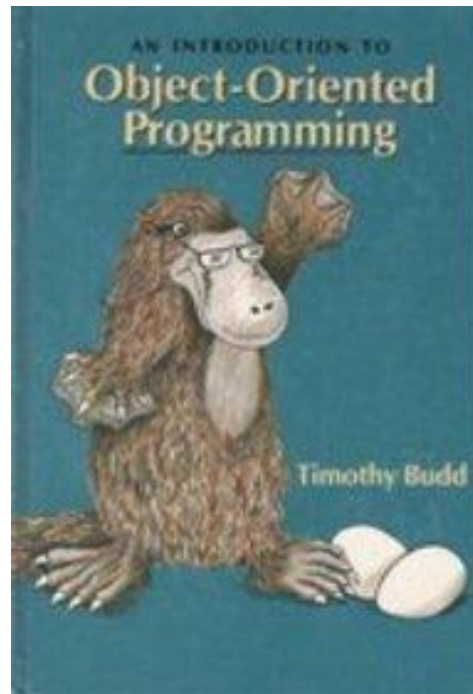
- Basic Idea:
  - In procedural programming, the data and functions that operate on the data are separate
  - In OOP, both data and the functions that operate on the data (called methods in OOP) are combined together in an *object*
- The textbook throws a lot of OOP terms at you that are well beyond the scope of this course – we would teach them in 2<sup>nd</sup> year computer science at WLU.
  - Inheritance, Encapsulation, Polymorphism, Overloading, Abstraction

# 12

# Object-Oriented Programming

- For more information on these terms
  - Inheritance, Encapsulation, Polymorphism, Overloading, Abstraction

Try “The Platypus Book”:



# 12

## Objects - Example

- An object represents an abstract or real-world entity
  - Such as a pizza
- The attributes of an object define the characteristics of the object – the data
  - Shape, size, toppings, etc
- Methods perform operations based on the attributes
  - Calculate number of toppings on the pizza, number of slices, etc

**CLASS: Pizza**



Pizza objects

**FIGURE 12-30**

A class, such as the Pizza class, is a general template for a group of objects with similar characteristics.

# Section E: Secure Programming

- Black Hat Exploits
- Secure Software Development
- Mitigation

# 12

## Hackers

- Anyone who unlawfully accesses a computer system
- Types of hackers
  - White hat
  - Black hat
  - Script kiddies



# Black Hat Exploits

- Viruses, worms, bots, malicious Web scripts, and other exploits creep into computer systems
  - Black-hat exploits
- Morris Worm (1988) – infected 6000 computers in 1 day
  - Intended to count the number of computers on the internet
- Antivirus software finds viruses by looking for a *virus signature* – part of the virus program code itself

# Secure Software Development

- Most software security problems can be traced back to defects that programmers unintentionally introduce in software during design and development
- Formal methods help programmers apply rigorous logical and mathematical models to software design, coding, testing, and verification
- Threat modeling (risk analysis)

# Secure Software Development

- Defensive programming (also referred to as secure programming) is an approach to software development in which programmers anticipate what might go wrong as their programs run and take steps to smoothly handle those situations
  - Source code walkthroughs
  - Simplification
  - Filtering input



# Secure Software Development

- Signed code is a software program that identifies its source and carries a digital certificate attesting to its authenticity

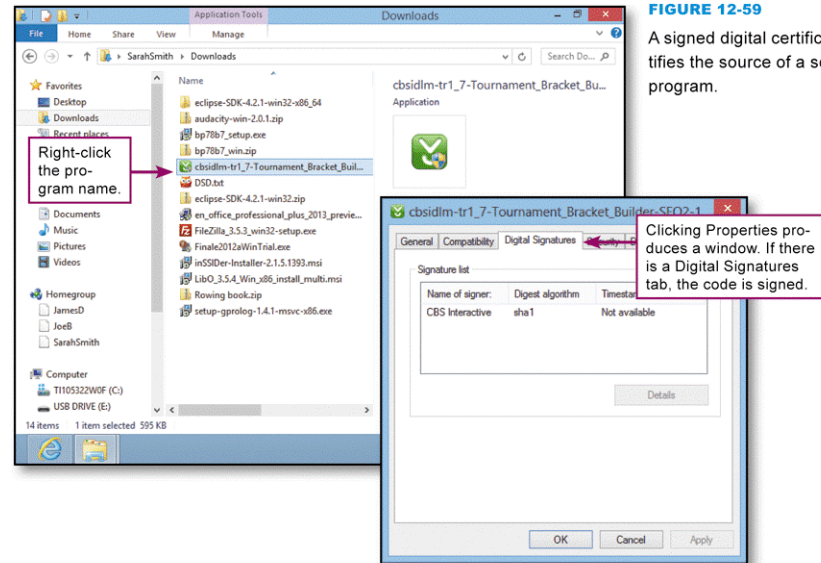


FIGURE 12-59

A signed digital certificate identifies the source of a software program.

Clicking Properties produces a window. If there is a Digital Signatures tab, the code is signed.

# Mitigation

- Despite defensive programming and other tactics to produce secure software, some defects inevitably remain undiscovered in products that end up in the hands of consumers
- When bugs are discovered, the programmer's remaining line of defense is to produce a bug fix or patch

# Mitigation

- Take the following steps to avoid security problems that stem from software defects:
  - Select applications from software publishers with a good security track record
  - Read reviews of products before you download them
  - Watch for patches and apply them
  - Consider using open source software, which has been extensively reviewed by the programming community
  - Keep your firewall and antivirus software deployed and up to date

# Chapter 12 Complete

## Computer Concepts 2014

