A Question

What are the five largest cities in the United States?

(Write down your answers in order on a piece of scratch paper)

The answer

Cities, by population
1. New York (8M)
2. Los Angeles (3.7M)
3. Chicago (2.9M)
4. Houston (2M)
5. Philadelphia (1.5M)

(Seattle is #24 at 560K)

What We Do Best And What Computers Do Best Are VERY DIFFERENT Things

❖ People are extremely good at:
  ❖ Resolving ambiguity
  ❖ Taking context (the particular situation) into account when processing information

❖ Computers are very good at:
  ❖ Following explicit instructions over, and over, and over....
  ❖ Never tiring of the same old routine

❖ Computer are NOT very good at:
  ❖ Resolving ambiguity
  ❖ Figuring out the “right” meaning based on a particular situation

❖ So if we want to tell a computer what to do, we must do so precisely and unambiguously

The Basics of Programming

To specify algorithms, especially to a computer, we must be precise. To be precise, we need a language that is more exact than our own. A programming language offers this advantage. All programming languages have a basic set of features
What’s Different About Programming Languages?

❖ The Alphabetize CD’s algorithm (see FIT 9) was precise enough for a person to execute successfully, but computers must have greater precision.

❖ Programming languages are formal notations specifically designed for specifying algorithms — that means each “word” or “sentence” in a programming language has one and only one interpretation.

❖ The programming language we will study this quarter is Visual Basic 6.0 (VB6).

What’s Different About Programming Languages?

❖ Programming involves two critical and interrelated tasks:
  ❖ Figuring out/understanding intuitively what steps need to be taken
  ❖ Figuring out how to specify those steps precisely

Introduction to Programming Concepts

❖ There are just a few general concepts that apply to virtually all programming languages.

❖ Once you have been exposed to them, you will practice your language proficiency using the Visual Basic IDE.

❖ In this environment you will take the general concepts you know and by adding other language features, implement programs with varying levels of complexity that become more involved over time.

Order Matters

❖ CONCEPT: Programming languages execute instructions in order (unless told to do otherwise…we’ll get to that point later)

❖ The first things listed in a program get done first

❖ Each instruction is executed one at a time – then the computer goes on to execute the next instruction

❖ Remember your web pages? The computer (browser) executed the HTML code in the order you wrote the statements.
General Concepts

❖ CONCEPT: Being able to store, “remember”, change and access data allows us to write programs that do the same thing but with different data each time.

❖ The following programming concepts are key:
  ❏ Variables, Names, Values
  ❏ Assignments
  ❏ Expressions
  ❏ Conditionals
  ❏ Iteration

❖ We will cover the first several of these concepts today

Also important:
❖ Objects
❖ Events

Names of Variables

❖ Using the term “variable” reminds us that the value can change, that it can vary

❖ The names used for variables are arbitrarily provided:
  ❏ Variable names must begin with a letter
  ❏ Variable names can contain any letter, numeral or _
  ❏ Most languages are case sensitive: a is different than A

❖ Good variable names are meaningful and accurate
  ❏ Total, avgOfClass, temp, etc. But not x, tToO, y83928 etc.

VB6: In all programming for FIT100, variable names should start with lowercase letters so as to avoid confusion with other reserved names in VB6. Ignore this convention at your own peril!

Values of Variables

❖ Values refer to the information stored in the variable (location in memory)

❖ Variables can take on different types of values
  ❏ Whole numbers or integers: 2, -9, 36452729
  ❏ Character sequences or strings: “2”, “dog”, “die90wk”, “ ”
  ❏ Decimal numbers or doubles: 2.3, 3.14159, -666.99

❖ In most programming languages, each variable should only hold one type of value. This is to:
  ❏ Let the computer know how much memory will be needed to store
  ❏ Allow the computer to help detect errors in the code. For example, when the program tries to put the wrong sort of value in a variable the programmer receives an error message
Declaring Variables

❖ Variable declaration tells the computer:
   ❖ That you want a location in memory (the variable)
   ❖ The way in which you will refer to that location in memory throughout your program (the variable name)
   ❖ What type of information you will store in that location in memory, so the computer will know how much space to set aside (the variable type)

❖ VB6 - some examples of declaring variables:
   ❖ Dim num1 As Integer
   ❖ Dim letter1 As String
   ❖ Dim avgOfClass As Double

   ✫ By the way, for Midterm 1, avgOfClass = 33.8 (out of 50!)

Assigning Values to Variables

❖ CONCEPT: Computers must be told what value to assign to variables

❖ CONCEPT: The general form of an assignment statement is <variable name> <assignment symbol> <expression>
   ❖ Each language may use a different assignment symbol: = :=
   ❖ Assignment means "gets", "becomes" or "is assigned" and we read it left to right: A = B A is assigned B
   ❖ All three components must always be present

❖ CONCEPT: Fundamental property of Assignment
   The flow of information is always right to left

❖ VB6: Some examples of variable assignment
   ❖ destination = "Chicago"
   ❖ changedVariable = value

A Class Demonstration

❖ We’ll use VB6 syntax for this example…

    Dim sleepy As Integer
    Dim grumpy As Integer
    Dim dopey As Integer
    sleepy = 9
    grumpy = 3
    dopey = 7
    sleepy = 12
    grumpy = sleepy
    dopey = grumpy

A Series of Assignments

❖ Now you work it out …

    Dim rock As Integer
    Dim paper As Integer
    Dim scissor As Integer
    rock = 9
    scissor = 3
    rock = 7
    scissor = 3
    rock = scissor
    scissor = 23
    paper = scissor
    rock = scissor + paper
    rock = scissor / paper

**Question:**
What’s in rock?
What’s in paper?
What is the Value of Dude?

 Dim dude As Integer
 dude = 0
 dude = dude + 1
 dude = dude + 1
 dude = dude + 1

❖ Questions:
1. What value does the variable \textit{dude} contain at the end of this code?
2. What is this code doing?
3. What would be a better variable name for \textit{dude}?

Expressions

❖ CONCEPT: Expressions are a means of performing the actual computation in a program. They are formulae made from variables and operators, e.g. calculator operations: 
\begin{itemize}
  \item [+] , [-], [\times], [\div], [^]
  \item weeks = days / 7 \hspace{1cm} \text{divide value of days by 7}
  \item totalAfterTax = totalPrice * 1.087 \hspace{1cm} \text{multiply the two values}
\end{itemize}

❖ The Fundamental Rules of Assignment:
\begin{itemize}
  \item The general form of an assignment statement is <variable name> <assignment symbol> <expression>
  \item The flow of information is always right - to – left
  \item The expression is evaluated before the assignment is made
  + score = score + 3