**Presentation Notes:**

Slide 1: IF Statement Control

1. Summarize the purpose of an ***if*** statement in programming.

1. Explain what happens in the sample program under the following conditions.
   1. classSize is greater than 30
   2. classSize is less than 30
   3. classSize is exactly equal to 30
2. Draw a sketch of the flowchart diagram   
   for an ***if*** statement.

Slide 2: Indentation & Code Blocks

1. Summarize the purpose and use of a code block in programming with respect to:
   1. Grouping conditional code statements
   2. Use of indentation

Slide 3: Conditional Expression

1. Summarize the Syntax (format and location) of a conditional expression.
2. Modify the sample program to check for a class size greater than or equal to 28.   
   Write your new conditional expression below.

Slide 4: Is Equal To (==)

1. Summarize the difference between ***==*** (is equal to) and ***=*** (assignment operator).
2. Explain what the ***!=*** comparison means.

Slide 5: ELSE Statement Control

1. Summarize the purpose of an ***else*** statement in programming.

1. Explain what happens in the sample program under the following conditions.
   1. classSize is greater than 30
   2. classSize is less than 30
   3. classSize is exactly equal to 30
2. Draw a sketch of the flowchart diagram   
   for an ***else*** statement.

Slide 6: elif Statement Control

1. Summarize the purpose of an ***elif*** statement in programming.

1. Explain what happens in the sample program under the following conditions.
   1. classSize is greater than 30
   2. classSize is less than 30
   3. classSize is greater than 14
   4. classSize is less than 14
2. Draw a sketch of the flowchart diagram   
   for an ***elif*** statement.

Slide 7: Program Comments

1. Summarize the purpose and syntax of a ***line*** comment.
2. Summarize the purpose and syntax of a ***block*** comment..

Slide 8: Conditional Loops

1. Summarize the purpose of an ***loop*** statement in programming.

1. Explain why you should not use “cut-and-paste” to repeat code blocks.
2. Draw a sketch of the flowchart diagram   
   for an ***loop*** statement.

Slide 9: While Loops

1. Explain how changes in the variable startCount affect the operation of the while loop.

1. Explain what happens to the while loop under the following conditions.
   1. When the condition statement is true
   2. When the condition statement is false

Slide 10: Infinite Loops

1. Explain what an infinite loop is and why it is a bad thing in a computer program.

1. Explain the difference between a Logic Error and a Syntax Error.

1. Explain the difference between a Logic Error and a Run-Time Error.

**Student Questions:**

Use the following code as a template for generating random numbers in your programs.

# This code generates a random number between 0 to 9

# Stores the value of the random number in the variable "targetNumber"

# Then prints out the result

import random

targetNumber = random.randint(0,9)

print("The random number %d is the target." % targetNumber)

1. Create a basic level program to implement a simple guessing game. Your program should do the following: (Provide a listing of your program below.)
   1. Generate and store a random number using the template code above
   2. Ask the user to guess the number (i.e. us the “input” command and sore the answer)
   3. Use an “if” statement to check if the guess is correct
   4. Print “Your guess was correct!” if the guess is correct
   5. Print “Try again” is the check is guess is wrong

# This code generates a random number between 0 to 9

# Stores the value of the random number in the variable "targetNumber"

# Then prints out the result

import random

targetNumber = random.randint(0,9)

print("The random number %d is the target." % targetNumber)

guess = int(input("Guess a number from one to ten!"))

if targetNumber!=guess:

print ("Try Again")

if targetNumber == guess:

print ("Your guess was correct!")

1. Create a medium level program to implement a better guessing game. Add the following to your basic level program: (Provide a listing of your program below.)
   1. Use “if” and “elif” statements to check if the guess is correct
   2. Print “Your guess was correct!” if the guess is correct
   3. Print “guess higher” if the guess was less than the target number
   4. Print “guess lower” if the guess was greater than the target number

# This code generates a random number between 0 to 9

# Stores the value of the random number in the variable "targetNumber"

# Then prints out the result

import random

targetNumber = random.randint(0,9)

print("The random number %d is the target." % targetNumber)

guess = int(input("Guess a number from one to ten!"))

if targetNumber < guess:

print ("Guess Lower")

if targetNumber > guess:

print ("Guess Higher")

elif targetNumber == guess:

print ("Your guess was correct!")

1. Create an enhanced level program to implement the ultimate guessing game. Add the following to your medium level program: (Provide a listing of your program below.)
   1. Add a loop to keep playing the game until the target number is guessed correctly
   2. Exit the loop when the target number is guessed correctly

# This code generates a random number between 0 to 9

# Stores the value of the random number in the variable "targetNumber"

# Then prints out the result

import random

targetNumber = random.randint(0,9)

print("The random number %d is the target." % targetNumber)

guess = int(input("Guess a number from one to ten!"))

while (targetNumber!=guess):

if (targetNumber < guess):

print ("Guess Lower")

guess = int(input("Please try again:"))

elif (targetNumber > guess):

print ("Guess Higher")

guess = int(input("Please try again:"))

else: print ("Your guess was correct!")