GWC Level 3

Week 2 C# Fundamentals

WIT Shout-out of the Week: Grace Brewster Murray Hopper

- Graduated from Vassar College in 1928 with a Bachelors in mathematics and physics and later earned a PhD in mathematics from Yale in 1934
- Later she joined the Navy after many attempts had been denied and worked on the Harvard Mark 1 computer after receiving training from the Naval Reserve Midshipmen's School
- She later worked as a research fellow at Harvard under a Naval contract where she began working on the idea of high level programming languages that used compliers.
- Later she developed Cobol -- one of the first high level languages like python, c , java that take human language and translate it into machine language or Assembly

Lesson Objectives

- Data Types and Variables
- Conditional Statements
- Loops
- Input / Output
- Methods

Data Types and Variables

- Types: define the blueprint for a value
  - In C# there are pre-defined types and custom types that you can create yourself
  - Pre-defined types are types that are specifically supported by the compiler:
    - int: a whole number that can fit into 32 bits of memory
    - string: a representation of a sequence of characters “hello”, “katie”
    - bool: a value that is either true or false -- these come from boolean algebra that deals with true and false mathematical statements
      - bool lessThanAMile = x < 5280 ← you define what x is later
      - bool test = true
  - Custom-Types: values that you create yourself using functions and primitive type values -- we will discuss these more later

- Variables: denote a storage location on your computer that can contain different values
  - Like java variables are ***strong typed*** meaning that you must define what type that value is before the name

```
int num = 5;
string name = "katie";
bool value = true;
bool otherValue = false;
```

Conditional Statements

What is a condition?

- Synonym: circumstance
  - It's required to happen before something else can happen
  - Ex: You must take Level 1 and 2 before you can be in Level 3
- Uses conditional operators
  - And (&&)
  - Or (||)
  - Equal To (==)
  - Greater Than (>)
  - Greater Than OR Equal To (>=)
  - Less Than (<)
  - Less Than OR Equal To (<=)

if


```
if (condition)
{
    //code
}
```

If the condition is met, the code within the if statement will run.

```
if (first_letter == 'C') {
  Console.WriteLine("You have a cool name!");
}
```

if

You can have multiple if statements - the compiler will check for each condition

Example:

```
if (input == 3) {
Console.Write(“the input is three”);
}
if (input == 4) {
Console.Write(“the input is four”);
}
```

else

```
if (condition)
{
        //code
}
else
{
        //code
}
```

If the condition is not met, then the code within the else statement will run.

YOU CANNOT HAVE AN ELSE STATEMENT WITHOUT AN IF STATEMENT!

else if

```
if (condition)
{
        //code
}
else if (condition)
{
        //code
}
```

If the condition in the if statement is not met and the condition in the else-if statement is met, the code in the else-if statement will run.

YOU CANNOT HAVE AN ELSE-IF STATEMENT WITHOUT AN IF STATEMENT!

While Loops

- There are 4 types of loops in C#: for, foreach, while, and do-while

- While Loops: repeatedly execute code within the loop until the condition specified is achieved.
  - Regular while loops test the condition specified before the code runs

- Do-While loops: run after the code has already been executed

```
//loop examples
//while loops repeadidly execute a body of code until
//the specified condition is met
int i = 0;
while(i < 3)
{
  Console.WriteLine(i++);
}


//do-while loop, logic is at the end of the loop
int j = 0;
do
{
  Console.WriteLine(j++);
} while(j<3);
```

For Loops

- For Loops: for loops will repeat a section of code until the amount of loops specified is complete.
  - For loops contain 3 clauses
    - Init clause: executes before the loop begins and initializes iteration values
    - Condition clause: a boolean expression that is tested before each loop iteration
    - Iteration clause: executed after each iteration of the body -- typically to increment or decrement the loop value
- For Each- loop: this statement iterates over each object in an indexable object (lists or strings)

```
//for loops
for (int k = 0; k < 3 ; k++)
{
  if (k  == 0)
  {
    Console.WriteLine("My name is Katie");
  }
  if ( k == 1)
  {
    Console.WriteLine("Welcome to level 3");
  }
  if (k ==2)
  {
    Console.WriteLine("I love coding");
  }
```

```
/*for each loops work very well for iterating over a
list of items */
foreach (char c in "programming")
{
  if (c != Convert.ToChar("o"))
  {
    Console.WriteLine(c + " ");
  }
}
```

Input / Output

- To get user input for your program we use the function ReadLine() and store the input in a variable like how you used the input() function in Python!
- Like Java you need to use a specific library to grab input from the console → for C# we use the “system” library

```
1   using System;
2
3   class MainClass {
4     public static void Main (string[] args) {
5       Console.WriteLine ("Tell me your password");
6
7       answer = ReadLine();
8
9       Console.WriteLine("Your password is not strong enough, make a new
        one");
10    }
11  }
```

Activities

1. Take user input and print it out 10 times

2. Take user input for the user's name and determine if the name is cool or lame (it's only cool if it starts with the same first letter as your name).

3. Print out numbers 1 through 20; however, for every number in the sequence divisible by 3, print "Bing", and for every number divisible by 4, pring "Pop". For numbers divisible by both 3 and 4, print "Bing Pop!".

- Take a user input that asks for the current temperature and it tells you whether or not you need a sweater, or no sweater depending on the weather
  - If temperature is above 70 tell them it's hot and suggest an outfit
  - If its less than 60 tell them it's cold and suggest an outfit

- Create an infinite while loop and Asks for at least 4 test scores until a person types in "q" or "quit"
  - Then compute the average of those test scores

- Create a program that takes an input from the user and then sums all numbers from 1 to the user input.
  - Modify the above program so that multiplies rather than sums
- Take user input for a password and turn all the vowels into a “*”
  - Ex: password → p*ssw*rd