GWC Lesson 10

Darling.
Luck favors
the prepared.

Source: http://www.thetreedom.com/2016/11/23/luck-favors-the-prepared/

Project from 11/ 06

- Last week we wrote a function that would return a string that replaced each vowel with the number of that vowel from a vowel counter:

“**Au**gustine **o**f Hipp**o**” —> “01g2st3n4 5f H6pp7”

- This code required a counter, type casting, and took most of the code from our first project but made small changes

Review

- Type casting (conversion): changing the data type of a variable or piece of data to another data type
  - num = str(5) integer to string
  - word = “apple”
  - str(word)
- Type Function: the type() function allows us to check what data type a variable is
  - print(type(“hi”))
  - This is used for debugging

Review

- Variables: placeholders for different data types you want to store for later use
  - Are usually lowercase
  - Don’t start with a number
  - Variable name is to the left of the = (the assignment operator)
- Comments: used in code to keep notes or to block out a portion of a program not to run
  - Achieved using the #
  - #this is my loop

Review

- Conditional Statements: If this do this, else if this do this, else do this.
  - If, elif, and else
- Lists: a type of data structure in which you can store multiple data types in one variable (a list object)
  - alist = [1,2,3,4]
  - can store any data type
  - mutable, you can change the value at any index using
    - alist[0] = “A”
  - Lists have indexes that start at 0 not 1
- Indexes: are the position of an element in a list

Review

- Strings: Immutable data types that can be looped through
  - Strings are literal text
  - Strings can be indexed like a list
  - Strings are IMMUTABLE: you cannot change what a letter or character is at some index of a string, in order to change a string you must create a new one.

Review

- <u>for loops:</u> a way to repeat code in a program where you can define the number of repetitions you want your code to perform
  - for i in range(0,10):
  - for i in range(10,0,-1):
  - for i in range(4):
  - for i in alist:
  - for i in “hello”:

Review

- The i in for loops stands for an arbitrary changing constant
  - Everytime your loop executes, the i value is changing based on the loop or index you are on

Consider the following:

-what is the range

-what is i at each loop

Your Code ...

```
1
2  aword = "marquette"
3  count_e = 0
4  for i in range(len(aword)):
5      if aword[i] in "e":
6          count_e += 1
7
8  print(count_e)
```

Review

- <u>While Loops:</u> a loop that will keep executing as long as the condition is not met
  - while (x <10) :
  - while (False):
  - These loops could potentially get stuck in an infinite loop if the while condition is never met
  - These loops are good to use when you don't know how many times you want your code to repeat
  - Don't make use of an arbitrary constant

Review

- Input Function: the input function allows you to get written input from the user
  - x = input(“Tell me your name”) (what data type is x)
  - a = int(input(“give me your age”)) (what is happening to the input here)
- User input is often stored in a variable to then be used later in code, which is why there is a variable on the left hand side of the code above

Functions

- Functions: chunks of code that you can reuse, and is assigned to a specific task in your program
  - Some functions are built in like range(), len(), type() input(), int(), str()
- Void functions: these functions do not return a value, they simply execute the code, they usually have print statements and only execute once when called
- Return Type Functions: these functions have return statements that will give you a value or a list of values, often these types of functions are used with other functions.

Review

```
def  jump():                -void function
        Code goes here
jump()

def name(astring):    -void function
        code goes here
name(“katie”)
```

Review

-return-type function:

```
def add(a,b):
    sum = a+b
    return sum
print(add(4,5))
```

Practice

- We can make a program that keeps asking a user for input
  - Use a while loop
- Try to finish the following code
  - You need to make it so the code will stop asking for user input

```
def repeat():

    while True:

        x = input("What day of the week is it?")

        if x == "Monday":
            print("It's the first day of the week!")

        elif x == "Saturday":
            print("Its the weekend, yay!")
           #put a statement here to stop the program


#function call here
```

Dice Roller Simulation

- For this you will need to use the python library called Random: import random
  - Use random.**randint**(*a*, *b*) where a = start and b = end value
- You will need to ask a user if they want to roll the dice, yes for go, no for quit the program.
- Use a while loop to keep rolling dice unless the user input equals “no”
- The random generator will generate a number between 1 and 6 and then print the number to the screen as “you rolled a “ + (number rolled)