CS 112 Practice Problems 2

True or False: A break statement causes us to immediately exit all loops we're currently inside of

False – it breaks the innermost loop we are inside

True or False: It is impossible for a while loop to run zero times

False – loop guard could evaluate to false at iteration 0

What does the following code print?

```
counter = 0
while counter < 6:
        print("?")
        counter+=1
print(counter)
```

Prints out:

?
?
?
?
?
6

What does the following code print?

```
xs = [2,4,6,8]
total = 0
for i in range(len(xs)):
        total += i
print(total)
```

range(len(xs)) = [0,1,2,3]
total = 0 + 1 + 2 + 3 => 6
prints out 6

Implement the count_evens() function. Count_evens() returns the number of evens in the list of integers.

Examples:

```
count_evens([])          ->      0
count_evens([2,4,6,8])   ->      4
count_evens([1,2,3])     ->      1
```

```
def count_evens(x):
        counter = 0
        for val in x:
                if x % 2 == 0:
                        counter++1
        return counter
```

How many times is "Hello" printed for the following code?

```
for I in range(5):
        for j in range(10):
                print("Hello")
                break
```

As soon as inner loop prints "Hello", it breaks to the outermost loop

a) 50 times
b) 1 time
c) 5 times
d) 4 times

Implement the max_location() function. It returns that index of the largest value in the list of integers. If the list is empty, return None. If the largest integer occurs multiple times, return the first occurrence's index.

Examples:

```
max_location([])                -> None
max_location([4,10,3,10,6])     -> 1
max_location([-5,-3,-1,-14])    -> 2
```

```
def max_location(xs):
    if len(xs) == 0:
        return None
    max = xs[0]
    index = 0
    for x in xs[1:]:
        if x > max:
            max = x
            index = xs.index(x)
    return index
```

What is printed by the following code?

```
width = 3
height = 2
for row in range(height):
    for column in range(width):
        print(row, column)
```

Prints out:

0 0
0 1
0 2
1 0
1 1
1 2

What is printed by the following code? Execution starts in the main() function.

```
def x(y):
    for i in range(len(y)):
        y[i] = -1

def main():
    xs = [5,3,7,2,3]
    x(xs)
    print(xs)
```

Lists are passed by reference, so original list values are changed

Prints out: [-1, -1, -1, -1, -1]

What is printed by the following code?

```
x = 0
for i in range(7):
  if i%2 == 0:
     continue
  x += I
print(x)
```

x = 1+3+5

Prints out: 9

Implement the two functions chartodecimal() and hextodecimal(). chartodecimal() has a single parameter hex_char that represents a hexadecimal character in string format. Chartodecimal() returns the base 10 representation of a hexadecimal string. Hextodecimal() returns the base 10 representation of a hexadecimal string. It has a single parameter hex_string. To complete the chartodecimal() function, use the following information:

The base 10 equivalent of hexadecimal characters are as follows:

0=0, 1=1, 2=2,, 9=9, A=10, B=11, C=12, D=13, E=14, F=15

The ord(c) function returns the decimal representation of the ascii character. c is the character to convert.

The decimal representations of the ascii characters are as follows:

Character	Decimal
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57
A	65
B	66
C	67
D	68
E	79
F	70

To complete the hextostring() function, use the following information:

The general formula to convert from base 16 to base 10 is as follows

$$\sum_{i=0}^{n} x * 16^{i}$$

Where x is the decimal representation of a hexadecimal character, I is the index, and n is the number of hexadecimal characters. Note that index 0 indicates the least significant hexadecimal character (rightmost character in the string)

** is the exponential operator

[::-1] is used to reverse a list

Examples:

hextodecimal("E7A9") -> 59305 $(14 \times 16^3) + (7 \times 16^2) + (10 \times 16^1) + (9 \times 16^0)$

hextodecimal("FFFF") -> 65535 5 $(15 \times 16^3) + (15 \times 16^2) + (15 \times 16^1) + (15 \times 16^0)$

hextodecimal("A") -> 10 (10×16^0)

Hint -Call chartodecimal() in a loop in hextodecimal()

```
def chartodecimall(character):
        val = ord(character)
        if val <= 57:
                return val-48
        return val-55


def hextodecimal(hex_string):
        index = 0
        result = 0
        for val in hex_string[::-1]:
                result+= chartodecimal(val) * (16 ** index)
                index+=1
        return result
```

Implement the function binarytodecimal(). binarytodecimal() accepts a string in binary format and returns its base 10 (decimal) equivalent). To complete the binarytodecimall() function, use the following information:

** is the exponential operator

The general formula to convert base 2 to base 10 is as follows:

$$\sum_{i=0}^{n} x * 2^{i}$$

Where x indicates if the character is 0 or 1, i indicates the index, and n represents the number of bits (characters in the string). Note that index 0 indicates the least significant bit (rightmost character in the string).

[::-1] is used to reverse a list

Examples:

binarytodecimal("0") -> 0 (0 * 2^0)
binarytodecimal("1111") -> 15 $(1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)$
binarytodecimal("11011") -> 27 $(1 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)$

```
def binarytodecimal(binary):

    index = 0
    result = 0
    for val in binary[::-1]:
        if val == "1":
            result+= 2 ** index
        index+=1
    return result
```