CS 112 Midterm Practice Questions

True or False: Printing a value and returning a value from a function have the same effect

True or False: A function call always returns a value

What is printed by the following code?? Answer to the right

```
def something(a, b):
        b[a] = 99
x = 2
y = [5,10,15,20]
z = something(x,y)
print(x,y,z)
```

True or False: the following code can execute successfully:
```
Stuff = ("abc", [1,2,3], 45, (6,7,8))
Stuff[1][2] = 9
```

Given the definition xs = [(3,40), True, 5, "hello"] what is the type of expression xs[0:1]

Which of the following 6 things are legal identifiers?

Num_gallons   1st_place

Last place   x1

Base&tax   i

Define the function is_palindrome(x). x is a string. Assume x has no spaces. A string is a palindromes of each itself if and only it reads the same backward as forward.

Examples:
```
Is_palindrome("racecar")            ->      True
Is_palindrome("tacocat")            ->      True
Is_palindrome("Billy")              ->      False
```

Match each term with its definition

A. Assign
B. Aliases
C. Update
D. Reassign

Modify what this name refers to, changing its id()

Modify part of this value, leave its id() unchanged

Attach a name to a memory location to keep a value

Multiple different names referring to the same spot in memory

Implement the function validgnumber that determines if a GMU Student ID is valid. A valid GMU Student ID is defined as follows:

exactly 9 characters long
begins with the uppercase character 'G'
all characters beside the beginning 'G' character must be numbers

The function accepts a parameter gnumber and returns True if the gnumber is valid otherwise False. The x.isdigit() function can be used to determine is a character is a number where x is an object
Examples:
validgnumber("0132")   ->   False
validgnumber("934523638")   ->   False
validgnumber("G12345678")   ->   True

def validgnumber(gnumber):

Trace the output of the following program. Then rewrite the program using a "while" loop instead of a "for" loop.

```
colors = ['Red', 'Green', 'Blue', 'Yellow', 'Pink']

for i in range(0,len(colors), 2):
    print (i, colors[i])
```

Which one of the following options is not a valid string literal?

a) 'Employees must "wash" hands before leaving restroom'
b) 'some escapes: \n\t\\"\''
c) '''\t\\t\\\t\\\\t'''
d) """he puts pumpkin spice in "everything…""""
e) '''where's the party?'''

What one of the following does not generate list [0,15,30,45,60]?

a) List(range(0,60,15))
b) List(range(0,65,15))
c) List(range(0,70,15))
d) List(range(0,75,15))

True or False: When two functions use the same variable name for a parameter, they are aliases

Which of the following signatures is not allowed?

a. Def something(a, b, c, d):
b. Def something(a, b, c=3, d=4):
c. Def something(a, b=2, c, d=4):
d. Def something(a=1, b=2, c=3, d=4):

True or False: We must have a default value for a parameter in order to supply the parameters argument by keyword

How many arguments are provided to print()?

print([1,2],max(3,4),foo(5,bar(6.bazz(7))))

When running the following statement, which function gets called first?

ans = foo(1, bar(2))

Given the definition of func, which call to it will return the largest value?

```
def func(a, b=5, c=10):

        Return a+b*c
```

a. func(30)
b. func(5,10)
c. func(45,1,5)
d. func(5,c=10)
e. func(a=100,c=-4,b=10)

Given the function definition for func, what is printed by each call?

func(3,5)   def func(first=2, second='#'):

func(second='+')   print(first*second)

func()

func(first=3, second='+')

Given the definition of change(), call it by only providing two arguments (any integers are fine) for dimes and pennies

>   Def change(quarters=0,dimes=0,nickels=0,pennies=0):

>   Return quarters*25 + dimes *10 + nickels*5 + pennies*1

Define the function build_coords(numrows, numcols). It creates a list of lists of integer pairs representing the coordinates. Note that it is zero-indexed so for example the last item in the first row is (0, numcols-1), and the last item of the last row is (numrows-1, numcols-1).

Examples:
        build_coords(1,1)   ->   [[(0,0]]
        build_coords(1,5)   ->   [[(0,0), (0,1), (0,2), (0,3), (0,4)]]
        build_coords(3,2)   ->   [[(0,0) (0,1)], [(1,0),(1,1)], [(2,0), (2,1)]]