CMSC 201 Fall 2017 Name_____

Midterm Review Worksheet

This worksheet is ***NOT*** guaranteed to cover every topic you might see on the exam. It is provided to you as a courtesy, as additional practice problems to help you study. You should also be reviewing the course notes and assignments as part of your preparation for the exam.
No answers will be provided for the questions on this worksheet. You are encouraged to work with other students in the class to confirm your answers and solidify your understanding of the material. Some of the questions on this worksheet are more difficult or tricky than ones you would see on an exam - you have much more time, as well as TA assistance available, when working on the review.

1. Name the three **control structures** used to control the flow of a Python program. What is the fourth structure that we use as a control structure?
2. True or False? If false, state why (or provide a counterexample).

(a) _____ To begin an interactive Python 3 session on GL, all you have to do is type **`python`** in to the terminal window.
(b) _____ The emacs program allows the programmer to create and edit Python programs as files. A Python file can then be given to the Python interpreter to execute the program.
(c) _____ In Python, variables can be changed simply by assigning new values to them.
(d) _____ In Python, the **`def`** keyword is used to define and redefine new variables.
(e) _____ A program that prints 3 lines of output ***must*** contain 3 **`print`** statements.
(f) _____ String and list indexing always starts at 0.

3. Write a simple Python program that (a) assigns a list of the prime numbers less than 10 to a variable called **`myList`**, (b) prints the length of this list, (c) prints each element of **`myList`** by using a **`while`** loop, and (d) prints each element of **`myList`** in reverse using a **`while`** loop. *(You can hard-code step (a) by creating* **`myList`** *and initializing it with the numbers 2, 3, 5, and 7.)*

4. Write Boolean expressions for each of the questions below. The first one has been done for you. **You do not need to include the "`if`" keyword in your answer.**
   a. An expression that evaluates to **`True`** only if the integer **`number`** is odd and not 19.
      ```
      number % 2 == 1 and number != 19
      ```
   b. An expression that evaluates to **`True`** only if the integer **`num`** is positive (greater than zero) and the string **`animal`** is equal to "dog".
   c. An expression that evaluates to **`True`** only if the boolean **`bool1`** is False, the boolean **`bool2`** is True, and the boolean **`bool3`** is not True.
   d. An expression that evaluates to **`True`** only if the integer **`a`** is greater than the integer **`b`**, the integer **`b`** is less than 7, the integer **`a`** is not equal to 5, and the integer **`c`** is positive.
   e. An expression that evaluates to **`True`** only if the integer **`number`** is not odd and the integer **`number`** is between 11 and 99, inclusive.
   f. An expression that evaluates to **`True`** only if the string **`name`** is lowercase.

g. An expression that evaluates to **`True`** only if the string **`name`** is less than 12 characters, and starts with a "C" or a "2".

5. Answer the questions below, given the following strings. (We've made the strings big enough so that you can write the indexes of the letters to help you count. It will be like this on the exam as well.)

```
daya      = "I don't wanna sit still look pretty"

glimpse   = "life wasn't easy, but living it was"

oliver    = "every boulevard is a miracle mile"

callMeAl  = "I can be your long-lost pal"

perry     = "baby you're a firework"
```

What do the following pieces of code evaluate to?	How would you get each of the following substrings from the indicated string?

`print(perry[ len(perry)-8 : ])`
`print(callMeAl[6:6+7])`
 	
`print(daya[24:28] + perry[4:12] +`
`  daya[ len(daya)-6 : ])`
 	
`text = perry[:4]`
`print(text + text + text)`
 	
`print(glimpse[6], oliver[6],`
`  callMeAl[2])`
 	
`var  = (len(perry) - 1) // 10`
`gLen = len(glimpse)`
`print(perry[5:8],`
`  callMeAl[var:var+3],`
`  daya[2:4],`
`  glimpse[gLen-6 : gLen-4])`
	"wanna" from daya
	
	"lost" from callMe
	
	"fire" from perry
	
	both "was" from glimpse
	
	"every" from oliver
	
	the apostrophes in daya, glimpse, and perry
	
	the hyphen in callMeAl
	
	"miracle mile" in all caps from oliver

6. Give three examples of legal **variable names** and three <u>different</u> examples of illegal variable names in Python. For the illegal variable names explain why they are illegal. (Do not use one-letter variable names.)

7. Complete the code by **filling in the blanks** for each question below. (Length of blank doesn't matter.)

   a. Print out the contents of the list `cars`.

```
cars = ["honda","ford","porsche","tesla"]
indx = 0
while _____ _____ _____:
    print(_____)
    _____
```

   b. Read in 10 integers from the user.

```
numberList = []
while _____  ____ 10:
    userNum = _____(_____("Enter a num: "))
    numberList._____(_____)
```

   c. Add the numbers 5 through 10 together.

```
total = _____
count = _____
while _____  ____ _____:
    total = _____ + _____
    count = _____
print("The total of 5 through 10 is", _____)
```

8. For each of the statements below, circle and **explain any errors** you find. (There may be more than one in a single statement! A statement may also be error-free.) You can assume that variables are initialized and contain what their names indicate (*e.g.*, `intCounter` is an integer, etc.), and that `main()` is called at the end.

   a.
```
def diff(num1, num2)
    ans = num1 - num2
    return ans

def main():
    diff(51, 23)
    print(ans)
```

   b.
```
main():
    course == "CMSC 201'
```

   c.
```
def printStatement(num):
    print("num is" + num)
```

```
def main():
    printStatement(5) * 3
```

9. Write a snippet of Python code that reads an **integer** from the user, and then computes that integer's **absolute value**. The absolute value of a number x is defined as

$$|x| = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{otherwise} \end{cases}$$

10. Write a snippet of Python code that continuously takes input from the user using a **`while`** loop, and adds that input to the end of a **list**. When they enter "**`quit`**" the program should print the list and terminate.

11. What is the **output** of each small Python program below?

(a)
```
x = 3
y = 4
print (x)
x = 4
```

(b)
```
x = 3
while x < 6:
    x = x + 1
print (x)
```

(c)
```
x = 6
y = 5
if x >= y:
    x = x - 2
print (x)
```

(d)
```
tc1 = 100
tf  = (9/5) * tc1 + 32
tc2 = (tf - 32) * 5/9
print (tf)
print(tc2)
```

(e)
```
x = 0
while x < 5:
    if x % 2 == 0:
        print("even: ", x)
    else:
        print("odd:  ", x)
    x = x + 1
```

(f)
```
x = 1
i = 1
while x <= 4:
    x = x * i
    i = i + 1
    print (x)
```

12. In each of the following three questions there is an error in the Python code. **Identify the error** by name and describe the problem. (Each piece of code is prefixed with a brief description of the programmer's intention.)
    a. The following code attempts to compute the equation **`m * x + b`** and assign that value to **`result`**.

```
result = mx + b
```

    b. The following code attempts to compute the first-order effects of some physical process. (You may assume the equation is correct.)

```
cofactor = alpha * x * x
1storder = 1.0 / cofactor
```

c. This code increases **x** every iteration until it is greater than **y**.

```
x = 5
y = 5
while x <= y:
    x = x + 1
    y = y + 1
print ("Done!")
```

13. Write a program that outputs every other word in this list: [candy, bar, candy, corn, candy, canes]

14. Write a program that allows the user to input 5 numbers and then uses a function called **addAll()** to add them all together and output the sum. (It is your decision if the numbers are passed in as a single list, or as five separate variables.)

15. Answer the following questions about **data types in Python**.
    a. What built-in function do you use to convert an integer into a floating-point number? Demonstrate this function by converting the number **3** into a floating-point number and assigning it to a variable called **flt3**.
    b. What built-in function do you use to convert a string into an integer?
    c. What happens when you try to convert a string into an integer but the string is not a number?
    d. When would one want to convert an integer into a string? In other words, in what cases would a programmer need to do such a conversion? Demonstrate with an example in Python.
    e. What is the most appropriate data type to represent real (decimal) numbers in Python?

16. What is the value of **mystery** after this sequence of statements?

```
mystery = 1
mystery = 1 - 2 * mystery
mystery = mystery + 1
```

17. What is the value of **mystery** after this sequence of statements?

```
mystery = 3
mystery = mystery + 1
mystery += 3 * 5 - 1
```

18. What is the value of **mystery** after this sequence of statements?

```
mystery = 5
mystery = (15 - mystery) % 2
mystery += 1
```

19. Write four **while** loops, each of which performs one of the following actions.
    a. Iterate over a list
    b. Input validation from user
    c. Count a specific number of times
    d. Loop until a specific condition is met

20. Circle and correct at least five errors and identify what kind of error they are (logical or syntax):

```
def outPrint(strName, age, gpa):
    print("The student's name is:", name)
    print("Their age is: " + int(age))
    print("Their gpa is: " + float(age))

def main():
    name       = input("Enter the student name: ")
    int(age)   = input("Enter the student's age: ")
    float(gpa) = input("Enter the student's gpa: ")
    outPrint(name, gpa, age)
main()
```

21. What is the output of this code?

```
def main():
    strStatement = "End of the world as we know it"
    print(strStatement[:])
    print(strStatement[4:6])
    print(strStatement[11:16])
    print(strStatement[(len(strStatement)-7):len(strStatement)])
main()
```

22. Find and correct at least eight errors in the code below. They may be syntax or logic errors. (FYI: This code is very broken, and requires a lot of "simple" fixes to be runnable.)

```
def main():

     BLACKJACK = 21
     1st_hand = ["13", "9"]

     cardSum = 0
     index   = BLACKJACK
     while index < cardSum:
          cardSum = cardSum + 1st_hand[index]

     if cardSum >= BLACKJACK:
          print("You lost!")
     else if cardSum < BLACKJACK:
          print("You can hit, or check.")
     else:
          print("You beat the dealer!")

     1st_hand.removeAll()
```

The rules of blackjack are that you want to get as close to a total of 21, without going over. Cards are dealt randomly, so although there is some strategy, it is largely a game of chance.

23. Find and correct at least eight errors in the code below. They may be syntax or logic errors.

```
main():

    # hours worked per weekday (Mon-Fri)
    hours = [6.8, 5.7, 4.9, 8.0, 3.2]

    # calculate total
    total = 1
    index = 0
    while index < len(total)
        total = total + hours[h]
        index = 1

    # calculate average
    avg = total % range(hours)

    # print total and average
    print("This worker worked", total, "hours this week.)
    print("For an average of", total, "hours per day.")
main()
```

24. Find and correct at least six errors in the code below. They may be syntax or logic errors.

```
# digits and their English equivalent
DIGITS = <"zero", "one", "two",   "three", "four",
          "five", "six", "seven", "eight", "nine">

def main():
    num = int(input("Please enter a positive integer: "))

    # input validation for only accepting positive numbers
    while (num > 0):
        print("The number", num, "is not a valid choice.")
        num = int(input("Enter a number 1 or higher: "))


    # create variables to use below
    num = copy
    string = ""


    # go through and convert each digit to English
    while copy > 0:
        string = string + " " + DIGITS(copy % 10)
        copy = copy / 10


    print("The number", num, "is", string)
```

25. Fix the code below so that it works correctly.

```
candyPrice = 1.25
myMoney = int("twenty")
moneyLeft = myMoney - candyPrice
print("I have $" + moneyLeft + " money after buying candy.")
```

26. What is the output of the code below?

```
wishList = ["PlayStation 4", "Puppy", "Chocolate"]
wishList.append("Puppy")
wishList[0] = "XBox One"
wishList[0:3]
print(wishList)
```

27. Define each of the following terms.
(This is meant to help test your **<u>understanding</u>** of the terms, not whether you can recall the "correct" definition from the slides or book.)

1. Algorithm
2. Argument
3. Boolean
4. Boolean Flag
5. Bracket (*e.g.*, square brackets)
6. Branching
7. Bug
8. Call (e.g., function)
9. Case Sensitive
10. Casting
11. Class
12. Code
13. Comment
14. Comparison
15. Concatenation
16. Conditional
17. Constant
18. Debugging
19. Decisions
20. Definition (e.g., function)
21. Delimiter
22. "Equivalent to"
23. Error (e.g., logic error)
24. Float
25. Formal parameter
26. Function
27. Incremental Development
28. Index
29. Infinite Loop
30. Initialize
31. Input Validation
32. Integer
33. Integer Division
34. Interactive Loop
35. Interpreter
36. Iterate
37. Keyword
38. List
39. Literal
40. Logic Error
41. Logical/Boolean Operators
42. Main
43. Membership Operator
44. Method
45. Modulus (or Modulo/Mod)
46. Nested (*e.g.*, loops)
47. Operator (*e.g.*, assignment)
48. Output
49. Program
50. Pseudocode
51. Return
52. Scope
53. Selection
54. Sentinel Loop
55. Sequential
56. Slicing
57. String
58. Syntax
59. Syntax Error
60. Value
61. Variable
62. Whitespace