CMSC 201 Section 60
Fall 2022
Sample Final Exam

Notes:

1 This test is open book/open notes. You may NOT get help from any outside source; e.g., no talking to classmates and no communicating with people outside the classroom
2 You have two hours to complete this exam from when you are permitted to start
3 This test is worth 150 points

Section 1: True/False and Multiple Choice

There will be 15 questions, worth 3 points each, for a total of 45 points. No partial credit will be given for problems in this section.

1. Which of the following variable names is NOT a valid Python variable name?

    a. 2CoolforSchool

    b. 2_Cool_for_School

    c. Too-Cool-for-School

    X d. None of them is a valid variable name

2. Which loop would you use if you wanted to get input from the user and stop when the user entered a specific value?

    a. for I loop

    b. for each loop

    X c. Sentinel while loop

    d. Any of the above loops would work

3. True or False: all input read from files enters as strings, while any immutable type – int, float, Boolean, or string – can be directly read from the keyboard

a. True

X b. False

4. The best way to write a program is:

X a. Write a little, test a little

b. Copy it from some site on the Internet

5. True or false: the only place in a Python program you CANNOT call a function is on the left side of an assignment statement – the left side of the equals sign.

X a. True

b. False

6. What is the Python symbol table?

a. A list of the values of all variables known to the program at that time

X b. A data structure that lists the symbols – variables, constants and functions – known to the program at that time, along with their type and the location in memory where the value can be found

c. A 2D list in Python

d. A 3D list in Python

7. Suppose that l is a 2D list. True or False: len(l[1]) tells how many rows are in the list?

a. True

X b. False

8. Why won't the following recursive program work?

```
def fact(num):
     if num == 1:
          return 1
     else:
          return fact(num)
if __name__ == "__main__":
     n = 6
     result = fact(n)
     print (result)
```

a. There is no base case

b. There is no recursive case

X c. The recursive case does not result is an simpler problem; that is a problem closer to the base case

d. What are you talking about? This program works fine.

9. Suppose I have a list consisting of the names of the states in the US, like this:

```
states = ['Idaho', 'Utah', 'Hawaii', 'Maine', 'New York',
'South Carolina', 'Illinois', 'Pennsylvania', 'North Carolina',
'Colorado', 'California', 'Alaska', 'Missouri', 'Kansas',
'Oklahoma', 'Connecticut', 'South Dakota', 'North Dakota',
'Louisiana', 'Nevada', 'Delaware', 'Washington', 'Wisconsin',
'Georgia', 'Nebraska', 'Virginia', 'Wyoming', 'New Hampshire',
'Texas', 'Kentucky', 'West Virginia', 'Rhode Island',
'Maryland', 'Massachusetts', 'Vermont', 'New Mexico',
'Florida', 'Tennessee', 'Iowa', 'Arizona', 'Montana',
'Minnesota', 'Alabama', 'Mississippi', 'Arkansas', 'Oregon',
'New Jersey', 'Ohio', 'Michigan', 'Indiana']
```

How would I create a new list consisting of the first five states in the list, "Idaho" through "New York"?

a. `new_list = states[:]`

b. `new_list = states{:6]`

c. `new_list = states[1:5]`

X d. `new_list = states{:5]`

10. What's the difference between readline() and readlines() when you are reading a text file in Python?

a. readline() reads one line of the file, which readlines() reads the entire file as a single string

b. There is no difference; Python does the same for both

c. readline() reads the entire file in as a single string, while readlines() reads the file into a list of strings, one list element per line

X d. readline() reads the next line of the file as a string, while readlines() reads the file into a list of strings, one list element per line.

11. True or False: it is possible to "nest" functions in Python; that is, declare one function to be wholly contained within another function.

X a. True

b. False

12. Suppose that you have the string `s = "University"` What statement would give you the same string, but without the last "y"?

```
a. new_s = s[:]

b. new_s = s[:1]

c. new_s = s

X d. new_s = s{:-1]
```

13. True or False: d, below, is a legal dictionary in Python

```
d = {
  "UMBC": "America East",
  "UMCP": "Big Ten",
  "Towson":   "CAA",
  "Stony Brook": "CAA",
  "UMCP": "ACC"
}
```

a. True

X b. False

14. Suppose that you have the list

```
l = [1,2,3,4,5]
```

You want to change l to contain the square of each number. That is, you want l to be [1,4,9,16,25]. True or False: you can use either a "for each" loop or a "for I" loop to accomplish this?

a. True

X b. False

15. True or false: big Theta of an algorithm exists only if its best-case performance and its worst-case performance are the same.

X a. True

b. False

Section 2: Short Answer

There will be 12 questions, worth 5 points each, for a total of 60 points. All questions should be answered in no more than three sentences, or in a few lines of code. Partial credit WILL be awarded for answers that show an understanding of the material, even if those answers are not completely correct.

16. Suppose I have a list l = [1,2,3,4,5]. len(l) returns the value 5. What happens if I try to print l[5]?

You get an index out of range error because the last element is always len(l) - 1.

17. Suppose I have the following Python statement, designed to get the user to enter a student's name and test score.

```
data = input("Please enter the student's name and test score. Then hit "enter" when done")
```

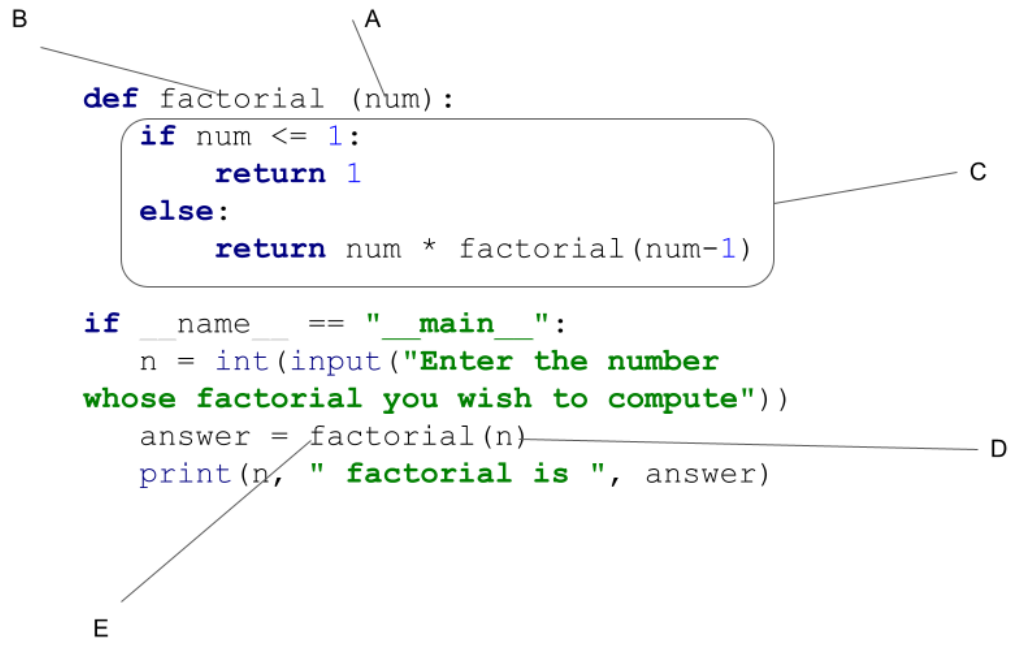
Write code that will split the user's input into a string containing the student's name, and an integer containing the student's test score.

```
datalist = data.split()

datalist[1] = int(datalist[1])
```

18. Suppose I have the following Python program. Explain why this would crash with the error "print_result undefined"

19. . Identify each labeled part of the program below:



20. Why can't you have both a function, `length`, accessible from the main program and a variable, `length`, declared in the main program?

Because the symbol table couldn't tell them apart

21.. Suppose that `students` is a 2D list with 5 rows and 4 columns. If I try to print `students[1][3]`, which subscript refers to the row number and which subscript refers to the column number?

First subscript is always the row number; second is always the column number

22. A Python dictionary has keys and values. What types can be legal values in a Python dictionary?

Any type

23. Explain what the “sentinel” is in a sentinel while loop.

The value you’re looking for to stop executing the loop

24. What’s the difference between a “local variable” and a “global variable” in a Python program?

Local variable: defined in a function; only accessible in that function

Global variable: defined in the main program; accessible everywhere in the program; don’t use because they’re known to cause program bugs that are impossible to fix

24. What happens if you try to open a file in append mode (“a” mode) in Python and the file does not exist?

Python creates the file for you

26. Write a recursive function that takes one parameter, a string s, and finds if s is a palindrome. Your function should return True if s is 0 or 1 characters long; it should return False if the first and last letters of s are different. Otherwise it should make a recursive call with a substring of s that drops the first and last letters from s.

```
def palindrome(word):
      if len(word) <= 1:
            return True
      elif word[0] != word[-1]:
            return False
      else:
            return palindrome(word[1:-1])
```

You do not need to write the whole program; just the recursive function.

27. What does the Big O value of an algorithm such as a sorting algorithm or a searching algorithm tell you?

Section 3: Programming

There will be three questions, worth 15 points each. Partial credit WILL be awarded for answers that show a significant understanding of the Python coding needed to solve the problems, even if those answers are not completely correct.

28. Write a Python program that takes the first 100 integers – start at 0; end at 99 – and puts them in a 2D list with 10 rows and 10 columns.

```
list_of_ints = []
```

```
for i in range(10):  # 10 rows
    new_row = []
    for j in range (10): # 10 columns
        new_row.append(i*10 + j)
    list_of_ints.append(new_row)
```

29. . Suppose I Have a file, grades.csv, that contains student grade records. The file looks like this:

```
Washington,George,85,82,93,81

Adams,John,88,77,66,55

Jefferson,Thomas,92,16,44,0
```

That is, each line contains the information for one student. There are six fields on each line, and those fields are separated by commas. The fields are: last_name, first_name, grade1, grade2, grade3, grade4.

Write a Python program that reads in this file, splits out the fields, adds the four grades to get a total, and prints out to the screen the last name and total grade. Your output should look like:

```
Washington: 341

Adams:286
```

```
with open("grades.csv","r") as infile:
    data = infile.readlines()
    for i in range(len(data)):
```

```
        data[i] = data[i].split(',')

        total_score = int(data[i][2]) +
int(data[i][3]) + int(data[i][4] +
int(data[i][5])

        print (data[i][0], ":", total_score)
```

30. Write a recursive Python function that computes a base raised to the exponent power. The function will have two parameters: `base`, which is the number to be raised; and `exponent`, the power to raise the base to. In other words, if we wanted to calculate 2 to the 4th power, we would call this function with 2 as the base and 4 as the exponent. If we wanted to calculate 3 squared, we would call this function with 3 as the base and 2 as the exponent.

If the exponent is 0, your function should just return 1 because any integer to the 0 power is 1. If the exponent is 1, return the base because any number to the first power is that number. If the exponent is greater than 1, return the base times what the recursive call to the function returns.

You may presume that `exponent` will never be negative.

You do not need to write the entire program; just the recursive function.

```
def exponentiate(base, exponent):

   if exponent == 0:

       return 1

   elif exponent == 1:
```

```
        return base
    else:
        return base * exponentiate(base,
exponent-1)
```