CSE 101: Introduction to Programming (IP)- 2024

(Solution Rubrics)

Mid-Semester Exam

03-October-2024

Duration: 45 mts **Total: 20 Marks**

Name: **Roll No:**

Your Section (A/B): **Group in the section (1-18):**

Note:

- *Marks for each question are given.*
- *Write your answer in the space provided. You can write in pencil (it will help you in changing). No resources of any kind are permitted.*
- *This is an exam - plagiarism policy for exam applies.*

---

1. Briefly describe how a python program is executed - give two main steps in it. **[2 marks]**

Answer:

1. Python compiler checks the syntax and compiles the program into bytecode (1 mark)
2. Python interpreter then executes each of the bytecode instructions (1 mark)

2. What does the following program do ? **[1 mark]**

```
1.    numbers = [6,4,9,1,5,3,2]
2.    c = 0
3.    x = int(input())
4.    for i in range(len(numbers)):
5.          if (x==numbers[i]):
6.                print (i)
7.                c = 1
8.                break
9.    if(c==0):
10.         print("Not found")
```

Answer:
The program performs a (linear) search for a given number in the list. Prints the **index** value if the number is found, else prints “Not found”.

3. **[1+1 marks]**

a. Give a suitable comment (a simple sentence) to describe what this program is computing.
b. What is the output of the program for input n = 30 ?

```
n = int(input("Enter a number: "))
p = 1
while(p <= n):
    k = 0
    if(n % p == 0):
        j = 1
        while(j <= p):
            if(p % j == 0):
                k = k+1
            j = j+1
        if(k == 2):
          print(p)
    p += 1
```

a.	**b.**
The program prints the prime factors of a given number n. (1 mark)	Output: 2, 3, 5 (1 mark)

4. Given two lists L1 and L2 of numbers of equal length, write a list comprehension statement to create a list of items that are in both L1 and L2 and at the same index.
[2 marks]

Answer:

[l1[i] for i in range(len(l1)) if l1[i]==l2[i]] (2 marks)

(Max 1.5 marks if the answer is logically correct, but not written as a single list comprehension statement)

(1 mark for the following as this does not give the expected result if the common element is duplicated in one of the lists. Eg: If L1 = [1,2,3,1] and L2=[4,2,3,1] the output will be[2,3] and not [2,3,1])

[e for e in L1 if e in L2 and L1.index(e) == L2.index(e)]
(Checking if e is in L2 in the if condition is essential, else no marks)

5. Given two lists of numbers L1 and L2 - assume L1 is smaller in length than L2. Write one statement using list comprehension that will return a list whose items are the sum of corresponding numbers in L1 and L2, and for extra numbers in L2, they are the numbers in L2, i.e., if L1 = [1, 2, 3] and L2 = [4, 5, 6, 7, 8], the expression should return a list [5, 7, 9, 7, 8]. **[3 marks]**

Answer:
[L1[i]+L2[i] if i<len(L1) else L2[i] for i in range(len(L2))]
or
[L1[i]+L2[i] for i in range(len(L1))] + [L2[i] for i in range(len(L1),len(L2))]

(Max 2.5 marks if the answer is logically correct, but not written as a single statement as asked for)

6. Given a string s which is a sentence without any punctuation. Write one statement to get a string sr which has words of s in reverse order. **[2marks]**

Answer:
sr = " ".join(s.split()[::-1])
(Words of s in reverse order - Eg: If s = "Hi Good Morning", sr = "Morning Good Hi")
(0.5 marks if the answer is sr=s[::-1], as this will reverse the string letterwise but the question is about wordwise reversing.)

7. What will be the output of this program: **[2 marks]**

```
L1 = [5,9]
L2 = [6,8]
L = L1*2 + L2
L.pop()
del(L[2])
L.insert(L.index(6), 1)
print(L)
L2.insert(len(L1),3)
L2.pop(len(L2)-1)
print(L2)
```

Answer:
Answer:
[5, 9, 9, 1, 6] (1 mark)
[6, 8] (1 mark)

8. What will be the output of the following program, if the inputs are: 15 6 9 25 5 1 (with a return after every input). **[1 mark]**

```
sum3, sum5 = 0, 0
while True:
    x = int(input())
    if x%3 == 0:
        sum3 += x
    elif x%5 == 0:
```

```
            sum5 += x
        else:
            break
print(sum3, sum5)
```

Answer:
30 30 (1 mark)

Explanation: Sum3 is the cumulative sum of multiples of 3. Sum5 is the cumulative sum of multiples of 5. The number 15 is added to sum3 as the program checks whether the number is a multiple of 3 and if not, it checks whether it's a multiple of 5.

9. **[1+1 marks]**

a. Explain in one short statement what this function computes.
b. What will be the output if an input of 3 2 is given when the program is run.

```
def f(n, m):
    result = 1
    for _ in range(m):
        result *= n
    return result

def g(n, m):
    total = 0
    for i in range(m + 1):
        total += f(n, i)
    return total

# Main program
n, m = 3, 2
print(g(n, m))
```

a.	**b.**
It computes n^m+n^m−1+⋯+n^0 (1 mark)	Output is 13 (1 mark)

10. The function below checks if the string has proper matching parenthesis, e.g. (()) is matching (function returns True), while ((())) is not (function returns False). The function has a bug. **[1+1+1 marks]**

a. Write an assertion statement which will pass (that is the function computes correctly and so the bug is not revealed)
b. Write an assertion statement which will fail (and reveal the bug).

c. Suggest minimal modifications to the program to fix the bug (if you want to insert some code, mention the line number after which you will insert).

```
1. def is_valid_parentheses(s):
2.     open_count = 0
3.     for char in s:
4.         if char == '(':
5.             open_count += 1
6.         elif char == ')':
7.             open_count -= 1
8.     return open_count == 0
```

(i). Assertion which passes:

s = "(()())"

assert(is_valid_parenthesis(s)) # Program works - no error

(1 mark)

(ii) Assertion which shows the bug:

s2 = "))(("

assert(is_valid_parenthesis(s2)==False) # Bug is shown (1 mark)

(Note: s2="(()" or s2 = "" are not valid values of s2 that will reveal the bug. No marks for these)

c. Modification to the code:

Fix:

Add this statement at the end of the for loop (after line 7):

if open_count < 0:

    return False

Modified code:

```
1. def is_valid_parentheses(s):
2.     open_count = 0
3.     for char in s:
4.         if char == '(':
5.             open_count += 1
6.         elif char == ')':
7.             open_count -= 1
8.         if open_count <0:
9.             return false
10.      return open_count == 0
```

(1 mark)
Note: 1) Deduct 0.5 if the value of *open_count* is returned instead of the boolean value, True or False, of the check (open_count ==0)
2) For any alternative solutions given the code will be executed and evaluated accordingly.