Indian Statistical Institute

BStat, First Year, Mid-Sem of First Semester Examination, 2023-24

Introduction to Programming and Data Structures

Full Marks: 30 Date: 08-12-2023 Time: 2 Hours

Answer any *three* of the following questions

3 × 10 = 30

1. Write down the output of the following Python code segments. Justify your answer.

(a)
```
if True or not True and not True:
     print ('True')
else:
     print ('False')
```

(b)
```
input = 'Toray'
text = [i for i in input if i not in 'aeiou']
print(''.join(text))
```

(c)
```
i = -1
if i>>2 + 1:
     print('True')
else:
     print('False')
```

(d)
```
ls, i = [1, 2, 3, 4, 5, 6, 7], 0
result = ls[i]
for i in range(1, len(ls)):
     result ^= ls[i]
print(result)
```

(e)
```
def function(n):
     m = 0
     t = n
     while t != 0:
          m = (m * 10) + (t % 10)
          t //= 10
     return m == n
n = 101
if function(n) == 1:
     print('True')
else:
     print('False')
```

2+2+2+2+2

2. (a) The following incomplete Python program aims to print all the `n`-digit numbers whose digits are strictly increasing from left to right (the Most Significant Digit to Least Significant Digit). For example, for `n = 2`,

the output will be as follows.

```
01 02 03 04 05 06 07 08 09 12 13 14 15 16 17 18 19 23 24
25 26 27 28 29 34 35 36 37 38 39 45 46 47 48 49 56 57 58
59 67 68 69 78 79 89
```

Complete it by filling up the blank portion and justify your answer.

```
def findStrictlyIncreasing(start, out, n):
     if n == 0:
          print(out, end = ' ')
          return
     for i in range(start, 10):
          string = out + str(i)
          findStrictlyIncreasing(_______________________)
n = 2
findStrictlyIncreasing(0, '', n)
```

(b) The following incomplete Python program aims to find out the second largest number in a list. Assume that the second largest number is different than the largest one if there exists more than one distinct element in the list. Complete it by filling up the blank portion and justify your answer.

```
ls = [5, 10, 5, 27, 100, 19]
if ls[0] > ls[1]:
     large, seclarge = ls[0], ls[1]
else:
     large, seclarge = ls[1], ls[0]
for i in range(2, len(ls)):
     if ls[i] > large:
          seclarge = large
          large = ls[i]
     elif ls[i] > seclarge and large != ls[i]:
          seclarge = ls[i]
     elif _____________________________________________:
          seclarge = ls[i]
print('The second largest is : ', str(seclarge))
```

5+5

3. (a) Prove or disprove the following statement:

"The Boolean expression `(A or B) or ((not A) and (not B))` gets always evaluated as `True`".

(b) Show that the value of the expression `(1 + n + ~n)` evaluates to 0, for any negative integer `n`. Recall that ~ denotes the 1's complement operation.

(c) Prove that the expression $m\%(n+1)$ will return the same result as that of $m\&n$, where m and n are positive integers, and n is in the form 2^k-1, where k is a positive integer. Recall that `%` and `&` denote the modular division and bitwise AND operation, respectively.

3+3+4

4. (a) What is the purpose of the following Python code? Justify your answer.

```
def function(ls, count):
    if count <= 1:
        return 0
    function(ls, count-1)
    end = ls[count -1]
    j = count -2
    while j >= 0 and ls[j] > end:
        ls[j+1] = ls[j]
        j = j-1
    ls[j+1] = end
ls = [25, 16, 9, 1, 4, 36]
count = len(ls)
function(ls, count)
for i in range(count):
    print(ls[i], end = ' ')
```

(b) Write a *pseudocode* to take multiple strings of same length from the user, by asking the user to provide the number of strings and length of those strings as inputs beforehand, and output the first character of each string (in the same order they were received from the user), followed by the second character, and so on in each separate line. For an example, consider the following sample I/O.

Sample Input:

```
2 4
IMRA
AGET
```

Sample Output:

```
I
A
M
G
R
E
A
T
```

5+5