National University of Computer and Emerging Sciences

School of Computing **Fall 2015** **Islamabad Campus**

CS101
Introduction to Computing

Monday, October 19, 2015

Course Instructor(s)

Dr. Sibt ul Hussain, Dr. Abdul Malik and Ms. Uzma Maroof

Serial No:

Mid-II

Total Time: 1 Hour

Total Marks: 55

_____

Signature of Invigilator

Student Name _____ Roll No _____ Section _____ Signature _____

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Attempt on question paper. Attempt all of them. Read the question carefully, understand the question, and then attempt it.
2. No additional sheet will be provided for rough work. If you need more space write on the provided rough pages (at the back end of the paper) and clearly mark question and part number etc.
3. After asked to commence the exam, please verify that you have (**10**) different printed pages including this title page. There are total of (**4**) questions.
4. Use of calculator is strictly prohibited.
5. Use permanent ink pens only. Any part done using soft pencil will not be marked and cannot be claimed for rechecking.
6. Use **proper indentation** while writing code and make sure that your code is legible. Failing to do so can cost you marks.
7. **Please allocate your time properly according to the marks distribution.**
8. Write proper explanation of the error (or bug) where required, without proper explanation no marks will be awarded.

	I	II	III	IV	**Total**
Total Marks	25	10	10	10	55
Marks Obtained					

Vetted By: _____ Vetter Signature: _____

Question I . . . . . (25 Marks)

Please write proper explanation of the bug where required, without proper explanation no marks will be awarded.

(1) **(5 Marks)** What would be the output produced by executing the following Python code? Identify and correct errors, and write output, if any.

```
1  def carbonated( coke,  soda,  pop):
2      print("say " + soda + " not " + pop + " or " + coke)
3
4
5  def main():
6      soda = "coke";
7      pop = "pepsi";
8      coke = "pop";
9      pepsi = "soda";
10     say = pop;
11
12     carbonated(soda, pop, pepsi);
13     carbonated(coke, soda, pop);
14     carbonated(pop, pepsi, pepsi);
15     carbonated("pop", pop, "koolaid");
16     carbonated(say, "say", pop);
17     carbonated(say, "say", say);
18
19 main()
```

Output

(2) **(5 Marks)** What would be the output produced by executing the following Python code?

```
1  def fun(x,y):
2          x=x+1
3          y=y-1
4          return x+y
5  x=4
6  y=5
7  x=fun(x,y)
8  print(x+y)
9  y=fun(y,x)
10 print(x+y)
```

Output

(3) **(5 Marks)** What would be the output produced by executing the following Python code?

```
x=1
y=1
z=1
while x <= 2:
        y=1;
        string=x*"-"+"/"

        while y <= 3 :
                z=2;

                while z <= 4 :

                        string+="#";
                        z+=1;

                string+="/";
                y+=1;

        string+=x*"-"
        print(string);
        x+=1;
```

Output

(4) **(5 Marks)** What would be the output produced by executing the following Python code?

```
def listMystery(list):
        i=2
        while i < len(list):
                list[i] = list[i] + list[i - 1] + list[i - 2];
                i=i+1

                return list

print ( listMystery( [8] ) )
print ( listMystery( [3, 0, 1, 4, 7] ) )
print ( listMystery( [7, 4, -10, 8, 2] ) )
```

Output

(5) **(5 Marks)** What would be the output produced by executing the following Python code?

```
def trick(l,i):
        if i==1:
                return l[0];
        else :
                return l[1]

def foo(x):
    x[0] = 'def'
    x[1] = 'abc'
    return x[1]


q = ['abc', 'def']
print(trick(q,1) == foo(q))
print(trick(q,1) == foo(q))
print(trick(q,0) == foo(q))
```

Output

Question II . . . . . (10 Marks)

(1) **(5 Marks)** Write a function `isAllEven` that takes an list of integers as a parameter and that returns a boolean value indicating whether or not all of the values are even numbers (True for yes, False for no). For example, if a variable called list stores the following values, *i.e.*:

```
list=[18, 0, 4, 204, 8, 4, 2, 18, 206, 1492, 42]
```

Then the call `isAllEven(list)` should return True because each of these integers is an even number. If instead the list had stored these values:

```
list=[2, 4, 6, 8, 10, 208, 16, 7, 92, 14]
```

Then the call should return False because, although most of these values are even, the value 7 is an odd number. Write your solution to `isAllEven(list)` below.

(2) **(5 Marks)** Write a C++ program to display following pattern (6 rows and 10 columns) using `cout` statements (without using loops).

```
+--------+
|\       /|
| \     / |
|  \   /  |
|   \ /   |
+--------+
```

Question III . . . . . (10 Marks)

Write a Python program that displays the following pattern (there are 8 rows and 11 columns) on screen using loops.

```
+----------+
       *
      * *
     *   *
    *     *
   *       *
  *         *
*             *
+----------+
```

Question IV . . . . . (10 Marks)

Write a function named `evenOdd` that accepts a list of integers as a parameter and rearranges the list elements so that all of its odd elements are in positions with odd-numbered indexes and all of its even elements are in positions with even-numbered indexes. The list passed in will always contain exactly as many even values as odd values. The exact order of the elements in the list after your function is run on it is unimportant as long as its content alternates between even and odd values, starting with even ([even value, odd value, even value, odd value...]). For example, if your function were passed the following list:

```
a1 = [5, 6, 3, 3, 2, 5, 2, 6]
evenOdd(a1)
```

One acceptable ordering of the elements after the call would be:

```
a1=[6, 5, 6, 3, 2, 5, 2, 3]
```

The even-numbered indexes of this list are 0, 2, 4, 6 and each of these positions contains an even integer. The odd-numbered indexes of this list are 1, 3, 5, 7 and each of these positions contains an odd integer.

```
def evenOdd(list):
    """
    Arrange the elements in the list so that even and
    odd numbers should be at even and odd indeces respectively.

    """
    # Your Code Here
```

Rough Work

Rough Work