Introduction to Programming and Data Structures – Mock Test 1

Name:
Roll Number:

++

1. The following incomplete program aims to verify whether a given array is monotonic or not. Complete it by filling up the blanks and justify your answer.

```
def isMonotonic(A):
    x, y = [], []
    x.extend(A)
    y.extend(A)
    x.sort()
    y.sort(reverse = True)
    if _____________________________:
        return True
    return False
A = [13, 10, 3, 1]
print(isMonotonic(A))
```

Answer:
Justification:

2. The following incomplete program aims to rotate an array by *k* places to the left of it. Complete it by filling up the blanks and justify your answer.

```
def rotateArray(arr, n, k):
    temp = []
    i = 0
    while i < k:
        temp.append(arr[i])
        i = i + 1
    i = 0
    while _____________________________:
        arr[i] = arr[k]
        i = i + 1
        k = k + 1
    arr[:] = arr[: i] + temp
    return arr
arr = [1, 2, 3, 4, 5, 6, 7]
k = 3
print (rotateArray(arr, len(arr), k))
```

Answer:
Justification:

3. The following incomplete program aims to find out the largest element in a given array. Complete it by filling up the blanks and justify your answer.

```
def largest(arr, n):
    max = arr[0]
    for i in range(1, n):
        if _____________________________:
            max = arr[i]
    return max
arr = [10, 324, 45, 90, 9808]
n = len(arr)
print ('The largest in the given array: ', largest(arr, n))
```

Answer:
Justification:

4. The following incomplete program aims to verify whether a given number is prime or not. Complete it by filling up the blanks and justify your answer.

```
def Prime(number, i):
    if i == 1:
        return True
    if number % i == 0:
        return False
    if _____________________________________:
        return False
    return True
num = 13
print (Prime(num, int(num**0.5)+1))
```

Answer:
Justification:

5. The following incomplete program aims to print the duplicate elements from a list of integers. Complete it by filling up the blanks and justify your answer.

```
def Repeat(x):
    _size = len(x)
    repeated = []
    for i in range(_size):
        k = i + 1
        for j in range(k, _size):
            if ____________________________________________:
                repeated.append(x[i])
    return repeated
ls = [10, 20, 30, 20, 20, 30, 40, 50, -20, 60, 60, -20, -20]
print (Repeat(ls))
```

Answer:
Justification: