19204757054 (1)

MCA 101

UPC 223401101 Object Oriented Programming Nov/Dec. 2019

(For admissions 2019 onwards)

Time Allowed: 3 Hours M.M. 70

Instructions for candidates

1. Write your Roll no. on the top immediately on receipt the question paper.
2. Parts of a question must be answered together.

1 In each part of this question, you are given the definition of a function **copyLst** that is intended to return a copy of the list **lst** passed as input argument. The function is invoked in the global namespace. In each case, point out whether the code will execute successfully. If so, what will be the output of the **print** statement. If not, briefly state what does the function achieve? Also, show the contents of the run-time stack as execution of the code proceeds.

(a) 6

```
def copyLst(lst, index = 0):
    if (index == 0):
        nLst = []
    if index != len(lst)-1:
        nLst.append(lst[index])
        return copyLst(lst, index+1)
    else:
        return nLst
lst = [1, 2, 3]
print(copyLst(lst))
```

(b) 6

```
def copyLst(lst, lst1 = []):
    if lst == []:
        return lst
    lst1.append(lst[0])
    copyLst(lst[1:], lst1)

lst = [10, 20]
lst1 = copyLst(lst)
print(lst1)
lst1 = []
copyLst(lst, lst1)
print(lst1)
```

2 In each part of this question, you are given the definition of a function **removeAll**, intended to remove all occurrences of **value** from the list **lst**.

The function is used in the global namespace. In each case, point out whether the code will execute successfully. If so, what will be the output of the `print` statement. If not, briefly state what does the function achieve? Also, show the contents of the run-time stack as execution of the code proceeds.

(a) 6

```
def removeAll(lst, value, lst1 = []):
    if len(lst) >0:
        if lst[0] != value:
            lst1.append(lst[1:])
            removeAll(lst[1:], value, lst1)
lst = [10, 20, 20, 30]
lst1 = []
removeAll(lst, 20, lst1)
print(lst1)
```

(b) 6

```
def removeAll(lst, value, index = 0):
    if index == len(lst):
        return lst
    else:
        if lst[index] == value:
            lst.remove(value)
        removeAll(lst, value, index+1)
        return lst

lst = [10, 20, 20, 30]
removeAll(lst, 20)
print(lst)
```

(c) 6

```
def removeAll(lst, value, lst1 = [], i = 0):
    if i<len(lst):
        if lst[i] != value:
            lst1.append(lst[i])
            removeAll(lst, value, lst1, i+1)

lst = [10, 20, 20, 30, 20]
lst1 = removeAll(lst, 20)
print(lst1)
```

3 Consider a class **Employee** each of whose objects comprises three attributes **empID**, **name**, and **hourlyWages**. **Employee** instances have been stored in file **empData**. A text file **updateData** contains information about those (not all) employees whose **hourlyWages** have been revised. The information **empID** and revised **hourlyWages** has been stored in the file **updateData** line by line for each employee in question. You are required to write a program to create the updated file **updatedEmpData**. You should include necessary validation checks for file operations. 10

4 For each part in this question, indicate whether the code will execute successfully. If so, what will be the output produced. If not, indicate why.

(a) 4

```
def f():
    a = 5
    def g():
        b = a
        print('a = ', a, 'b = ', b)
        a = 5
    g()
f()
```

(b) 6

```
a = 4
def f():
    a = 5
    def f1():
        a = 7
        def g():
            nonlocal a
            print('in g, before update, a = ', a)
            a = 10
            print('in g, after update, a = ', a)
        print('in f1, before g(), a = ', a)
        g()
        print('in f1, after g(), a = ', a)
    f1()
    print('in f, after f1(), a = ', a)
f()
print('after f(), a = ', a)
```

5 Write a recursive function **commonDivisors** which accepts as input two non-negative numbers and produces the list of their common divisors. For example the function call `commonDivisors(48, 24)` will yield the list `[1, 2, 3, 4, 6, 8, 12, 24]`. 7

6 Write a function **dictUnion** that accepts two dictionaries as input arguments and returns their union as illustrated by the following example. For example, if `d1 = {1:10, 2:20, 3:30}`, `d2 = {4:10, 2:40, 3:30, 5:50}`, are the input arguments to the function **dictUnion**, it should return the dictionary `{1:10, 2:[20, 40], 3:30, 4:10, 5:50}` 7

7 Write a function which accepts as input two non-negative arguments **k** and **n**, and returns the largest **k** digit number divisible by **n**. If the input arguments are not valid non-negative integers the function should raise an exception. 6