Total sheets Used in this question- 6

4/4/2022, 9:00 AM
MCA
SEM-I
OBJECT ORIENTED PROGRAMMING
Course Code- 223401101
Exam Rollno- 2123475 7009

---

(3) →
(a) →
Ans-

```
dictionary = {'7410023111': 'Ajay', '7411122222':
                                        "Aniket"}

def MyTelephoneDict(name):
    res = []   # will store number
    for number, n in dictionary:
        if n == name:
            res.append(number)
                        # Appending the numbers
    if len(res) == 0:
        return "No Record Found"

    else:
        return res  # list of number
                    # where len(res) ≥ 1
# Driver code
if __name__ == "__main__":
    name = input("Enter name")
    result = MyTelephoneDict(name)
    print(result)
```

(1)

Output

```
Enter name   Anibet
['7411122222']
```

b) ⟶
Ans-

```
def   checkPalindrome (lst):
    """
        It will return list of True or
        False. represents the Corresponding
        element is palindrome or not

    """

    result = []
    for ele in lst:
            ele_str = str(ele) # because ele can of numeric
                                  type as well.


            i = 0
            j = len(ele_str) - 1
            flag = True   # Default Value
            while  i < j :
                if ele_str[i] == ele_str[j]:
                        i += 1
                        j -= 1

                else:
                        flag = False   # Not Palindrome
                        break
```

(2)

```
        result.append(flag)   # out of while loop
                              # but inside for loop
    return result


# Driver code

if __name__ == '__main__':
    res = checkPalindronge(['aniket', 'nitin', 'a',
                    777, 989, 99291, 1])

    print (res)
```

Output

```
[False, True, True, True, True, False, True]
```

```
def   getInterleaving(string1, string2, i, j, result, tempstring):
~~[illegible]~~
"""
  It is a helper method which is
  recursive. will be adding our result
  into 'result' list, tempstring holding the
  string ~~from which we are making~~ for
  temparary basis helps to generate interleaved
   string.
   i is the pointer to string1 and
   j is the pointer to string2.

"""
if   i< len(string1) and j< len(string2):
        getInterleaving(string1, string2, i+1, j, result,
                                    tempstring+string1[i])


        getInterleaving(string1, string2, i, j+1, result,
                                    tempstring + string2[j])


elif    i< len(string1):
        tempstring+=string1[i: ]
        result.append(tempstring).
        elif  j< len(string2):
              tempstring += string2[j:]
              result.append(tempstring)


def  printInterleaving (string1, string2):
        """
          It     prints  the   interleaving strings

        """
        res= []
        getInterleaving (string1, string2, 0, 0, res, '')
        print(res)
# Driver Code
if  __name__=='__main__':

        str1= input(" Enter String 1")
        str2= input (" Enter String 2")
        print Interleaving (str1, str2)
```

Output

```
Enter String1 AB
Enter String2 CD

['ABCD', 'ACBD', 'ACDB', 'CABD', 'CADB',
 'CDAB']
```