Solutions for Programming Questions (15CS664 – Python)

1. **Design a Python program to find the average of best two marks out of three marks taken as input.**

   Solution: Logic is : Find the smallest of three numbers using *min()* function. Then subtract smallest number from sum of all the 3 numbers. You will get two better marks!! Find their average.

```
x=int(input("Enter m1:"))
y=int(input("Enter m2:"))
z=int(input("Enter m3:"))

smallest=min(x,y,z)
avg= (x+y+z-smallest)/2
print("Average of best two marks = ",avg)
```

2. **A string with parentheses is well bracketed if all parentheses are matched: every opening bracket has a matching closing bracket and vice versa. Write a Python function *wellbracketed(s)* that takes a string s containing parentheses and returns True if s is well bracketed and False otherwise. Here are some examples to show how your function should work.**

>>> **wellbracketed("22)")**	**should return False**
>>> **wellbracketed("(a+b)(a-b)")**	**should return True**
>>> **wellbracketed("(a(b+c)-d)((e+f)")**	**should return False**

   Solution: Logic is: Initialize a counter variable (here it is *depth*) to 0. Take a loop which iterates on every character of input string. Each time when you encounter an opening bracket, increment *depth*. When you encounter closing bracket, decrement *depth*. In any step, if *depth* becomes negative, it indicates that a closing bracket has appeared before a matching opening bracket. In such a situation, you can terminate the loop, because the string is not-wellbracketed. At the end of all iterations, if *depth* value is zero, then the string is well-bracketed.

```
def wellbracketed(s):
    depth=0
    for i in s:
        if i=='(':
            depth+=1
        elif i==')':
            depth-=1

        if depth<0:
            return False
    if depth==0:
        return True
    else:
        return False

string=input("Enter a string:")
x=wellbracketed(string)
print(x)
```

3. **A positive integer m is a sum of squares if it can be written as k + x where k > 0, x > 0 and both k and x are perfect squares. Write a Python function *sumofsquares(m)* that takes an integer m returns True if m is a sum of squares and False otherwise. Here are some examples to show how your function should work.**

>>> **sumofsquares(41)**	**should return True**
>>> **sumofsquares(30)**	**should return False**
>>> **sumofsquares(17)**	**should return True**

Solution:

```
import math

def sumofsquares(m):
    r=math.floor(math.sqrt(m))
    for j in range(1,r+1):
        for k in range(r,0,-1):
            if j**2+k**2==m:
                return True
    return False

x=int(input("Enter a number:"))
result=sumofsquares(x)
print(x)
```

4. **Write a program to reverse a string.**
Solution: Reverse of a string can be done using string slicing with stride = -1

```
s= input("Enter a string")
s1=s[::-1]
print("Reverse is", s1)
```

5. **Write a program to display all the palindrome words appearing in an input text.**
Solution: Logic is – Read a string (in fact, a sentence). Write a loop which iterates on a string word-by-word. As there will be a space between two words, if you use *split()* function based on white-space, you will get one word at a time in a loop. Apply slicing on this word to reverse. Compare original word with reversed word to check any word is palindrome. Print only those which are palindrome.

```
s=input("Enter a string:")
for word in s.split(' '):
    if word==word[::-1]:
        print(word)
```

6. **Write a program to list out all prime numbers between two integers m and n.**
Solution: A prime number is one, which is divisible by 1 and itself. Note that, a number cannot have a factor more than half of itself. So, we will take outer loop which ranges from m to n, and an inner loop which ranges from 2 to half of the outer loop variable.

```
m=int(input("Enter m:"))
n=int(input("Enter n:"))

for i in range(m,n+1):
    for j in range(2,i//2 +1):
        if i%j==0:
            break
    else:
        print(i,end='\t')
```

7. **Write a program to count the frequency of characters in a string.**
Solution: Use *count()* method of string class.

```
string=input("Enter a string:")
char=input("Enter character to be counted:")
c=string.count(char)
print(char, "has appeared",c, "times")
```

8. **Write a Python function to get a string made of the first 2 and the last 2 chars from a given string. If the string length is less than 2, return empty string, otherwise return new string created.**
Solution: Task can be done using slicing.

```
def substring(s):
    if len(s)<2:
        return ''
    else:
        sub=s[:2]+s[-2:]
        return sub

s=input("Enter a string:")
s1=substring(s)
print("Extracted string is:",s1)
```

9. **Consider a string s="Hello how are you?". Write a program to display this string without spaces.**

   Solution: Use *replace()* function of string class. Replace white spaces with empty string and print the same.

```
s=input("Enter a string:")
s1=s.replace(' ', '')
print("String without space:",s1)
```

10. **Write a Python program to remove the characters which have odd index values of a given string.**

    Solution: Can be achieved using slicing with stride value =2

```
s=input("Enter a string:")
s1=s[::2]
print("String with odd indexed characters removed:",s1)
```

11. **Write a Python program to change a given string to a new string where the first and last chars have been exchanged.**

    Solution: Can be done with slicing with negative indexing.

```
s=input("Enter a string:")
s1=s[-1]+s[1:-1]+s[0]
print("String first and last characters swapped:",s1)
```

12. **Write a Python program to add 'ing' at the end of a given string (length should be at least 3). If the given string already ends with 'ing' then add 'ly' instead. If the string length of the given string is less than 3, leave it unchanged**

Sample String : 'abc'	**Expected Result : 'abcing'**
Sample String : 'string'	**Expected Result : 'stringly'**

    Solution: Whether string ends with 'ing' or not can be verified using slicing. Then based on this check, we can add either 'ing' or 'ly'.

```
def addprefix(s):
    if len(s)<3:
        return s
    else:
        if s[-3:]!='ing':
            return s+'ing'
        else:
            return s+'ly'

s=input("Enter a string:")
s1=addprefix(s)
print("After adding suitable prefix:",s1)
```

13. Write a program to check whether a given number is palindrome or not.

Solution: Read a number. Apply modulus operator to get a remainder after dividing number by 10. Multiply remainder by 10. Divide the number by 10 to get integral part of quotient. Continue this procedure till original number becomes zero. You will get the reversed number. Compare original number and reversed one to check for palindrome. (Note: if you treat the number as string, without converting it into integer format, then you can use slicing to reverse it!!)

```
x=int(input("Enter a number:"))
n=x
rev=0
while n>0:
    rem=n%10
    rev=rev*10+rem
    n=n//10

print("Reverse number is:",rev)
if x==rev:
    print(x, "is a palindrome")
else:
    print(x, "is not a palindrome")
```

14. Write a program to perform simulation of simple calculator for basic operations like +, -, *, /, % etc. (Hint: Read a character +, - etc as operator from the user. Perform respective operation based on the operator. And Display the result)

```
def Calci(a,b,op):
    if op=='+':
        return a+b
    elif op=='-':
        return a-b
    elif op=='*':
        return a*b
    elif op=='/':
        return a/b
    elif op=='%':
        return a%b
    else:
        return "Invalid Operator"

x=int(input("Enter x:"))
y=int(input("Enter y:"))
op=input("Enter Operator:")

result=Calci(x,y,op)
```