Started on	Saturday, 20 April 2024, 2:18 PM
State	Finished
Completed on	Saturday, 20 April 2024, 3:49 PM
Time taken	1 hour 31 mins
Marks	5.00/5.00
Grade	**50.00** out of 50.00 (**100**%)
Name	SHEEBA SHARON A 2022-CSD-A

Question **1**

Correct

Mark 1.00 out of 1.00

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.

Hint: Keep track of the nesting depth of brackets. Initially the depth is 0. The depth increases with each opening bracket and decreases with each closing bracket. What are the constraints on the value of the nesting depth for the string to be wellbracketed?

Here are some examples to show how your function should work.

>>> wellbracketed("22)")
False

>>> wellbracketed("(a+b)(a-b)")
True

>>> wellbracketed("(a(b+c)-d)((e+f)")
False

Answer: (penalty regime: 0 %)

Reset answer

```
def wellbracketed(s):
    open_brac=0
    close_brac=0
    flag=0
    for ele in s:
        if(ele=='('):
            open_brac+=1
        elif ele==')':
            open_brac-=1
        if open_brac<0:
            flag+=1
    return flag==0 and not open_brac>0
```

	Test	Expected	Got	
✔	print(wellbracketed("22)"))	False	False	✔
✔	print(wellbracketed("(a+b)(a-b)"))	True	True	✔
✔	print(wellbracketed("(a(b+c)-d)((e+f)"))	False	False	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **2**

Correct

Mark 1.00 out of 1.00

Euclid was a Greek mathematician who lived approximately 2,300 years ago. His algorithm for computing the greatest common divisor of two positive integers, a and b, is both efficient and recursive. It is outlined below:

If b is 0 then
  eturn a
Else
  Set c equal to the remainder when a is divided by b
  Return the greatest common divisor of b and c

Write a program that implements Euclid's algorithm and uses it to determine the greatest common divisor of two integers entered by the user. Test your program with some very large integers. The result will be computed quickly, even for huge numbers consisting of hundreds of digits, because Euclid's algorithm is extremely efficient.

Answer: (penalty regime: 0 %)

```
def gcf(a,b):
    if b==0:
        return a
    else:
        return gcf(b,a%b)

n1=int(input())
n2=int(input())
print(gcf(n1,n2))
```

	Input	Expected	Got	
✔	8 12	4	4	✔
✔	720 1000	40	40	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **3**

Correct

Mark 1.00 out of 1.00

Given an integer n, return an list of length n + 1 such that for each i (0 <= i <= n), ans[i] is the number of 1's in the binary representation of i.

Example:

```
Input: n = 2
Output: [0,1,1]
Explanation:
0 --> 0
1 --> 1
2 --> 10
```

Example2:

```
Input: n = 5
Output: [0,1,1,2,1,2]
Explanation:
0 --> 0
1 --> 1
2 --> 10
3 --> 11
4 --> 100
5 --> 101
```

Note: Complete the given function alone

For example:

Test	Result
print(CountingBits(5))	[0, 1, 1, 2, 1, 2]

Answer: (penalty regime: 0 %)

Reset answer

```
def CountingBits(n):
    count=[]
    i=0
    for num in range(0,n+1):
        s=0

        while num!=0:
            if num%2==1:
                s+=1
            num=num//2

        count.append(s)
    return count
```

	Test	Expected	Got	
✔	`print(CountingBits(2))`	`[0, 1, 1]`	`[0, 1, 1]`	✔
✔	`print(CountingBits(5))`	`[0, 1, 1, 2, 1, 2]`	`[0, 1, 1, 2, 1, 2]`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **4**

Correct

Mark 1.00 out of 1.00

Write a Python function sumofsquares(m) that takes an integer m returns True if m is a sum of squares and False otherwise. (If m is not positive, your function should return False.)

Here are some examples to show how your function should work.
>>> sumofsquares(41)
True

>>> sumofsquares(30)
False

>>> sumofsquares(17)
True

Answer: (penalty regime: 0 %)

Reset answer

```
from math import *

def issquare(n):
    k = int(sqrt(n))
    return(k*k == n)

def sumofsquares(m):
    for num in range(m//2):
        res=issquare(num) and issquare(m-num)
        if res==True:
            break
    return res

```

	Test	Expected	Got	
✔	print(sumofsquares(41))	True	True	✔
✔	print(sumofsquares(30))	False	False	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **5**

Correct

Mark 1.00 out of 1.00

A prime number is an integer greater than one that is only divisible by one and itself. Write a function that determines whether or not its parameter is prime, returning True if it is, and False otherwise.

Answer: (penalty regime: 0 %)

Reset answer

```
def isPrime(n):
    count=0
    for i in range(1,n+1):
        if n%i==0:
            count+=1

    return count==2
```

	Test	Expected	Got	
✔	print(isPrime(1))	False	False	✔
✔	print(isPrime(2))	True	True	✔
✔	print(isPrime(3))	True	True	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Week-07_MCQ

Jump to...

WEEK-07-Extra ►