Rajalakshmi Engineering College

Name: Nishal P
Email: 241501130@rajalakshmi.edu.in
Roll no: 241501130
Phone: 9790965308
Branch: REC
Department: I AI & ML FB
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 2_COD_Updated

Attempt : 1
Total Mark : 50
Marks Obtained : 50

Section 1 : Coding

1. Problem Statement

You work as an instructor at a math enrichment program, and your goal is to develop a program that showcases the concept of using control statements to manipulate loops. Your task is to create a program that takes an integer 'n' as input and prints the squares of even numbers from 1 to 'n', while skipping odd numbers.

Input Format

The input consists of a single integer, which represents the upper limit of the range.

Output Format

The output displays the square of even numbers from 1 to 'n' separated by lines.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10

Output: 4
16
36
64
100

Answer

```
n=int(input())
for i in range(2,n+1,2):
    if i<=n:
        sq=pow(i,2)
        print(sq)
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

As a junior developer working on a text analysis project, your task is to create a program that displays the consonants in a sentence provided by the user, separated by spaces.

You need to implement a program that takes a sentence as input and prints the consonants while skipping vowels and non-alphabetic characters using only control statements.

Input Format

The input consists of a string representing the sentence.

Output Format

The output displays space-separated consonants present in the sentence.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: Hello World!

Output: H l l W r l d

Answer

```
s=input()
v=['A','E','I','O','U']
c=[]
for l in s:
   if l.isalpha() and l.upper() not in v:
      c.append(l)
for l in c:
   print(l,end=" ")
print()
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Ethan, a curious mathematician, is fascinated by perfect numbers. A perfect number is a number that equals the sum of its proper divisors (excluding itself). Ethan wants to identify all perfect numbers within a given range.

Help him write a program to list these numbers.

Input Format

The first line of input consists of an integer start, representing the starting number of the range.

The second line consists of an integer end, representing the ending number of the range.

Output Format

The output prints all perfect numbers in the range, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
100

Output: 6 28

Answer

```
a=int(input())
b=int(input())

def is_perfect(n):
   sum_of_divisors=sum(i for i in range(1,n) if n%i==0)
   return sum_of_divisors==n
#perfect_numbers=n
for n in range(a,b):
   if is_perfect(n):
      print(n,end=" ")
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Emma, a mathematics enthusiast, is exploring a range of numbers and wants to count how many of them are not Fibonacci numbers.

Help Emma determine the count of non-Fibonacci numbers within the given range [start, end] using the continue statement.

Input Format

The first line of input consists of an integer, representing the starting number of the range.

The second line consists of an integer, representing the ending number of the range.

Output Format

The output prints a single integer, representing the count of numbers in the range that are not Fibonacci numbers.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
10

Output: 5

Answer

```
s=int(input())
e=int(input())
def is_fibonnaci(n):
   a,b=0,1
   while a<=n:
      if a==n:
         return True
      a,b=b, a+b
   return False
start, end=s,e
count=0
for num in range(start,end+1):
   if is_fibonnaci(num):
      continue
   count+=1
print(count)
```

Status : Correct ***Marks : 10/10***

5. Problem Statement

John, a software developer, is analyzing a sequence of numbers within a given range to calculate their digit sum. However, to simplify his task, he excludes all numbers that are palindromes (numbers that read the same backward as forward).

Help John find the total sum of the digits of non-palindromic numbers in the range [start, end] (both inclusive).

Example:

Input:

10

20

Output:

55

Explanation:

Range [10, 20]: Non-palindromic numbers are 10, 12, 13, 14, 15, 16, 17, 18, 19 and 20.

Digit sums: 1+0 + 1+2 + 1+3 + 1+4 + 1+5 + 1+6 + 1+7 + 1+8 + 1+9 + 2+0 = 55.

Output: 55

Input Format

The first line of input consists of an integer, representing the starting number of the range.

The second line of input consists of an integer, representing the ending number of the range.

Output Format

The output prints a single integer, representing the total sum of the digits of all non-palindromic numbers in the range.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10

20
Output: 55

Answer

```
def is_palindrome(n):
   return str(n) ==str(n)[::-1]

def sum_of_digits(n):
   return sum(int(digit) for digit in str(n))
start=int(input())
end=int(input())
total_sum=0
for num in range(start,end+1):
   if is_palindrome(num):
      continue
   total_sum+=sum_of_digits(num)
print(total_sum)
```

Status : Correct ***Marks : 10/10***

Rajalakshmi Engineering College

Name: Nishal P
Email: 241501130@rajalakshmi.edu.in
Roll no: 241501130
Phone: 9790965308
Branch: REC
Department: I AI & ML FB
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 2_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

John is tasked with configuring the lighting for a high-profile event, where different lighting modes affect the ambiance of the venue. He can choose from three distinct lighting modes, each requiring a specific adjustment to the initial light intensity:

Ambient Lighting (Mode 1): The intensity level is multiplied by 1.5.Stage Lighting (Mode 2): The intensity level is multiplied by 2.0.Spotlight (Mode 3): The intensity level is multiplied by 1.8.

In the event that an invalid mode is provided, the program should output an error message indicating the invalid selection.

Your task is to write a program that reads the selected lighting mode and the initial intensity level, applies the appropriate adjustment, and prints the final intensity.

Input Format

The first line of input is an integer n, representing the lighting mode.

The second line is a floating value m, representing the initial intensity level of the light.

Output Format

The output displays "Intensity: " followed by a float representing the adjusted intensity level, formatted to two decimal places, if the mode is valid.

If the mode is invalid, the output should display "Invalid".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
10.0

Output: Intensity: 15.00

Answer

```
a=int(input())
b=float(input())
if(a==1):
   c=b*1.5
   print(f"Intensity: {c:.2f}")
elif(a==2):
   c=b*2.0
   print(f"Intensity: {c:.2f}")
elif(a==3):
   c=b*1.8
   print(f"Intensity: {c:.2f}")
else:
   print("Invalid")
```

Status : Correct *Marks : 10/10*

2. Problem Statement

Rohith is a data analyst who needs to categorize countries based on their population growth rates. Each country is assigned a unique code. Rohith will receive a code and corresponding data based on the code. If the data falls within specific thresholds, he needs to classify the country's priority level.

Your task is to write a program that reads a country code and its associated data, and then determines if the priority is "High" or "Low."

Thresholds:France: Priority is "High" if the percentage < 50, else "Low".Japan: Priority is "High" if life expectancy > 80, else "Low".Brazil: Priority is "High" if the urban population > 80, else "Low".

Input Format

The first line of input consists of an integer, representing the country code (1 for France, 2 for Japan, 3 for Brazil).

If the country code is 1,

- The second line consists of a floating-point value N, representing the percentage of the English-speaking population.

If the country code is 2,

- The second line consists of a floating-point value A, representing the average life expectancy in years.

If the country code is 3,

- The second line consists of a floating-point value P, representing the percentage of the urban population.

Output Format

The first line of output displays "Priority: High" or "Priority: Low" based on the input data.

If the country code is invalid, print "Invalid".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
30.0

Output: Priority: High

Answer

```
a=int(input())
if(a==1):
   b=float(input())
   if(b<50):
      print("Priority: High")
   else:
      print("Priority: Low")
elif(a==2):
   b=float(input())
   if(b>80):
      print("Priority: High")
   else:
      print("Priority: Low")
elif(a==3):
   b=float(input())
   if(b>80):
      print("Priority: High")
   else:
      print("Priority: Low")
else:
   print("Invalid")
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Taylor is tasked with a mathematical challenge that requires finding the smallest positive number divisible by all integers from 1 to n.

Help Taylor to determine the smallest positive number that is divisible by all integers from 1 to n. Make sure to employ the break statement to ensure efficiency in the program.

Input Format

The input consists of a single integer, n.

Output Format

The output displays the smallest positive number that is divisible by all integers from 1 to n.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10

Output: 2520

Answer

```
import math
def lcm(n):
   ans=1
   for i in range(1, n+1):
      ans=int((ans*i)/math.gcd(ans,i))
   return ans
n=int(input())
print(lcm(n))
```

Status : Correct *Marks : 10/10*

4. Problem Statement

Nisha is a mathematics enthusiast, eager to explore the realm of twin prime numbers. The objective is to develop a program that enables the discovery and presentation of twin prime pairs.

The program should take an integer 'n' as input and generate 'n' pairs of twin primes, displaying the pairs with a difference of 2 between them.

Input Format

The input consists of a single integer, n.

Output Format

The output displays the 'n' pairs of twin primes, the pairs with a difference of 2 between them.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5

```
Output: 3 5
5 7
11 13
17 19
29 31
```

Answer

```
def is_prime(num):
    if num==1:
        return False
    for i in range(2,int(num**0.5)+1):
        if num%i==0:
            return False
    return True

def generate_twin_prime(n):
    count=0
    num=2
    while count<n:
        if is_prime(num) and is_prime(num+2):
            print(f"{num} {num+2}")
            count+=1
        num+=1

n=int(input())
generate_twin_prime(n)
```

Status : Correct *Marks : 10/10*