Rajalakshmi Engineering College

Name: Prasham Jaganathan
Email: 241501148@rajalakshmi.edu.in
Roll no: 241501148
Phone: 9445840008
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 : 42.5

Section 1 : Coding

1. 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

```
# You are using Python
# Function to extract consonants
def extract_consonants(sentence):
   vowels = "aeiouAEIOU"
   consonants = ""

   for char in sentence:
      if char.isalpha() and char not in vowels:
         consonants += char + " "
   print(consonants.strip())

# Input from user
sentence = input()

# Call the function with user input
extract_consonants(sentence)
```

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

2. 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

```
# You are using Python
# Function to print the squares of even numbers
def print_even_squares(n):
    for i in range(1, n + 1):
        if i % 2 == 0:  # Check if the number is even
            print(i * i)  # Print the square of the even number

# Input from user
n = int(input())

# Validate the input and execute the function
if 1 <= n <= 30:  # Ensure the input meets constraints
    print_even_squares(n)
else:
    print("Invalid input")
```

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

3. 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

```
# You are using Python
# Function to check if a number is a palindrome
def is_palindrome(num):
   str_num = str(num)
   return str_num == str_num[::-1]

# Function to calculate the total digit sum of non-palindromic numbers
def digit_sum_non_palindromes(start, end):
   total_sum = 0
   for num in range(start, end + 1):
      if not is_palindrome(num):  # Check if the number is not a palindrome
         # Add the sum of digits of the number
         total_sum += sum(int(digit) for digit in str(num))
   return total_sum

# Input start and end of the range
try:
   start = int(input(""))
   end = int(input(""))

   # Validate constraints and compute the result
   if 1 <= start <= end <= 100:  # Ensure inputs fall within the constraints
      result = digit_sum_non_palindromes(start, end)
      print(result)
   else:
      print("Invalid input")
except ValueError:
   print("Invalid input. Please enter valid integers.")
```

Status : Partially correct ***Marks : 2.5/10***

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

```
# You are using Python
# Input start and end of range
start = int(input())
end = int(input())

# Generate Fibonacci numbers up to 'end'
fib_set = set()
a, b = 0, 1
while a <= end:
    fib_set.add(a)
    a, b = b, a + b
```

```
# Count non-Fibonacci numbers in the range
count = 0
for i in range(start, end + 1):
   if i in fib_set:
      continue  # Skip if it's a Fibonacci number
   count += 1

# Output the result
print(count)
```

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

5. 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

```
# You are using Python
# Input start and end of the range
start = int(input())
end = int(input())

# Function to check if a number is perfect
def is_perfect(n):
    sum_divisors = 0
    for i in range(1, n):
        if n % i == 0:
            sum_divisors += i
    return sum_divisors == n

# List to store perfect numbers
perfect_numbers = []

for num in range(start, end + 1):
    if is_perfect(num):
        perfect_numbers.append(num)

# Print the result
print(" ".join(map(str, perfect_numbers)))
```

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

Rajalakshmi Engineering College

Name: Prasham Jaganathan
Email: 241501148@rajalakshmi.edu.in
Roll no: 241501148
Phone: 9445840008
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

Max is fascinated by prime numbers and the Fibonacci sequence. He wants to combine these two interests by creating a program that outputs the first n prime numbers within the Fibonacci sequence.

Your task is to help Max by writing a program that prints the first n prime numbers in the Fibonacci sequence using a while loop along with the break statement to achieve the desired functionality.

Input Format

The input consists of an integer n, representing the number of prime Fibonacci numbers to generate.

Output Format

The output displays space-separated first n prime numbers found in the Fibonacci sequence.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5

Output: 2 3 5 13 89

Answer

```
n = int(input())


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


fib_primes = []
a, b = 0, 1

while True:
   if is_prime(a):
      fib_primes.append(a)
      if len(fib_primes) == n:
         break
   a, b = b, a + b


print(" ".join(map(str, fib_primes)))
```

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

2. 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

```
n = int(input())


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

```
        return False
    return True


count = 0
num = 3

while count < n:
    if is_prime(num) and is_prime(num + 2):
        print(num, num + 2, end=" ")
        count += 1
    num += 1
```

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

3. Problem Statement

Gabriel is working on a wildlife research project where he needs to compute various metrics for different animals based on their characteristics. Each animal type requires a different calculation: a deer's distance traveled, a bear's weight based on footprint size, or a bird's altitude based on its flying pattern.

Conditions:

For Deer (Mode 'D' or 'd'): Distance = speed of sound * time taken, where the speed of sound in air is 343 meters per second.For Bear (Mode 'B' or 'b'): Weight = footprint size * average weight, where the average weight per square inch for a bear is 5.0 pounds.For Bird (Mode 'F' or 'f'): Altitude = flying pattern * distance covered (in meters).

Write a program to help Gabriel analyze the characteristics of animals based on the given inputs.

Input Format

The first line of input consists of a character, representing the type of animal 'D/d' for deer, 'B/b' for bear, and 'F/f' for bird.

If the choice is 'D' or 'd':

The second line of input consists of a floating-point value T, representing the time taken from the deer's location to the observer.

If the choice is 'B' or 'b':

The second line of input consists of a floating-point value S, representing the size of the bear's footprint in square inches.

If the choice is 'F' or 'f':

1. The second line of input consists of a floating-point value P, representing the bird's flying pattern.
2. The third line consists of a floating-point value D, representing the distance covered by the bird in meters.

Output Format

The output prints one of the following:

If the choice is 'D' or 'd':

The output prints "Distance: X m" where X is a floating point value rounded off to two decimal places, representing the calculated distance traveled by the sound wave in meters.

If the choice is 'B' or 'b':

The output prints "Weight: Y lb" where Y is a floating point value rounded off to two decimal places, representing the estimated weight of the bear in pounds.

If the choice is 'F' or 'f':

The output prints "Altitude: Z m" where Z is a floating point value rounded off to two decimal places, representing the calculated altitude of the bird's flight in meters.

If the given choice is invalid, print "Invalid".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: d
2.5
Output: Distance: 857.50 m

Answer

```
choice = input().strip()


SPEED_OF_SOUND = 343
BEAR_WEIGHT_PER_SQIN = 5.0


if choice in ('D', 'd'):
   T = float(input())
   distance = SPEED_OF_SOUND * T
   print(f"Distance: {distance:.2f} m")

elif choice in ('B', 'b'):
   S = float(input())
   weight = S * BEAR_WEIGHT_PER_SQIN
   print(f"Weight: {weight:.2f} lb")

elif choice in ('F', 'f'):
   P = float(input())
   D = float(input())
   altitude = P * D
   print(f"Altitude: {altitude:.2f} m")

else:
   print("Invalid")
```

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

4. Problem Statement

Alex is practicing programming and is curious about prime and non-prime digits. He wants to write a program that calculates the sum of the non-prime digits in a given integer using loops.

Help Alex to complete his task.

Example:

Input:

845

output:

12

Explanation:

Digits: 8 (non-prime), 4 (non-prime), 5 (prime)

The sum of Non-Prime Digits: 8 + 4 = 12

Output: 12

Input Format

The input consists of a single integer X.

Output Format

The output prints an integer representing the sum of non-prime digits in X.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 845
Output: 12

Answer

```
X = input().strip()

prime_digits = {'2', '3', '5', '7'}
```

```
non_prime_sum = 0


for digit in X:
    if digit not in prime_digits:
        non_prime_sum += int(digit)


print(non_prime_sum)
```

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