Lab:2

Name:M.Ramsai

Roll Number: 2503A51L53

Course Code: 24CS002PC215

Course Title: AI Assisted Coding

Assignment Number: 2

Academic Year: 2025-2026

**TASK 2**

**Prompt given** :write a python function that check whether the number is a amstrong number…

def is\_armstrong(number):

  """Checks if a number is an Armstrong number."""

  num\_str = str(number)

  n = len(num\_str)

  armstrong\_sum = 0

  for digit in num\_str:

    armstrong\_sum += int(digit) \*\* n

  return armstrong\_sum == number

# Example usage

num = 153

if is\_armstrong(num):

  print(f"{num} is an Armstrong number.")

else:

  print(f"{num} is not an Armstrong number.")

num = 123

if is\_armstrong(num):

  print(f"{num} is an Armstrong number.")

else:

  print(f"{num} is not an Armstrong number.")

OUTPUT: 153 is an Armstrong number.

123 is not an Armstrong number.

**COPILOT CODE:**

def is\_armstrong(number):

    """Check if a number is an Armstrong number."""

    digits = [int(d) for d in str(number)]

    power = len(digits)

    total = sum(d \*\* power for d in digits)

    return total == number

# Example usage

example\_numbers = [153, 370, 371, 9474, 123, 100]

for num in example\_numbers:

    print(f"{num} is Armstrong: {is\_armstrong(num)}")

OUTPUT: 153 is Armstrong: True

370 is Armstrong: True

371 is Armstrong: True

9474 is Armstrong: True

123 is Armstrong: False

100 is Armstrong: False

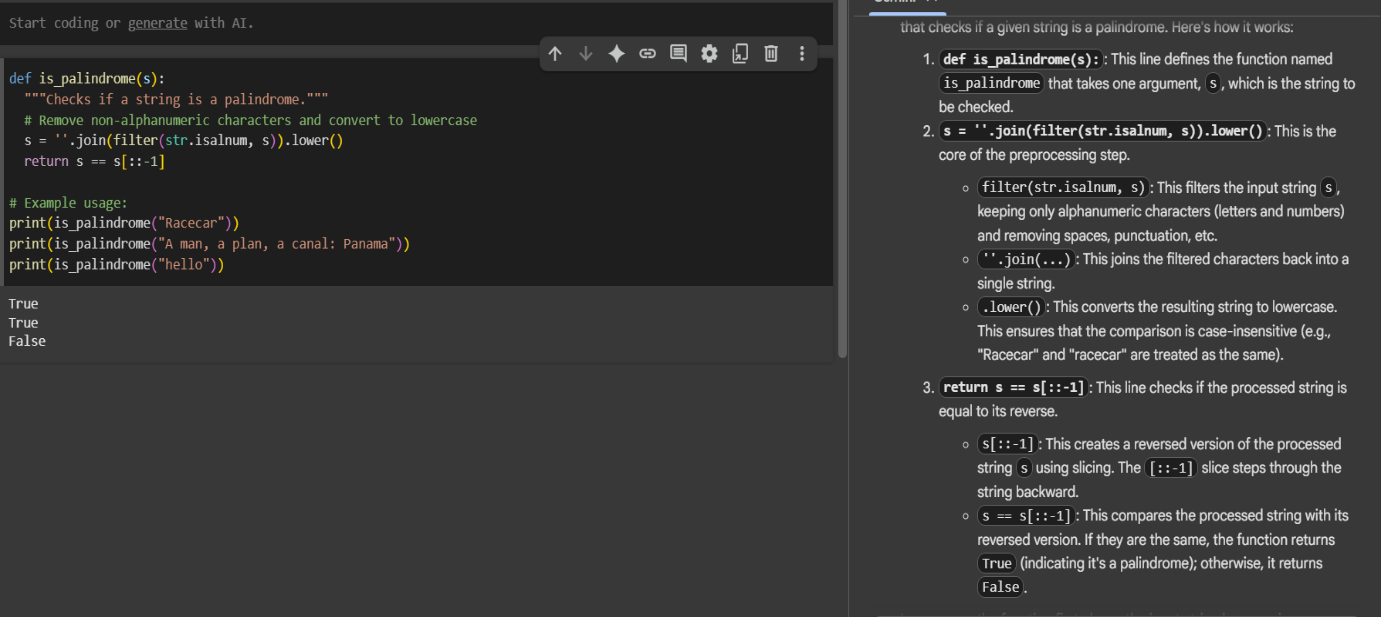
PS C:\Users\ramsa>..

PROMPT GIVEN: write a python function that check whether the number is a amstrong number…

A screenshot of a computer program

AI-generated content may be incorrect.

**TASK 3**



**EXPLANATION:** This code snippet defines a function called is\_palindrome that checks if a given string is a palindrome. Here's how it works:

1. **def is\_palindrome(s):**: This line defines the function named is\_palindrome that takes one argument, s, which is the string to be checked.
2. **s = ''.join(filter(str.isalnum, s)).lower()**: This is the core of the preprocessing step.
   * filter(str.isalnum, s): This filters the input string s, keeping only alphanumeric characters (letters and numbers) and removing spaces, punctuation, etc.
   * ''.join(...): This joins the filtered characters back into a single string.
   * .lower(): This converts the resulting string to lowercase. This ensures that the comparison is case-insensitive (e.g., "Racecar" and "racecar" are treated as the same).
3. **return s == s[::-1]**: This line checks if the processed string is equal to its reverse.
   * s[::-1]: This creates a reversed version of the processed string s using slicing. The [::-1] slice steps through the string backward.
   * s == s[::-1]: This compares the processed string with its reversed version. If they are the same, the function returns True (indicating it's a palindrome); otherwise, it returns False.

In summary, the function first cleans the input string by removing non-alphanumeric characters and converting it to lowercase, and then it checks if the cleaned string reads the same forwards and backward.

**PROMPT GIVEN: 1.** Write a python function to check the palindrome…

**2.**Explain the function that had given by the gemini above…

**TASK 4**

**CURSUR AI CODE**: **A computer screen shot of a program

AI-generated content may be incorrect.**

**PROMPT GIVEN:** Generate a Python function for sum of the first N natural numbers.

OUTPUT: sum\_first\_n(0) = 0

sum\_first\_n(1) = 1

sum\_first\_n(5) = 15

sum\_first\_n(10) = 55

sum\_first\_n(100) = 5050

**GEMINI AI CODE:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

Prompt Given: Generate a Python function for sum of the first N natural numbers.

Output: The sum of the first 10 natural numbers is: 55.

**TASK 5**

* Python program to calculate the sum of odd numbers and even numbers in a given tuple.

# Given tuple

numbers = (12, 3, 5, 8, 15, 20, 7, 10)

# Initialize sums

sum\_even = 0

sum\_odd = 0

# Loop through the tuple

for num in numbers:

if num % 2 == 0:

sum\_even += num

else:

sum\_odd += num

# Display results

print("Sum of even numbers:", sum\_even)

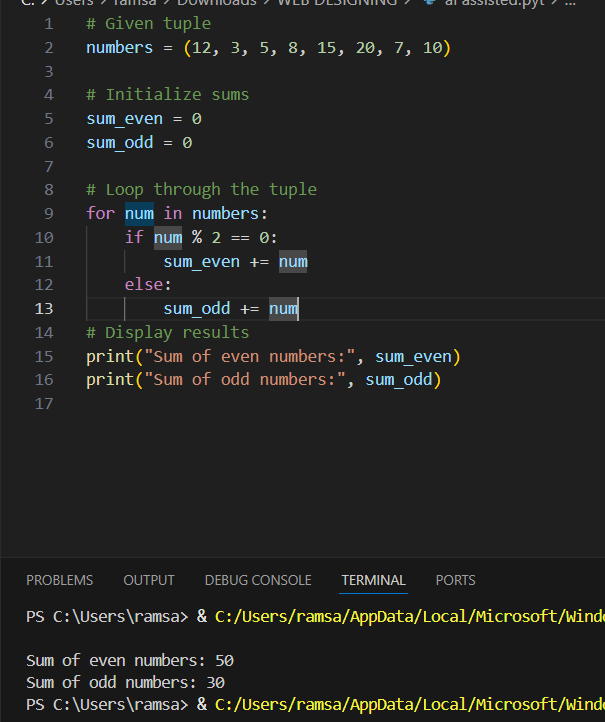
print("Sum of odd numbers:", sum\_odd)

**OUTPUT:**

Sum of even numbers: 50

Sum of odd numbers: 30

SCREENSHOT:



EXPLANATION:

We have a set of numbers stored in a tuple.  
We want to find two things:

1. The total (sum) of all the **even** numbers.
2. The total (sum) of all the **odd** numbers.

To do this:

* First, we create two counters: one for even numbers and one for odd numbers. Both start at zero.
* Then, we look at each number one by one.
* If the number can be divided by 2 without leaving anything left over, it’s an **even** number, so we add it to the “even” total.
* Otherwise, it’s an **odd** number, so we add it to the “odd” total.
* After checking all the numbers, we show both totals.

For example, with these numbers:

12, 3, 5, 8, 15, 20, 7, 10

* Even numbers are: 12, 8, 20, 10 → sum = **50**
* Odd numbers are: 3, 5, 15, 7 → sum = **30**

So the program will display:

Sum of even numbers: 50

Sum of odd numbers: 30….

**TASK:1**

PROMPT GIVEN: write a Python function that reads a list of numbers and calculates the mean, minimum, and maximum values..A screenshot of a computer program

AI-generated content may be incorrect.