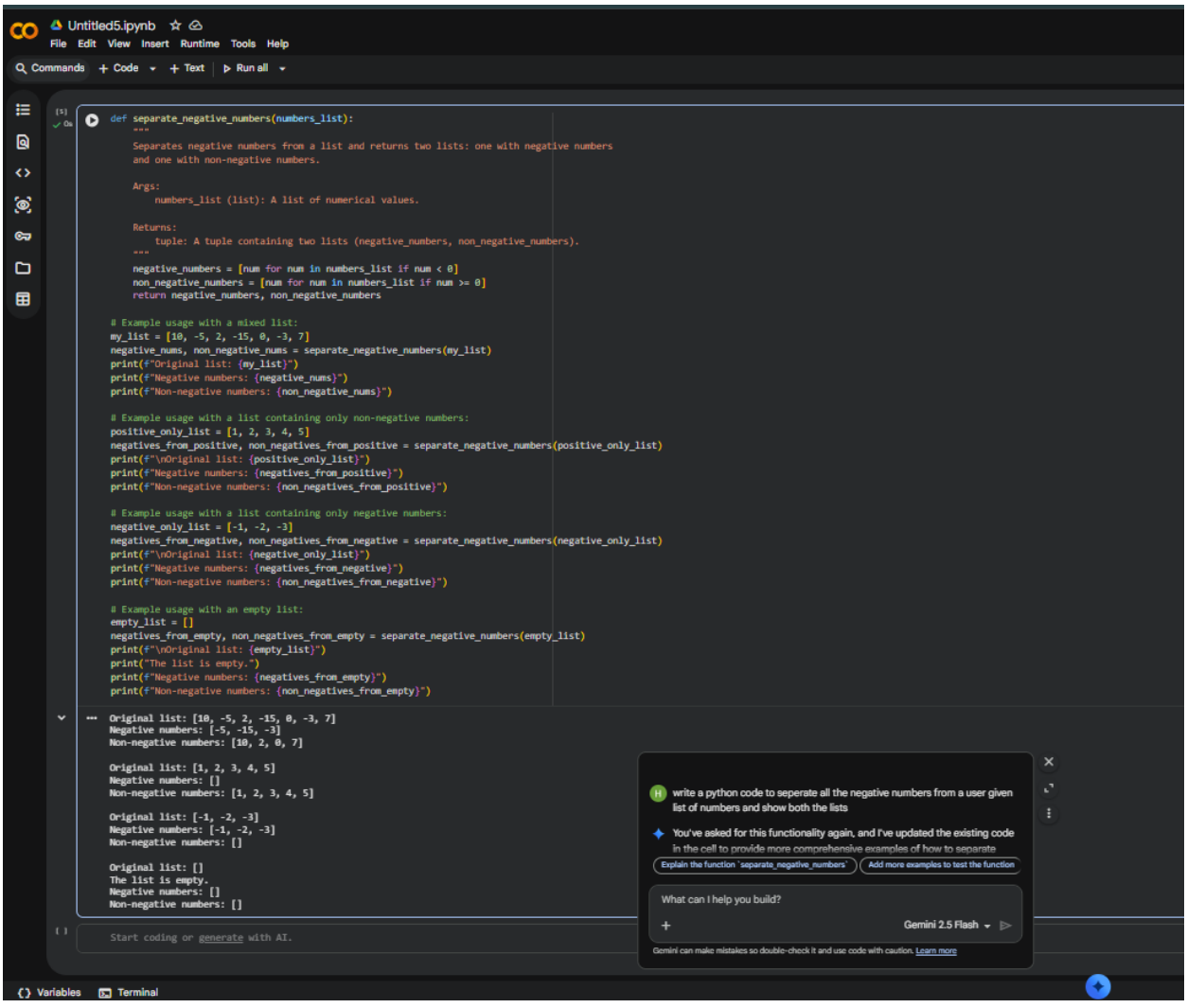
ASSIGNMENT – 2.2

Thumma Hasini | 2303A52076 | Batch – 37

Task 1 : Use Gemini in Colab to generate a function that filters out all negative numbers from a list.

Prompt Used : Write a python code to separate all the negative numbers from a user given list of numbers and show both the lists



Explanation :

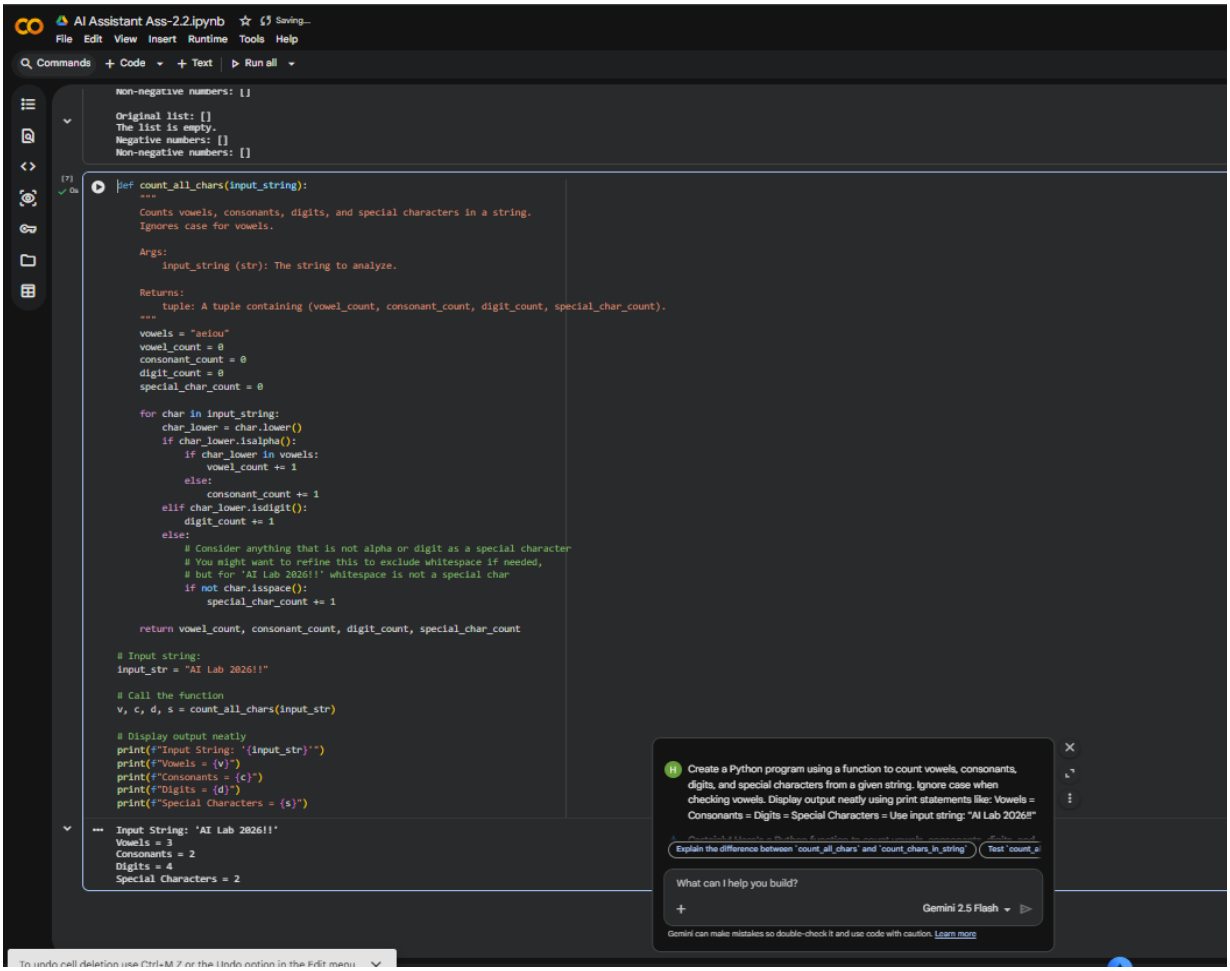
In this task, I removed unwanted negative values from sensor readings to make the data clean and usable. Negative readings can occur due to sensor errors or noise, so filtering them

improves accuracy. I used a simple function to check each value and keep only valid readings. Finally, I displayed the list before and after cleaning to verify the result clearly.

Task 2 : Use Gemini to generate a Python function that counts vowels, consonants,

and digits in a string.

Prompt Used : Create a Python program using a function to count vowels, consonants, digits, and special characters from a given string.
Ignore case when checking vowels.
Display output neatly using print statements like:
Vowels =
Consonants =
Digits =
Special Characters =
Use input string: "AI Lab 2026!!"



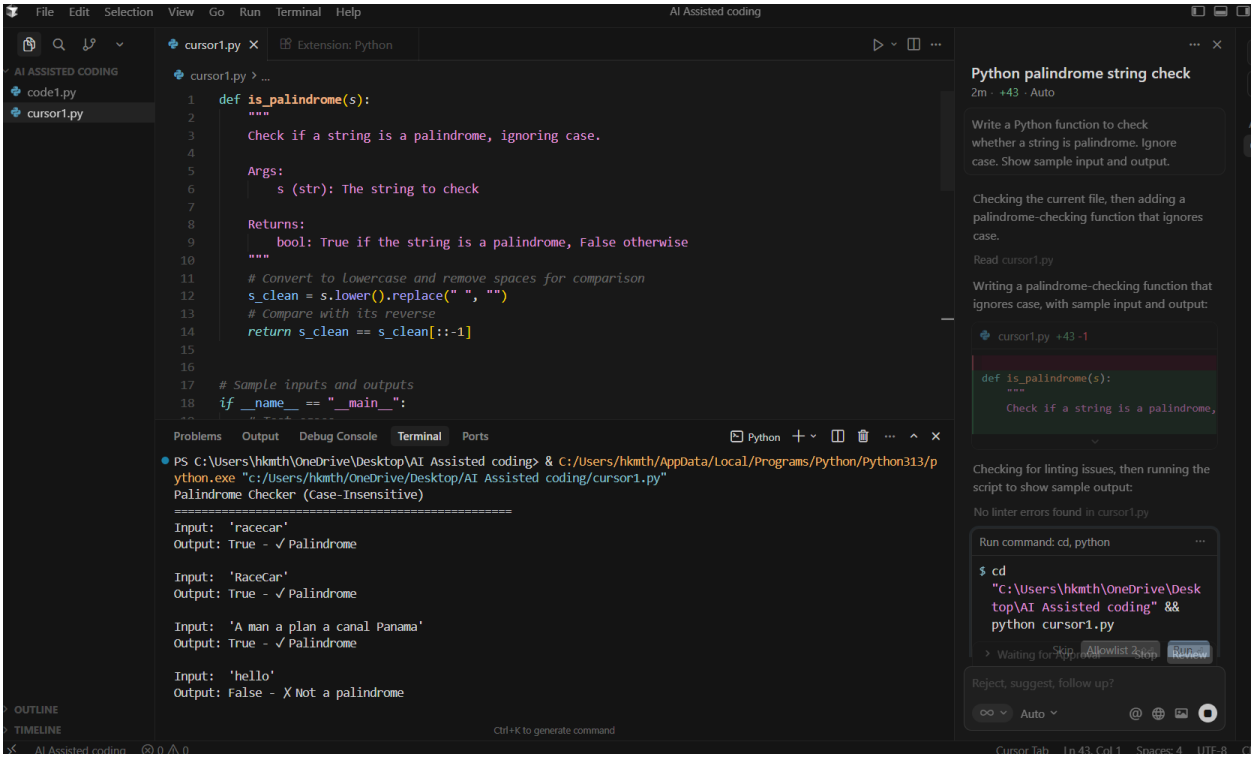
Explanation :

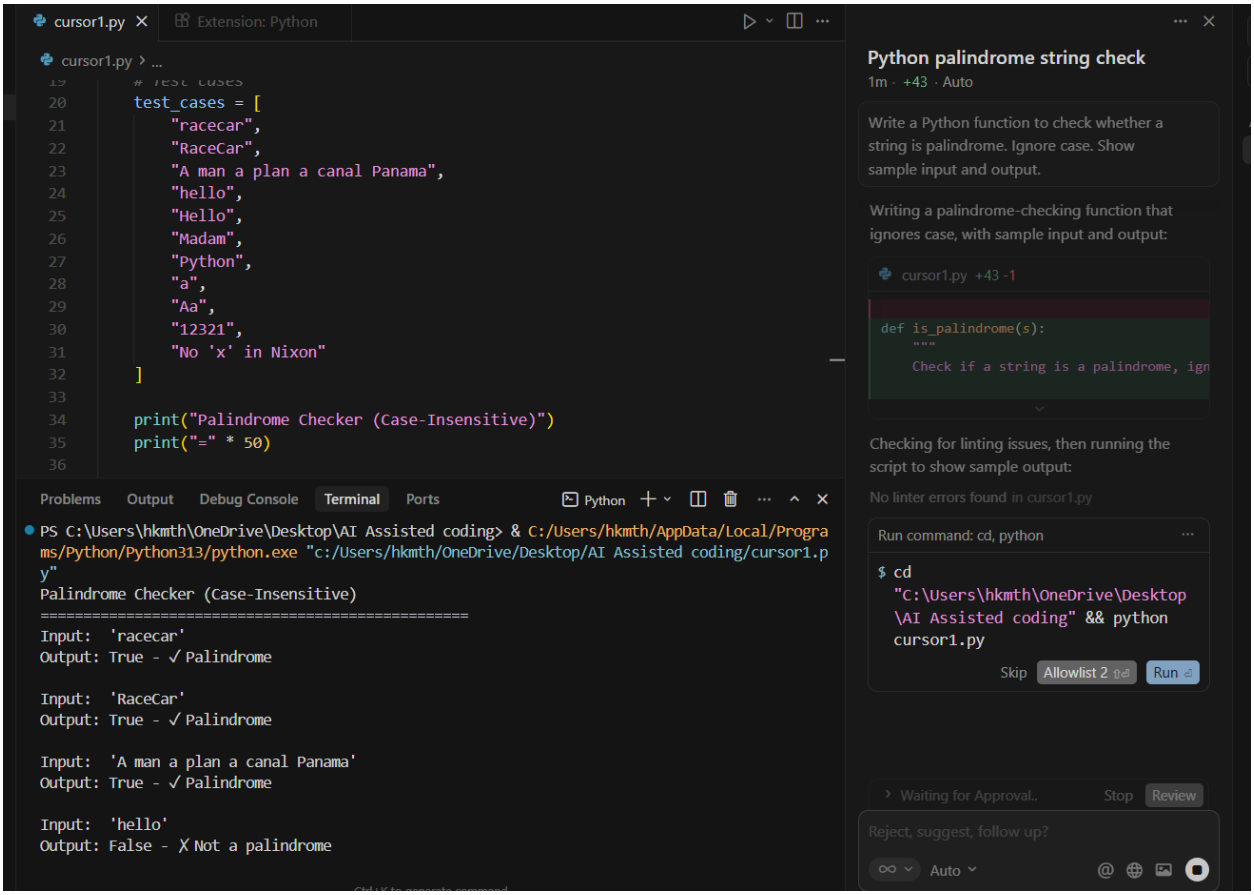
In this task, I analyzed a given string and counted vowels, consonants, and digits separately. This helps in understanding string processing and character classification in Python. I used conditions like isalpha() and isdigit() to differentiate letters and numbers. The output clearly shows how many vowels, consonants, and digits are present in the input text.

Task 3 : Generate a palindrome-checking function using Gemini and Copilot, then

compare the results.

Prompt Used : Write a Python function to check whether a string is palindrome. Ignore case. Show sample input and output.





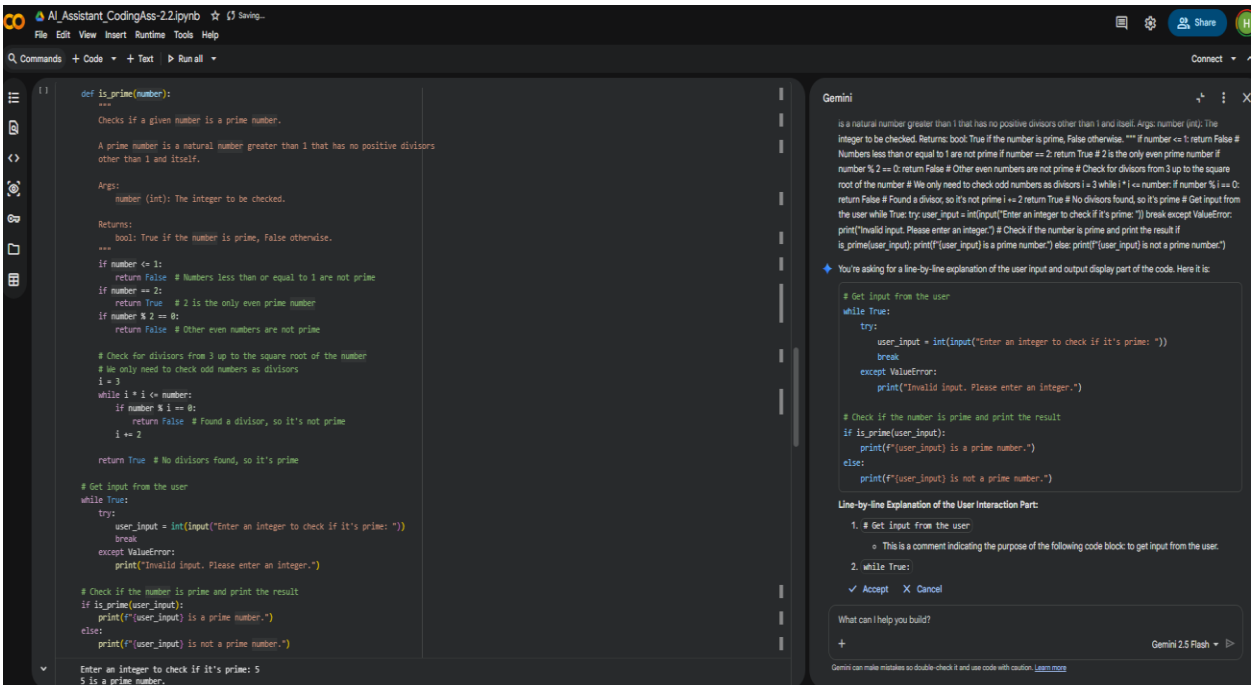
Explanation :

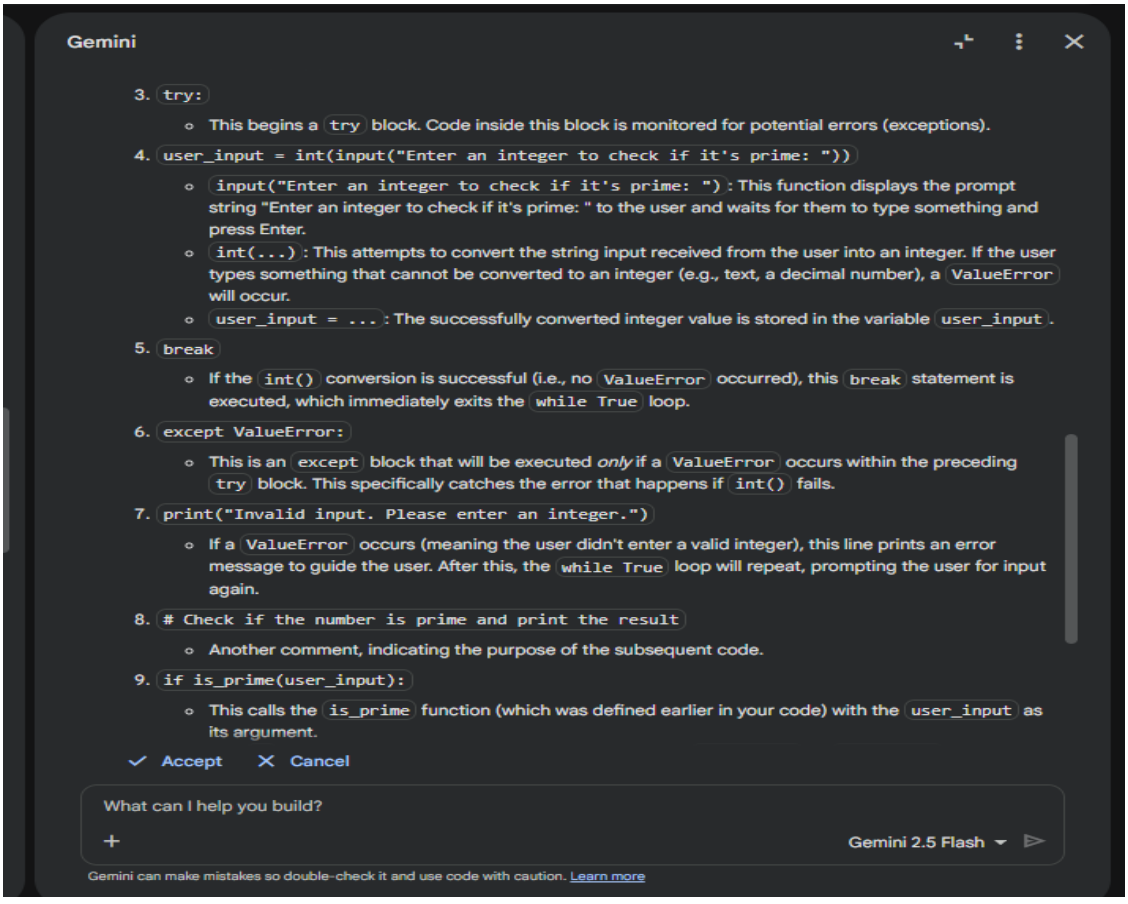
In this task, I used Cursor AI to generate a Python program that checks whether a given string is a palindrome. I entered the prompt in the Cursor AI chat panel and the tool generated the required function automatically. The code was executed in the terminal and tested with multiple inputs such as “racecar” and “hello”. Cursor AI helped in quickly generating code inside the editor and verifying output easily.

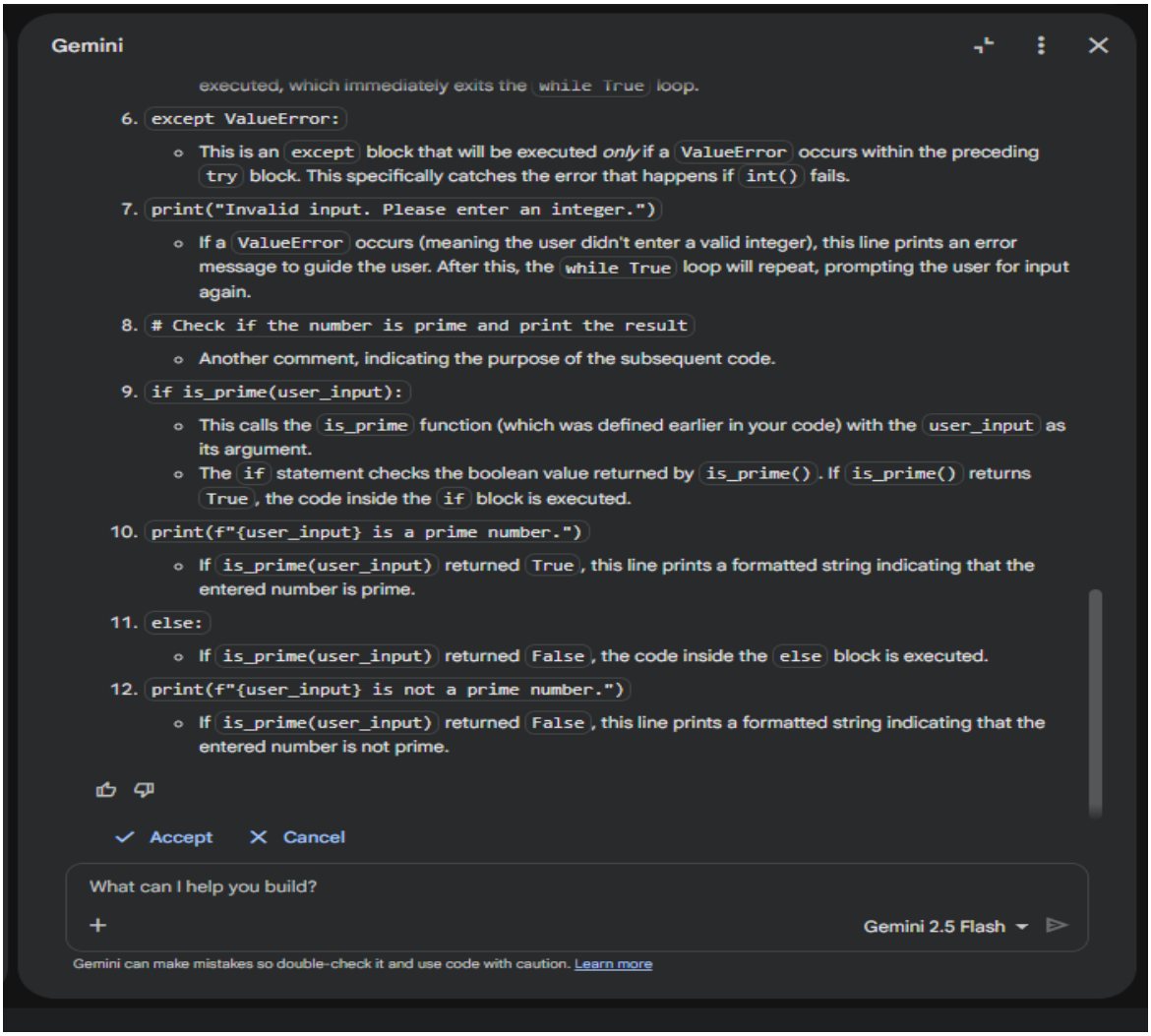
Task 4 : Ask Gemini to explain a Python function (prime check OR palindrome

check) line by line.

Prompt Used : Explain the above prime check code


AI_Assistant_CodingAss-2.2.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Connect
Share
def is_prime(number):
"""
Checks if a given number is a prime number.
A prime number is a natural number greater than 1 that has no positive divisors
other than 1 and itself.
Args:
number (int): The integer to be checked.
Returns:
bool: True if the number is prime, False otherwise.
"""
if number <= 1:
return False # Numbers less than or equal to 1 are not prime
if number == 2:
return True # 2 is the only even prime number
if number % 2 == 0:
return False # Other even numbers are not prime
# Check for divisors from 3 up to the square root of the number
# We only need to check odd numbers as divisors
i = 3
while i * i <= number:
if number % i == 0:
return False # Found a divisor, so it's not prime
i += 2
return True # No divisors found, so it's prime
# Get input from the user
while True:
try:
user_input = int(input("Enter an integer to check if it's prime: "))
break
except ValueError:
print("Invalid input. Please enter an integer.")
# Check if the number is prime and print the result
if is_prime(user_input):
print(f"{user_input} is a prime number.")
else:
print(f"{user_input} is not a prime number.")
Enter an integer to check if it's prime: 5
5 is a prime number.
Gemini
is a natural number greater than 1 that has no positive divisors other than 1 and itself. Args: number (int): The integer to be checked. Returns: bool: True if the number is prime, False otherwise. """ if number <= 1: return False # Numbers less than or equal to 1 are not prime if number == 2: return True # 2 is the only even prime number if number % 2 == 0: return False # Other even numbers are not prime # Check for divisors from 3 up to the square root of the number # We only need to check odd numbers as divisors i = 3 while i * i <= number: if number % i == 0: return False # Found a divisor, so it's not prime i += 2 return True # No divisors found, so it's prime # Get input from the user while True: try: user_input = int(input("Enter an integer to check if it's prime: ")) break except ValueError: print("Invalid input. Please enter an integer.") # Check if the number is prime and print the result if is_prime(user_input): print(f"{user_input} is a prime number.") else: print(f"{user_input} is not a prime number.")
You're asking for a line-by-line explanation of the user input and output display part of the code. Here it is:
# Get input from the user
while True:
try:
user_input = int(input("Enter an integer to check if it's prime: "))
break
except ValueError:
print("Invalid input. Please enter an integer.")
# Check if the number is prime and print the result
if is_prime(user_input):
print(f"{user_input} is a prime number.")
else:
print(f"{user_input} is not a prime number.")
Line-by-line Explanation of the User Interaction Part:
1. # Get input from the user
This is a comment indicating the purpose of the following code block: to get input from the user.
2. while True:
Accept
Cancel
What can I help you build?
Gemini 2.5 Flash
Gemini can make mistakes so double-check it and use code with caution. Learn more


Gemini
3. try:
This begins a try block. Code inside this block is monitored for potential errors (exceptions).
4. user_input = int(input("Enter an integer to check if it's prime: "))
input("Enter an integer to check if it's prime: "): This function displays the prompt string "Enter an integer to check if it's prime: " to the user and waits for them to type something and press Enter.
int(...): This attempts to convert the string input received from the user into an integer. If the user types something that cannot be converted to an integer (e.g., text, a decimal number), a ValueError will occur.
user_input = ...: The successfully converted integer value is stored in the variable user_input.
5. break
If the int() conversion is successful (i.e., no ValueError occurred), this break statement is executed, which immediately exits the while True loop.
6. except ValueError:
This is an except block that will be executed only if a ValueError occurs within the preceding try block. This specifically catches the error that happens if int() fails.
7. print("Invalid input. Please enter an integer.")
If a ValueError occurs (meaning the user didn't enter a valid integer), this line prints an error message to guide the user. After this, the while True loop will repeat, prompting the user for input again.
8. # Check if the number is prime and print the result
Another comment, indicating the purpose of the subsequent code.
9. if is_prime(user_input):
This calls the is_prime function (which was defined earlier in your code) with the user_input as its argument.
Accept
Cancel
What can I help you build?
Gemini 2.5 Flash
Gemini can make mistakes so double-check it and use code with caution. Learn more



Explanation :

In this task, I used Gemini to explain the generated code line-by-line in a simple way. This helped me understand what each line does and how the logic works internally. Gemini also explained why checking divisors up to √n improves performance. By reading the explanation, I learned how time complexity is reduced and how code can be optimized for efficiency.