

ASSIGNMENT-8

NAME: K.Bharath

HT.NO: 2403A52065

BATCH: AIB03

Task-1

Write a test case to check if a function returns the square of a number. Then write the function with help from GitHub Copilot or Cursor AI.

Prompt:

$1^{**}2=1, 2^{**}2=4, 3^{**}2=9, 4^{**}2=16, 5^{**}2=25, 6^{**}2=36, 7^{**}2=49, 8^{**}2=64, 9^{**}2=81, 10^{**}2=100$. according to the test case given to you, write a function to generate square of a given number. input is given dynamically

Code:

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files like 1-2.py, task(1).py, task(2).py, task(3).py, task(4).py, task(5).py, and task(1).py.
- Variables:** Shows a variable named "Name" with the value "Alice" and "Age" with the value "25".
- Code Editor:** Displays the following Python code:

```
8th lab > task(1).py > ...
1 # Function to calculate the square of a given number
2 def calculate_square(number):
3     return number ** 2
4
5 # Test the function with dynamic input
6 if __name__ == "__main__":
7     num = int(input("Enter a number to find its square: "))
8     print(f"The square of {num} is {calculate_square(num)}")
```
- Terminal:** Shows the command PS C:\Users\tharu\OneDrive\Desktop\AI CODING> & 'c:\Program Files\Python311\python.exe' 'c:\Users\tharu\.vscode\extensions\ms-python.debugger-2025.10.0-win32-x64\bundled\libs\debugger\launcher' '53052' '--' 'C:\Users\tharu\Desktop\AI CODING\8th lab\task(1).py'
- Chat Panel:** Shows a message from Alice: "Name: Alice, Age: 25". Below it, a message from the AI: "12=1,22=4,32=9,42=16,52=25,62=36,72=49,82=64,92=81,102=100, according to the test case given to you, write a function to generate square of a given number. input is given dynamically".
- Code Snippet:** Shows a snippet of code: "# Function to calculate the square of a given number def calculate_square(number): return number ** 2".

OP:

The screenshot shows the VS Code interface with the following details:

- Terminal:** Shows the command PS C:\Users\tharu\OneDrive\Desktop\AI CODING> & 'c:\Program Files\Python311\python.exe' 'c:\Users\tharu\.vscode\extensions\ms-python.debugger-2025.10.0-win32-x64\bundled\libs\debugger\launcher' '53052' '--' 'C:\Users\tharu\Desktop\AI CODING\8th lab\task(1).py'. The output includes "Enter a number to find its square: 3" and "The square of 3 is 9".
- Code Editor:** Shows the same Python code as in the previous screenshot.
- Breakpoints:** Shows a list of breakpoints: "Raised Exceptions", "Uncought Excep...", and "User Uncaught E...".
- Status Bar:** Shows "Ln 8, Col 61 Spaces: 4 UTF-8 3.11.5 Go Live".

Observation:

The calculate_square function takes a number as input and returns its square using the `**` operator. I have prompted to input a number dynamically. The program calculates the square of the input number and prints the result.

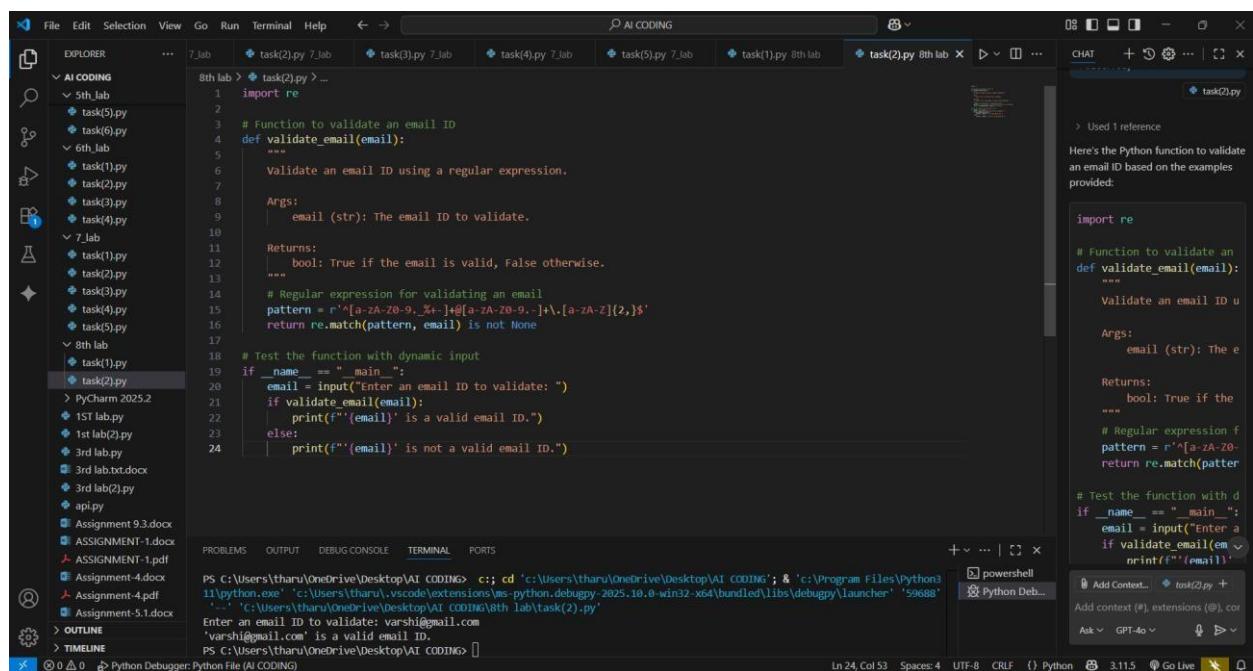
Task-2:

Create test cases to validate an email address (e.g., contains @ and .com). Use AI assistance to implement the validate_email() function

Prompt:

write a function which validates a email id. examples of a mail id is containt@gmail.com, madam@gmail.com, anyone@gmail.com. according to the test case/examples given generate code.

Code:



The screenshot shows the PyCharm IDE interface with the following details:

- File Structure (EXPLORER):** Shows a project structure with folders like AI CODING, Sth_lab, task(1).py, task(2).py, task(3).py, task(4).py, task(5).py, 7th_lab, task(1).py, task(2).py, task(3).py, task(4).py, task(5).py, 8th_lab, task(1).py, task(2).py, task(3).py, task(4).py, task(5).py, task(6).py, and task(7).py.
- Code Editor (7th_lab/task(2).py):**

```

1 import re
2
3 # Function to validate an email ID
4 def validate_email(email):
5     """
6         Validate an email ID using a regular expression.
7
8     Args:
9         email (str): The email ID to validate.
10
11    Returns:
12        bool: True if the email is valid, False otherwise.
13    """
14        # Regular expression for validating an email
15        pattern = r'^[a-zA-Z0-9.-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
16        return re.match(pattern, email) is not None
17
18    # Test the function with dynamic input
19    if __name__ == "__main__":
20        email = input("Enter an email ID to validate: ")
21        if validate_email(email):
22            print(f"{email} is a valid email ID.")
23        else:
24            print(f"{email} is not a valid email ID.")

```
- Terminal (bottom):**

```

PS C:\Users\tharu\OneDrive\Desktop\AI CODING> cd 'c:\Users\tharu\OneDrive\Desktop\AI CODING'; & 'c:\Program Files\Python311\python.exe' 'c:\Users\tharu\vscode\extensions\ms-python.python\2025.10.0-win32-x64\bundled\libs\debugpp\launcher' '59688'
...
C:\Users\tharu\OneDrive\Desktop\AI CODING\8th_lab\task(2).py
Enter an email ID to validate: varshigmail.com
'varshigmail.com' is a valid email ID.
PS C:\Users\tharu\OneDrive\Desktop\AI CODING>

```

OP:

```
PS C:\Users\tharu\OneDrive\Desktop\AI CODING> c; cd 'c:\Users\tharu\OneDrive\Desktop\AI CODING'; & 'c:\Program Files\Python311\python.exe' 'c:\Users\tharu\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '59688'
...> C:\Users\tharu\OneDrive\Desktop\AI CODING>8th lab> task2.py
Enter an email ID to validate: varshith@gmail.com
'varshith@gmail.com' is a valid email ID.
PS C:\Users\tharu\OneDrive\Desktop\AI CODING>
```

```
email (str): The e
Returns:
    bool: True if the
    ...
# Regular expression f
pattern = r'^[a-zA-Z0-
return re.match(patter
# Test the function with d
if __name__ == "__main__":
    email = input("Enter a
    if validate_email(em
        print(f'{email} is v
    else:
        print(f'{email} is n
    print("Email validation completed")
```

Observation:

Examples like containt@gmail.com, madam@gmail.com, and anyone@gmail.com are valid because, They contain alphanumeric characters before the @. They have a domain name after the @ (e.g., gmail.com). The domain name ends with a valid top-level domain (e.g., .com). Emails without an @ symbol or domain name are invalid. Emails with special characters in invalid positions (e.g., @example.com or name@.com) are invalid.

Task-3:

Write test cases for a function that returns the maximum of three numbers. Prompt Copilot/Cursor to write the logic based on tests

Prompt:

in 13,54,98 98 is the maximum of three numbers. 1,2,3 3 is the maximum of three numbers. now generate a function that returns the maximum of three numbers, according to the test case given. take input dynamically Code:

The screenshot shows the PyCharm IDE interface with the 'AI CODING' feature open. The left sidebar shows a project structure with several files under '8th lab'. The main editor window displays a Python script named 'task(3).py' with the following code:

```

# Function to find the maximum of three numbers
def find_maximum(a, b, c):
    """
    Find the maximum of three numbers.

    Args:
        a (int): The first number.
        b (int): The second number.
        c (int): The third number.

    Returns:
        int: The maximum of the three numbers.
    """
    return max(a, b, c)

# Test the function with dynamic input
if __name__ == "__main__":
    print("Enter three numbers:")
    num1 = int(input("Enter the first number: "))
    num2 = int(input("Enter the second number: "))
    num3 = int(input("Enter the third number: "))

    maximum = find_maximum(num1, num2, num3)
    print(f"The maximum of {num1}, {num2}, and {num3} is {maximum}.")

```

The right panel shows the AI-generated code and its explanation:

in 13,54,98 98 is the maximum of three numbers. 1,2,3 is the maximum of three numbers. now generate a function that returns the maximum of three numbers, according to the test case given. take input dynamically

Used 1 reference

Here's the Python function to find the maximum of three numbers based on the test cases provided:

```

# Function to find the max
def find_maximum(a, b, c):
    """
    Find the maximum of th

    Args:
        a (int): The first
        b (int): The secon
        c (int): The third

    Returns:
        int: The maximum o
    """
    return max(a, b, c)

```

Add Context... task(3).py + Add context (#), extensions (@), cor Ask v GPT-4o v Go Live

OP:

The screenshot shows the PyCharm IDE interface with the 'AI CODING' feature open. The left sidebar shows a project structure with several files under '3rd lab'. The main editor window displays a Python script named 'task(3).py' with the following code:

```

# Function to find the maximum of three numbers
def find_maximum(a, b, c):
    """
    Find the maximum of three numbers.

    Args:
        a (int): The first number.
        b (int): The second number.
        c (int): The third number.

    Returns:
        int: The maximum of the three numbers.
    """
    return max(a, b, c)

# Test the function with dynamic input
if __name__ == "__main__":
    print("Enter three numbers:")
    Enter the first number: 87
    Enter the second number: 99
    Enter the third number: 99
    The maximum of 87, 99, and 99 is 99.

```

The right panel shows the AI-generated code and its explanation:

0 (int): The second
c (int): The third
Returns:
int: The maximum o
return max(a, b, c)

Add Context... task(3).py + Add context (#), extensions (@), cor Ask v GPT-4o v Go Live

Observation:

The `find_maximum` function takes three numbers as arguments and returns the maximum using Python's built-in `max()` function. I have prompted to input three numbers dynamically. The program calculates the maximum of the three numbers and prints the result.

Task-4:

Use TDD to write a shopping cart class with methods to add, remove, and get total price. First write tests for each method, then generate code using AI.

Prompt: Now generate a Python class `ShoppingCart` that can add items, remove items, and calculate the total price according to the given test

cases. Take input dynamically from the user. Input: Add Apple with quantity 2 and price 3.0 → Output: {"Apple": {"quantity": 2, "price": 3.0}}. Input: Remove Apple from the cart → Output: {}. Input: Add Apple (quantity 2, price 3.0) and Banana (quantity 1, price 1.5) → Output:

Total price is 7.5.

Code:

File Edit Selection View Go Run Terminal Help

AI CODING

EXPLORER

```

8th lab > task4.py 7.lab task5.py 7.lab task1.py 8th lab task2.py 8th lab task3.py 8th lab task4.py 8th lab
  1 class ShoppingCart:
  2     def __init__(self):
  3         self.items = {}
  4     def add_item(self, name, quantity, price):
  5         if name in self.items:
  6             self.items[name]["quantity"] += quantity
  7         else:
  8             self.items[name] = {"quantity": quantity, "price": price}
  9     def remove_item(self, name):
 10         if name in self.items:
 11             del self.items[name]
 12         else:
 13             print(f"Item '{name}' not found in the cart.")
 14     def get_total_price(self):
 15         total = 0
 16         for item in self.items.values():
 17             total += item["quantity"] * item["price"]
 18         return total
 19 if __name__ == "__main__":
 20     cart = ShoppingCart()
 21     while True:
 22         print("In Shopping Cart Menu:")
 23         print("1. Add Item")
 24         print("2. Remove Item")
 25         print("3. View Total Price")
 26         print("4. View Cart")
 27         print("5. Exit")
 28         choice = input("Enter your choice: ")
 29         if choice == "1":
 30             name = input("Enter the item name: ")
 31             quantity = int(input("Enter the quantity: "))
 32             price = float(input("Enter the price per unit: "))
 33             cart.add_item(name, quantity, price)
 34         elif choice == "2":
 35             name = input("Enter the item name to remove: ")
 36             cart.remove_item(name)
 37         elif choice == "3":
 38             total_price = cart.get_total_price()
 39             print(f"The total price of items in the cart is: {total_price}")
 40         elif choice == "4":
 41             print("Current items in the cart:")
 42             for name, details in cart.items.items():
 43                 print(f"Name: {name}, Quantity: {details['quantity']}, Price: {details['price']} ")
 44         elif choice == "5":
 45             print("Exiting the shopping cart. Thank you!")
 46             break
 47         else:
 48             print("Invalid choice. Please try again.")
 49

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Exiting the shopping cart. Thank you!

PS C:\Users\tharu\OneDrive\Desktop\AI CODING>

Ln 47, Col 18 Spaces: 4 UTF-8 CRLF () Python 3.11.5 Go Live

File Edit Selection View Go Run Terminal Help

AI CODING

EXPLORER

```

8th lab > task4.py 7.lab task5.py 7.lab task1.py 8th lab task2.py 8th lab task3.py 8th lab task4.py 8th lab
  1 print("1. Add Item")
  2 print("2. Remove Item")
  3 print("3. View Total Price")
  4 print("4. View Cart")
  5 print("5. Exit")
  6 choice = input("Enter your choice: ")
  7 if choice == "1":
  8     name = input("Enter the item name: ")
  9     quantity = int(input("Enter the quantity: "))
 10     price = float(input("Enter the price per unit: "))
 11     cart.add_item(name, quantity, price)
 12     print(f"Added {quantity} of {name} to the cart.")
 13 elif choice == "2":
 14     name = input("Enter the item name to remove: ")
 15     cart.remove_item(name)
 16 elif choice == "3":
 17     total_price = cart.get_total_price()
 18     print(f"The total price of items in the cart is: {total_price}")
 19 elif choice == "4":
 20     print("Current items in the cart:")
 21     for name, details in cart.items.items():
 22         print(f"Name: {name}, Quantity: {details['quantity']}, Price: {details['price']} ")
 23 elif choice == "5":
 24     print("Exiting the shopping cart. Thank you!")
 25     break
 26 else:
 27     print("Invalid choice. Please try again.")
 28

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Exiting the shopping cart. Thank you!

PS C:\Users\tharu\OneDrive\Desktop\AI CODING>

Ln 47, Col 18 Spaces: 4 UTF-8 CRLF () Python 3.11.5 Go Live

OP:

```

task4.py
task1.py
task2.py
task3.py
task4.py
task5.py
7_lab
task1.py
task2.py
task3.py
task4.py
task5.py
8th_lab
task1.py
task2.py
task3.py
task4.py
task5.py
PyCharm 2025.2
1st_lab.py
1st_lab2.py
3rd_lab.py
3rd_lab2.py
api.py
Assignment 9.3.docx
ASSIGNMENT-1.docx
ASSIGNMENT-1.pdf
Assignment-4.docx
Assignment-4.pdf
Assignment-5.pdf
Assignment-5.pdf
OUTLINE
TIMELINE
Python Debugger: Python File (AI CODING)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
POWERFUL View LOCAL PROJECT
powerhell
Python Deb...
input dynamically

Here's the ShoppingCart program with dynamic input for adding, removing items, and calculating the total price:

class ShoppingCart:
    def __init__(self):
        self.items = {}

    def add_item(self, name):
        if name in self.items:
            self.items[name] += 1
        else:
            self.items[name] = 1

    def remove_item(self, name):
        if name in self.items:
            del self.items[name]
        else:
            print(f"Item {name} not found in cart.")

    def view_total_price(self):
        total = 0
        for item, quantity in self.items.items():
            total += item.price * quantity
        return total

    def view_cart(self):
        print("Cart Contents:")
        for item, quantity in self.items.items():
            print(f"{item.name} x {quantity} - ${item.price * quantity:.2f}")

    def exit(self):
        print("Exiting the shopping cart. Thank you!")
        exit()

    def main_menu(self):
        print("Shopping Cart Menu:")
        print("1. Add Item")
        print("2. Remove Item")
        print("3. View Total Price")
        print("4. View Cart")
        print("5. Exit")

    def get_user_choice(self):
        choice = input("Enter your choice: ")
        return choice

    def run(self):
        while True:
            self.main_menu()
            choice = self.get_user_choice()
            if choice == "1":
                name = input("Enter the item name: ")
                quantity = int(input("Enter the quantity: "))
                self.add_item(name)
            elif choice == "2":
                name = input("Enter the item name: ")
                self.remove_item(name)
            elif choice == "3":
                print(f"Total Price: ${self.view_total_price():.2f}")
            elif choice == "4":
                self.view_cart()
            elif choice == "5":
                self.exit()
            else:
                print("Invalid choice. Please enter a number between 1 and 5.")


if __name__ == "__main__":
    ShoppingCart().run()

```

PS C:\Users\tharu\OneDrive\Desktop\AI CODING> [REDACTED]

Observation: The program uses `input()` to allow the user to interact with the shopping cart dynamically. Users can add items, remove items, view the total price, and see the cart's contents.

- **Option 1:** Add an item to the cart. ○ **Option 2:** Remove an item from the cart.
- **Option 3:** Calculate and display the total price of items in the cart.
- **Option 4:** Display all items in the cart.
- **Option 5:** Exit the program.

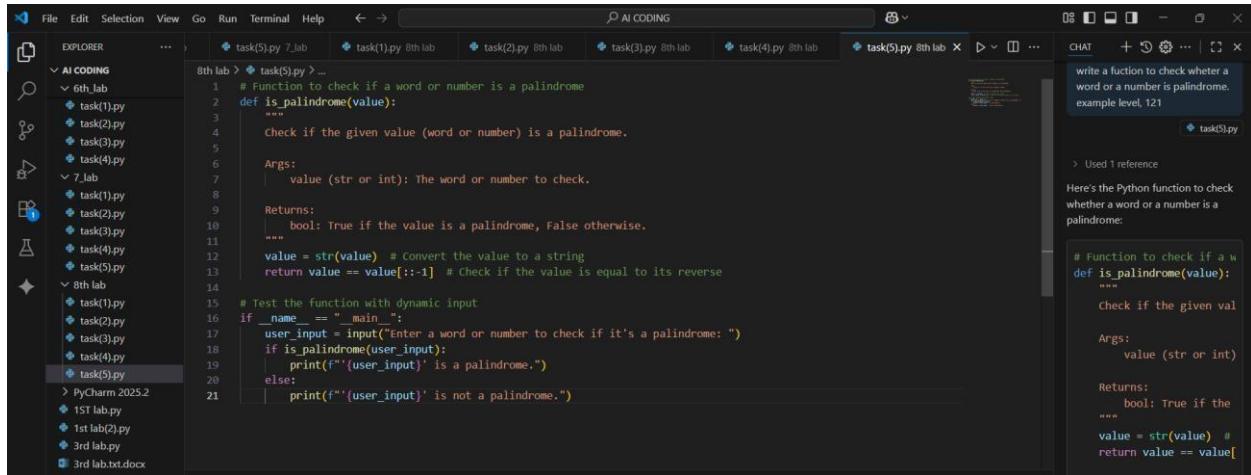
If the user tries to remove an item that doesn't exist, the program displays an appropriate message

Task-5:

Write tests for a palindrome checker (e.g., `is_palindrome("level")` → True). Let Copilot suggest the function based on test case expectations.

Prompt:

write a function to check whether a word or a number is palindrome or not. Example: level, 121 Code:



The screenshot shows the PyCharm IDE interface. In the top right, there's an "AI CODING" button. A tooltip from this button says: "write a function to check whether a word or a number is palindrome. example level, 121". Below the tooltip, the code editor shows a Python file named `task(5).py` with the following content:

```
# Function to check if a word or number is a palindrome
def is_palindrome(value):
    """
    Check if the given value (word or number) is a palindrome.

    Args:
        value (str or int): The word or number to check.

    Returns:
        bool: True if the value is a palindrome, False otherwise.
    """
    value = str(value) # Convert the value to a string
    return value == value[::-1] # Check if the value is equal to its reverse

# Test the function with dynamic input
if __name__ == "__main__":
    user_input = input("Enter a word or number to check if it's a palindrome: ")
    if is_palindrome(user_input):
        print(f"{user_input} is a palindrome.")
    else:
        print(f"{user_input} is not a palindrome.")
```

OP:



The screenshot shows the PyCharm interface with the terminal tab active. The terminal window displays the following command and output:

```
'-- C:\Users\tharu\OneDrive\Desktop\AI CODING\8th lab\task(5).py'
Enter a word or number to check if it's a palindrome: 141
'141' is a palindrome.
PS C:\Users\tharu\OneDrive\Desktop\AI CODING>
```

Observation:

The input value is converted to a string using `str(value)` to handle both words and numbers. The function checks if the string is equal to its reverse using slicing (`value[::-1]`). I have prompted to enter a word or number dynamically. The program prints whether the input is a palindrome

