

SR UNIVERSITY

AI ASSIST CODING

Lab-3.2

NAME:HASINI BEGALA

2503A51L13

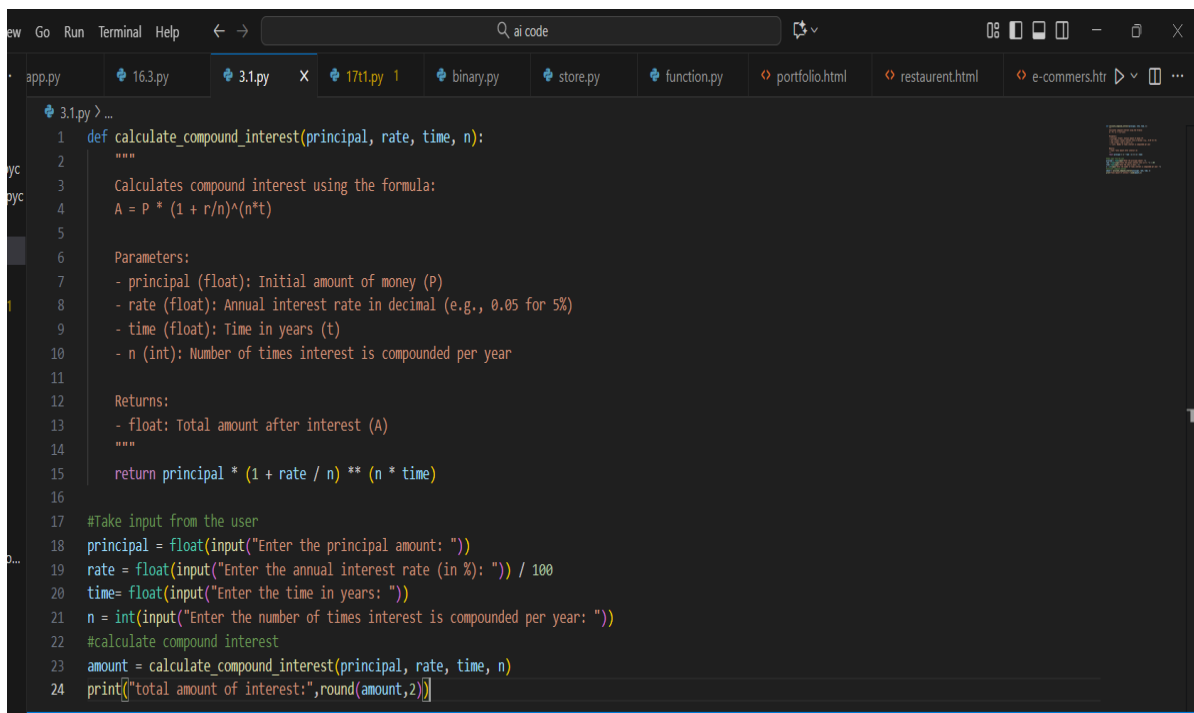
Batch:19

TASK #1:

Prompt Used:

Ask AI to write a function to calculate compound interest, starting with only the function name. Then add a docstring, then input-output example.

Code generated:

A screenshot of a code editor window with a dark theme. The editor shows a Python script named '3.1.py'. The code defines a function 'calculate_compound_interest' with parameters 'principal', 'rate', 'time', and 'n'. It includes a docstring explaining the formula $A = P * (1 + r/n)^{(n*t)}$ and lists parameters and returns. Below the function, there is a section for taking user input for each parameter and then calling the function to calculate the amount, which is printed with two decimal places. The code is as follows:

```
1 def calculate_compound_interest(principal, rate, time, n):
2     """
3     Calculates compound interest using the formula:
4     A = P * (1 + r/n)^(n*t)
5
6     Parameters:
7     - principal (float): Initial amount of money (P)
8     - rate (float): Annual interest rate in decimal (e.g., 0.05 for 5%)
9     - time (float): Time in years (t)
10    - n (int): Number of times interest is compounded per year
11
12    Returns:
13    - float: Total amount after interest (A)
14    """
15    return principal * (1 + rate / n) ** (n * time)
16
17 #Take input from the user
18 principal = float(input("Enter the principal amount: "))
19 rate = float(input("Enter the annual interest rate (in %): ")) / 100
20 time = float(input("Enter the time in years: "))
21 n = int(input("Enter the number of times interest is compounded per year: "))
22 #calculate compound interest
23 amount = calculate_compound_interest(principal, rate, time, n)
24 print("total amount of interest:",round(amount,2))
```

Output After executing Code:

```
y"
Total amount after 3 years: $1160.75
PS C:\Users\HASINI\OneDrive\Desktop\ai code> & "C:/Users/HASINI/OneDrive/Desktop/ai code/venv/Scripts/python.exe" "c:/Users/HASINI/OneDrive/Desktop/ai code/venv/Scripts/python.exe" "c:/Users/HASINI/OneDrive/Desktop/ai code/venv/Scripts/python.exe"
y"
Enter the principal amount:
```

Observations:

- The code correctly implements the compound interest formula using inputs (principal, rate, time, compounds per year) and returns the calculated final amount.
- The docstring explains the function with parameter details, return type, and an example, making the code clear and professional.

TASK #2:

Prompt Used:

Do math stuff, then refine it to: # Write a function to calculate average, median, and mode of a list of numbers.

Code Generated:

The screenshot shows a Windows desktop with a VS Code editor window open. The editor displays a Python script named `3.1.py` that calculates the average, median, and mode of a list of numbers. The script includes error handling for empty lists and non-numeric input. The Windows taskbar at the bottom shows various icons, including the Start button, search bar, and several application icons. The system clock in the bottom right corner indicates the date is 27-10-2022 and the time is 13:35.

```

File Edit Selection View Go Run Terminal Help
ai code
app.py 16.3.py 3.1.py x 1711.py 1 binary.py store.py function.py portfolio.html restaurant.html e-commers.html fest.html
3.1.py > -
1 from statistics import mean, median, mode, StatisticsError
2
3 def calculate_stats(numbers):
4     if not numbers:
5         return "the list is empty"
6     avg = mean(numbers)
7     med = median(numbers)
8     try:
9         mod = mode(numbers)
10    except StatisticsError:
11        mod = "no unique mode"
12    return {
13        "average": avg,
14        "median": med,
15        "mode": mod
16    }
17
18 if __name__ == "__main__":
19     user_input = input("Enter numbers separated by commas: ")
20     try:
21         number_list = [float(num.strip()) for num in user_input.split(',') if num.strip() != ""]
22         results = calculate_stats(number_list)
23         print("\nStatistics:")
24         if isinstance(results, dict):
25             for key, value in results.items():
26                 print(f"{key.capitalize()}: {value}")
27         else:
28             print(results)
29     except ValueError:
30         print("Invalid input. Please enter numeric values separated by commas.")
  
```

Ln 30, Col 81 Spaces: 4 UTF-8 CRLF Python 3.13.7 (venv) Go Live

Trending videos
A Knight of the...

Search

13:35
27-10-2022

Output After executing Code:

```
File "c:\Users\HASINI\OneDrive\Desktop\ai code\3.1.py", line 18, in <module>
    median = sorted_nums[n // 2]
3.1.py"
PS C:\Users\HASINI\OneDrive\Desktop\ai code> & "C:/Users/HASINI/OneDrive/Desktop/ai code/venv/Scripts/pyth
Enter numbers separated by commas: 4,5,6,8

Statistics:
Average: 5.75
Median: 5.5
3.1.py"
PS C:\Users\HASINI\OneDrive\Desktop\ai code> & "C:/Users/HASINI/OneDrive/Desktop/ai code/venv/Scripts/pyth
```

Observations:

- The program defines a function to calculate the average, median, and mode of a list of numbers using Python's built-in statistics functions: `mean()`, `median()`, and `mode()`.
- A try-except block handles cases where there is no unique mode, preventing errors.
- User input is taken as a space-separated string, converted into floats, and results are displayed in a structured dictionary format.

TASK #3:

Prompt Used:

Provide multiple examples of input-output to the AI for `convert_to_binary(num)` function. Observe how AI uses few-shot prompting to generalize.

Code Generated:



```
1 def convert_to_binary(num):
2     return bin(num)[2:]
3
4 try:
5     user_input = int(input("enter a decimal number to convert to binary:"))
6     binary_result=convert_to_binary(user_input)
7     print(f"The binary representation of {user_input} is {binary_result}")
8 except ValueError:
9     print("Invalid input. Please enter a valid integer.")
```

Output After executing Code:

```
PS C:\Users\HASINI\OneDrive\Desktop\ai code> & "C:/Users/HASINI/OneDrive/Desktop/ai code/venv/Scripts/python.exe" "c:/Users/HASINI/OneDrive/Desktop/ai code/3.1.py"
enter a decimal number to convert to binary:10000011110
The binary representation of 10000011110 is 1001010100000011000000111101100110
PS C:\Users\HASINI\OneDrive\Desktop\ai code> |
```

Observations:

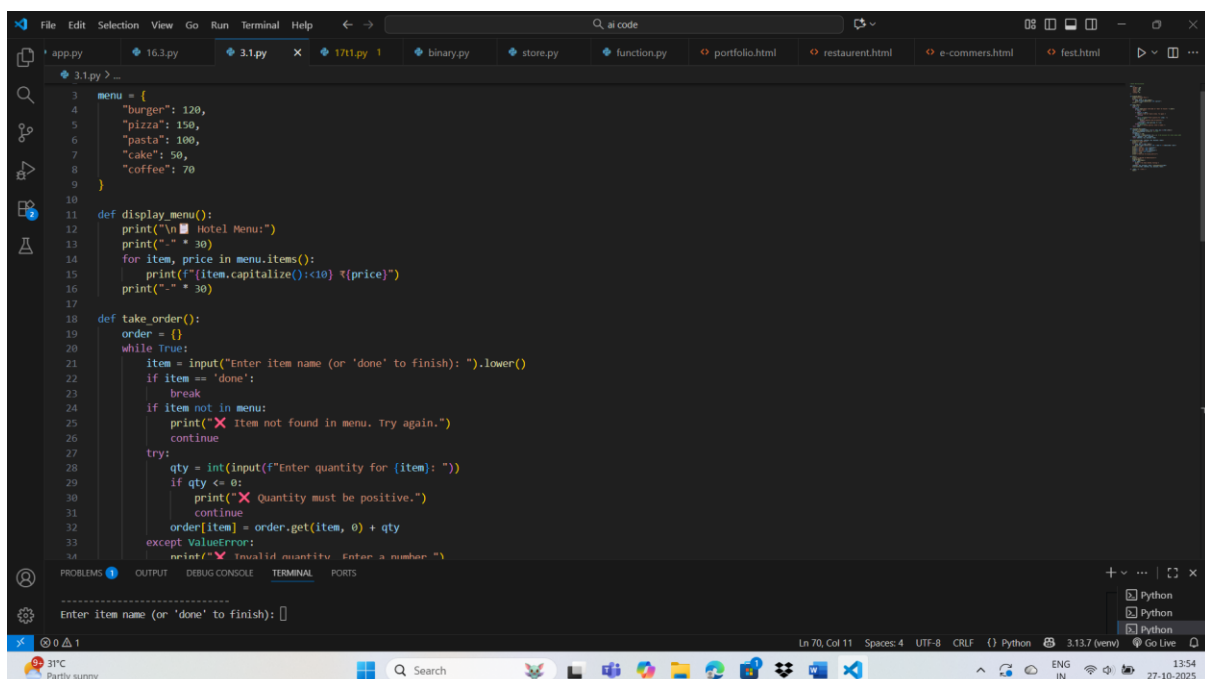
- bin(num) converts a decimal number to a binary string prefixed with '0b'.
- Using [2:] slices off the '0b' to return only the binary digits.
- The function uses a try-except block to check whether the user's input is a valid number and to handle errors if it is not.

TASK #4:

Prompt Used:

Create an user interface for an hotel to generate bill based on customer requirements

Code Generated



```
File Edit Selection View Go Run Terminal Help
3.1.py 16.3.py 3.1.py 1711.py 1 binary.py store.py function.py portfolio.html restaurant.html e-commers.html fest.html

3 menu = {
4     "burger": 120,
5     "pizza": 150,
6     "pasta": 100,
7     "cake": 50,
8     "coffee": 70
9 }
10
11 def display_menu():
12     print("\n Hotel Menu:")
13     print("-" * 30)
14     for item, price in menu.items():
15         print(f"{item.capitalize():<10} ₹{price}")
16     print("-" * 30)
17
18 def take_order():
19     order = {}
20     while True:
21         item = input("Enter item name (or 'done' to finish): ").lower()
22         if item == 'done':
23             break
24         if item not in menu:
25             print("❌ Item not found in menu. Try again.")
26             continue
27         try:
28             qty = int(input(f"Enter quantity for {item}: "))
29             if qty <= 0:
30                 print("❌ Quantity must be positive.")
31                 continue
32             order[item] = order.get(item, 0) + qty
33         except ValueError:
34             print("❌ Invalid quantity. Enter a number.")
35
36 Enter item name (or 'done' to finish):
```

Output :

```

    🍽️ Hotel Menu:
    -----
    Burger      ₹120
    Pizza       ₹150
    Pasta       ₹100
    Pasta       ₹100
    Cake        ₹50
    Coffee     ₹70
    -----

    Enter item name (or 'done' to finish): burger
    Enter quantity for burger: 5
    Enter item name (or 'done' to finish): done

    📄 Final Bill
    -----
    Burger      x 5    = ₹600
    -----
    Subtotal                ₹600
    Tax (5%)                ₹30.0
    Discount                -₹60.0
    Total                  ₹570.0
    -----

    ✅ Thank you for dining with us!
    PS C:\Users\HASINI\OneDrive\Desktop\ai code>

```

Observations:

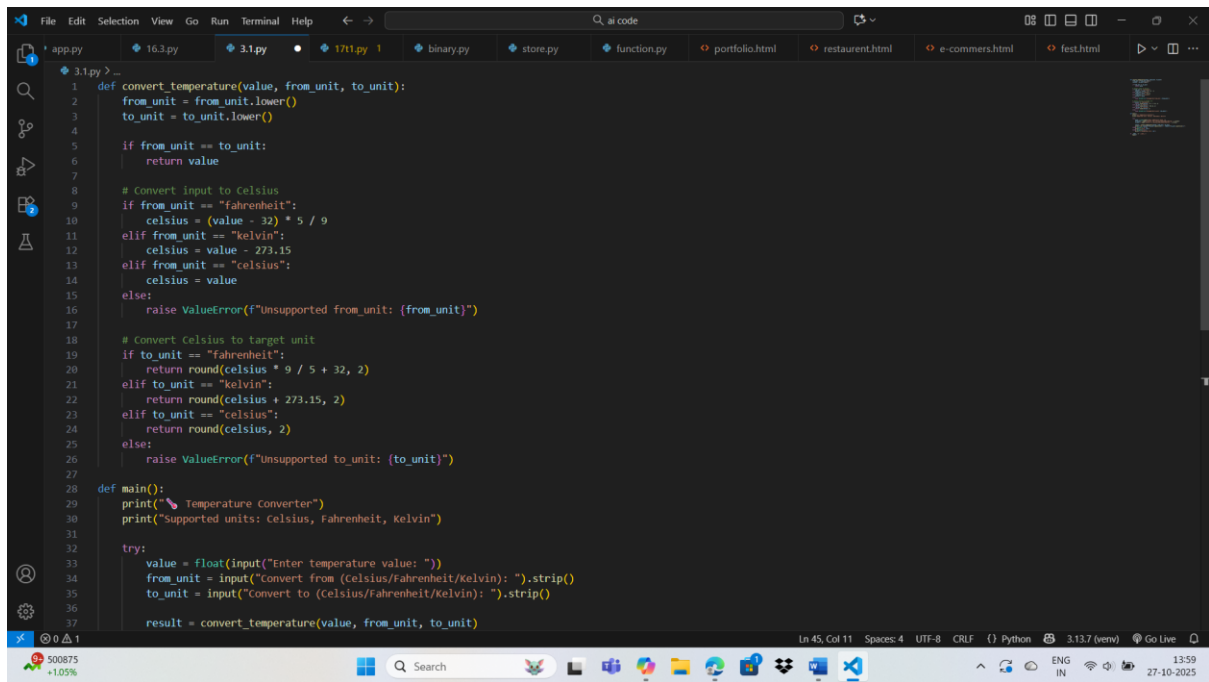
- The menu is stored as a dictionary with items as keys (capitalized) and their prices as values.
- User input is processed with `.capitalize()` so it matches the menu keys, making input case-insensitive for practical purposes.
- Separate functions are defined for repeated tasks like displaying the menu and generating the bill, ensuring code reusability.

TASK #5:

Prompt Used:

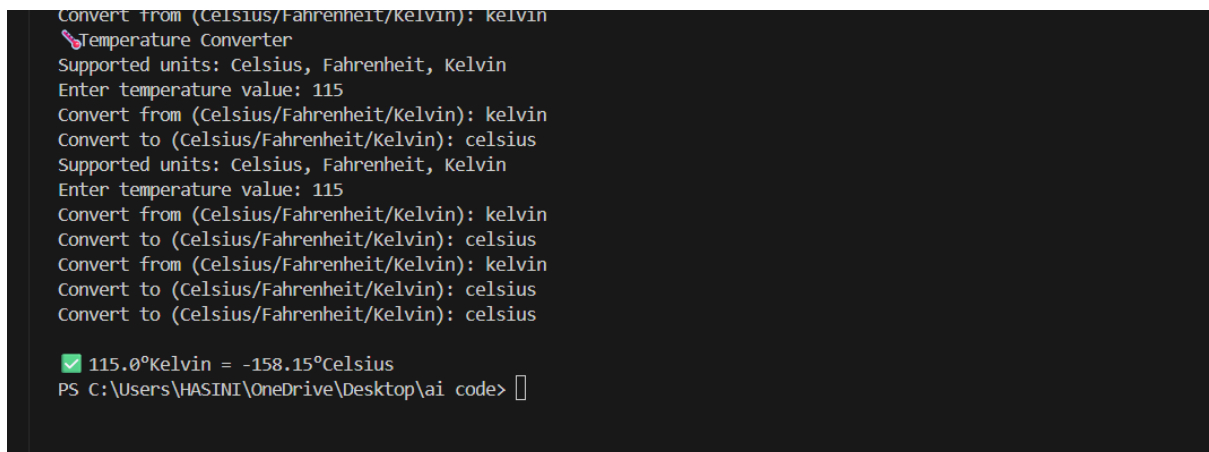
Analyzing Prompt Specificity: Improving Temperature Conversion Function with Clear Instructions.

Code Generated:



```
1 def convert_temperature(value, from_unit, to_unit):
2     from_unit = from_unit.lower()
3     to_unit = to_unit.lower()
4
5     if from_unit == to_unit:
6         return value
7
8     # Convert input to Celsius
9     if from_unit == "fahrenheit":
10        celsius = (value - 32) * 5 / 9
11    elif from_unit == "kelvin":
12        celsius = value - 273.15
13    elif from_unit == "celsius":
14        celsius = value
15    else:
16        raise ValueError(f"Unsupported from_unit: {from_unit}")
17
18    # Convert Celsius to target unit
19    if to_unit == "fahrenheit":
20        return round(celsius * 9 / 5 + 32, 2)
21    elif to_unit == "kelvin":
22        return round(celsius + 273.15, 2)
23    elif to_unit == "celsius":
24        return round(celsius, 2)
25    else:
26        raise ValueError(f"Unsupported to_unit: {to_unit}")
27
28 def main():
29     print("Temperature Converter")
30     print("Supported units: Celsius, Fahrenheit, Kelvin")
31
32     try:
33         value = float(input("Enter temperature value: "))
34         from_unit = input("Convert from (Celsius/Fahrenheit/Kelvin): ").strip()
35         to_unit = input("Convert to (Celsius/Fahrenheit/Kelvin): ").strip()
36
37         result = convert_temperature(value, from_unit, to_unit)
```

Output :



```
Convert from (Celsius/Fahrenheit/Kelvin): kelvin
Temperature Converter
Supported units: Celsius, Fahrenheit, Kelvin
Enter temperature value: 115
Convert from (Celsius/Fahrenheit/Kelvin): kelvin
Convert to (Celsius/Fahrenheit/Kelvin): celsius
Supported units: Celsius, Fahrenheit, Kelvin
Enter temperature value: 115
Convert from (Celsius/Fahrenheit/Kelvin): kelvin
Convert to (Celsius/Fahrenheit/Kelvin): celsius
Convert from (Celsius/Fahrenheit/Kelvin): kelvin
Convert to (Celsius/Fahrenheit/Kelvin): celsius
Convert to (Celsius/Fahrenheit/Kelvin): celsius
Convert to (Celsius/Fahrenheit/Kelvin): celsius

✅ 115.0°Kelvin = -158.15°Celsius
PS C:\Users\HASINI\OneDrive\Desktop\ai code> |
```

Observations:

Prompt-1:

Temperature Conversion: Basic Version

- Handles only Celsius → Fahrenheit conversion
- Code is simple, minimal, and easy to follow
- No error handling

Prompt-2:

Temperature Conversion: Enhanced Version

- Converts both Celsius ↔ Fahrenheit based on user input unit.

- Includes docstring, comments, and validation for invalid inputs
- . • Robust, readable, and user-friendly.