**LAB ASSIGNMENT – 3. 2**

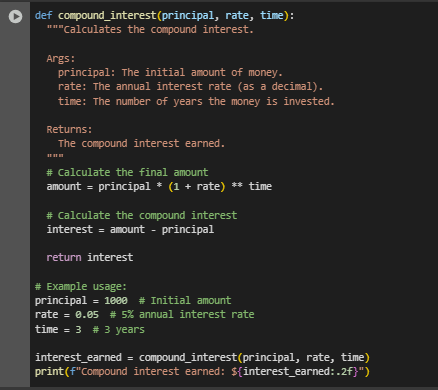
NAME : YUVARAJ REGULA

HALL.NO : 2403A52386

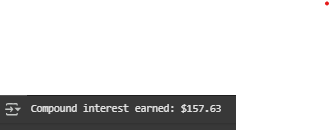
BATCH.NO : AIML-14

ASSIGNMENT : 3.2

**PROMPT 01 :**

****

**OUTPUT :**

****

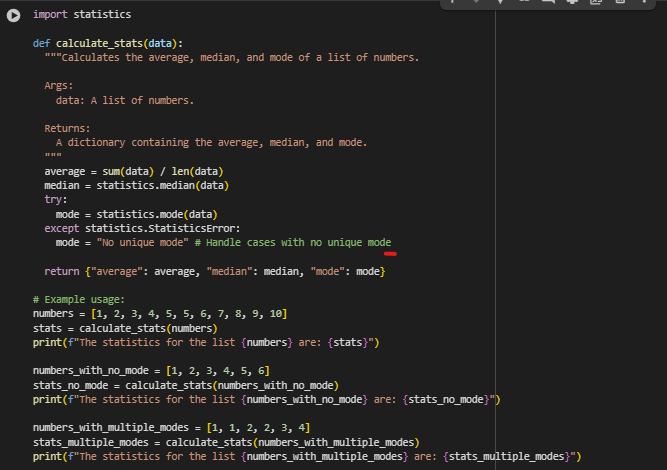
**CODE EXPLANATION :**

* def compound\_interest(principal, rate, time):: This line defines a function named compound\_interest that takes three arguments: principal, rate, and time.
* amount = principal \* (1 + rate) \*\* time: Inside the function, this line calculates the final amount after applying compound interest. It uses the formula: Amount = Principal \* (1 + Rate)^Time.
* interest = amount - principal: This line calculates the compound interest earned by subtracting the initial principal from the calculated amount.
* return interest: The function then returns the calculated interest.
* principal = 1000, rate = 0.05, time = 3: These lines set the values for the principal amount, interest rate (as a decimal), and time in years for the example usage.
* interest\_earned = compound\_interest(principal, rate, time): This line calls the compound\_interest function with the specified values and stores the returned interest in the interest\_earned variable.
* print(f"Compound interest earned: ${interest\_earned:.2f}"): This line prints the calculated compound interest, formatted to two decimal places.

**PROMPT 02 :**

I NEED A PYHTON CODE , FUNCTION TO CALCULATE AVERAGE , MEDIAN AND MODE OF A LIST OF NUMBERS

**CODE :**

****

**OUTPUT :**

****

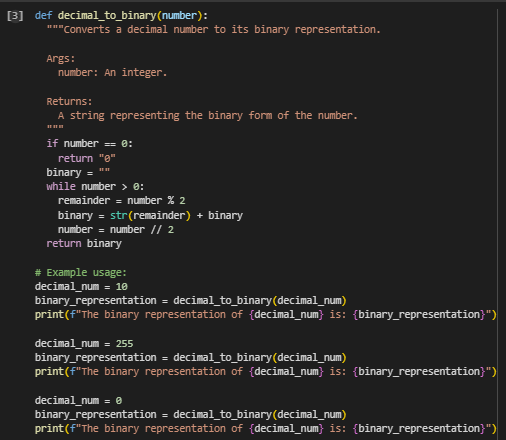
**CODE EXPLANATION :**

* import statistics: This line imports the statistics module, which provides functions for mathematical statistics.
* def calculate\_stats(data):: This line defines a function named calculate\_stats that takes one argument: data, which is expected to be a list of numbers.
* average = sum(data) / len(data): This line calculates the average (mean) of the numbers in the data list by dividing the sum of the elements by the number of elements.
* median = statistics.median(data): This line uses the median() function from the statistics module to calculate the median of the numbers in the data list.
* try...except statistics.StatisticsError:: This block attempts to calculate the mode using statistics.mode(data). The statistics.mode() function raises a StatisticsError if there is no unique mode (i.e., all values appear the same number of times or there are multiple values with the highest frequency).
* mode = statistics.mode(data): If a unique mode exists, this line calculates it using the mode() function.
* mode = "No unique mode": If a StatisticsError occurs (meaning no unique mode), this line sets the mode variable to the string "No unique mode".
* return {"average": average, "median": median, "mode": mode}: The function returns a dictionary containing the calculated average, median, and mode.
* Example Usage: The code then provides example lists (numbers, numbers\_with\_no\_mode, numbers\_with\_multiple\_modes) and calls the calculate\_stats function with each list, printing the results.

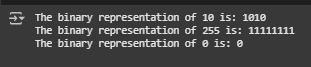
**PROMPT 03 :**

I NEED A PYHTON CODE , FUNCTION THAT CONVERTS THE NUMER INTO BINAY

**CODE :**

****

**OUTPUT :**

****

**CODE EXPLANATION :**

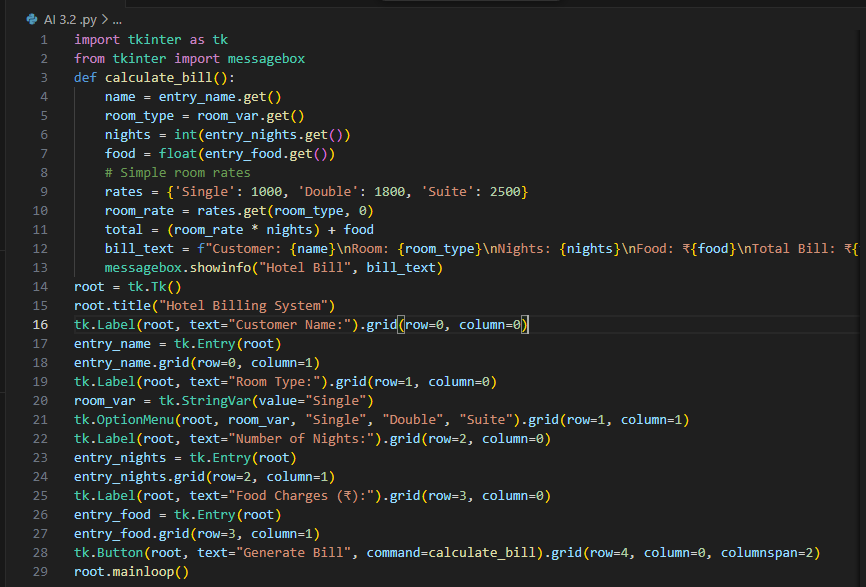
* def decimal\_to\_binary(number):: This line defines a function called decimal\_to\_binary that takes one argument, number, which is the decimal integer you want to convert.
* if number == 0:: This is a base case. If the input number is 0, the binary representation is simply "0", so the function immediately returns "0".
* binary = "": Initializes an empty string called binary. This string will be used to build the binary representation.
* while number > 0:: This starts a loop that continues as long as the value of number is greater than 0.
* remainder = number % 2: Inside the loop, this line calculates the remainder when number is divided by 2. In binary conversion, this remainder (either 0 or 1) is the next digit in the binary representation, starting from the rightmost digit.
* binary = str(remainder) + binary: This line converts the remainder (which is an integer) to a string and adds it to the *beginning* of the binary string. This is because we are calculating the binary digits from right to left.
* number = number // 2: This line updates the number by performing integer division by 2. This prepares for the next iteration of the loop to find the next binary digit.
* return binary: Once the while loop finishes (when number becomes 0), the function returns the binary string, which now contains the complete binary representation of the original decimal number.
* Example Usage: The code then shows how to use the function with different decimal numbers (10, 255, and 0) and prints the results.

**PROMPT 04 :**

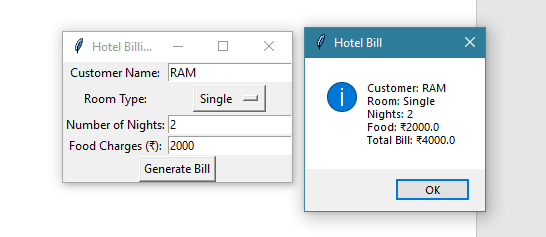
I NEED A CODE IN PYTHON TO CREATE AN USER INTERFACE FOR AN HOTEL TO GENRATE BILL BASED ON CUSTOMER REQUIREMENTS , need just a simple code as a student

NOTE : the code is runned in vs code because , directly using libraries like tkinter for a graphical user interface is not supported in this Colab environment.

**CODE :**

****

**OUTPUT :**

****

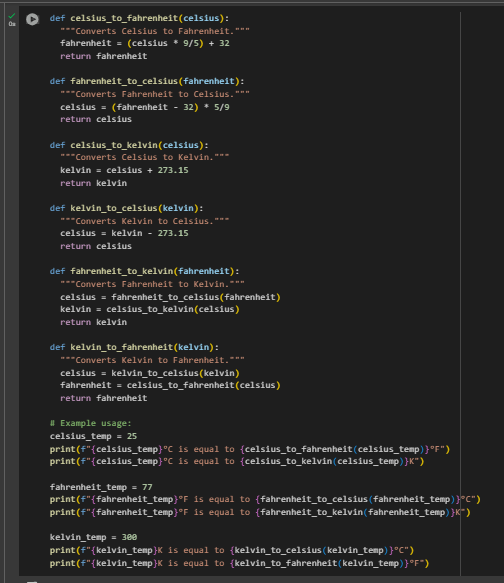
**CODE EXPLANATION :**

1. Imports tkinter modules:
   * Imports [tkinter](vscode-file://vscode-app/c:/Users/SATISH/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) for GUI and [messagebox](vscode-file://vscode-app/c:/Users/SATISH/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) for pop-up dialogs.
2. Defines [calculate\_bill](vscode-file://vscode-app/c:/Users/SATISH/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) function:
   * Gets user inputs (name, room type, nights, food charges).
   * Uses a dictionary for room rates.
   * Calculates total bill: (room rate × nights) + food charges.
   * Shows the bill in a pop-up message box.
3. Creates main window:
   * Sets the window title to "Hotel Billing System".
4. Adds input fields:
   * Customer name (Entry box).
   * Room type (Dropdown: Single, Double, Suite).
   * Number of nights (Entry box).
   * Food charges (Entry box).
5. Adds button:
   * "Generate Bill" button calls [calculate\_bill](vscode-file://vscode-app/c:/Users/SATISH/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) when clicked.
6. Runs the GUI loop:
   * [root.mainloop()](vscode-file://vscode-app/c:/Users/SATISH/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html) keeps the window open and responsive.

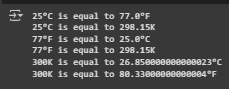
**PROMPT 05 :**

I NEED A PYTHON CODE THAT CONVERTS CELCIUS , FAREN HEIT , KELVIN INTO THE INSTRUCTUION GIVEN

**CODE :**

****

**OUTPUT :**

****

**CODE EXPLANATION :**

* **celsius\_to\_fahrenheit(celsius)**: Converts a temperature from Celsius to Fahrenheit using the formula (celsius \* 9/5) + 32.
* **fahrenheit\_to\_celsius(fahrenheit)**: Converts a temperature from Fahrenheit to Celsius using the formula (fahrenheit - 32) \* 5/9.
* **celsius\_to\_kelvin(celsius)**: Converts a temperature from Celsius to Kelvin by adding 273.15.
* **kelvin\_to\_celsius(kelvin)**: Converts a temperature from Kelvin to Celsius by subtracting 273.15.
* **fahrenheit\_to\_kelvin(fahrenheit)**: Converts a temperature from Fahrenheit to Kelvin by first converting Fahrenheit to Celsius and then Celsius to Kelvin.
* **kelvin\_to\_fahrenheit(kelvin)**: Converts a temperature from Kelvin to Fahrenheit by first converting Kelvin to Celsius and then Celsius to Fahrenheit.
* **Example Usage**: The code then demonstrates how to use these functions with example temperatures for each conversion type, printing the results.