**Manual of code**

**Lab 02**

The movies class holds a list of movies, each with a name and budget. The class includes the following methods:

**\_\_init\_\_(self):**

* The constructor method initializes an instance of the movies class with a pre-defined list of movies and their budgets. Each movie is stored as a tuple where:
  + The first element is the movie name (string).
  + The second element is the budget of the movie (integer).

**add\_movie(self):**

* This method allows the user to add new movies to the list.
* The program first asks how many movies the user wants to add.
* For each movie, the user is prompted to input the name and budget, which are then appended as a tuple to the self.movies list.

**average\_budget(self):**

* This method calculates the average budget of all the movies in the list.
* It sums the budgets of all movies and then divides the total by the number of movies to find the average.
* The result is returned as a formatted string.

**higest\_budget(self):**

* This method compares the budget of each movie with the average budget.
* It counts how many movies have a budget greater than the average and then prints the count.
* The method calls average\_budget() to get the average budget, but the result is converted from a string to a float for comparison.

**display\_movies(self):**

* This method simply prints out the list of movies, displaying each movie's name and budget.

**2. Main Program Logic (main() function):**

* The main() function is where the user interacts with the program.
* It runs a while loop that continuously shows the menu until the user chooses to exit.
* The menu includes the following options:
  + **1. Add movie**: This option calls the add\_movie() method to add new movies to the list.
  + **2. Calculate average budget**: This option calls the average\_budget() method to display the average budget of all movies.
  + **3. Display movies**: This option calls the display\_movies() method to show the list of all movies and their budgets.
  + **4. Display high budget movies**: This option calls the higest\_budget() method to count and display how many movies have a budget higher than the average.
  + **5. Exit**: This option breaks the loop and exits the program.
* The program repeatedly asks for the user's choice until they decide to exit by entering "5".

**3. Program Flow:**

* When the program starts, it displays the menu options to the user.
* Depending on the user's choice, the appropriate method from the movies class is called.
  + If the user wants to add movies, the program will ask for the movie name and budget.
  + If the user wants to calculate the average budget, it will compute and display the result.
  + If the user wants to see the list of movies, it will display each movie's name and budget.
  + If the user wants to know how many movies have a budget higher than the average, it will calculate and show the count.
* This process continues in a loop until the user selects the option to exit.

The movies class holds a list of movies, each with a name and budget. The class includes the following methods:

**\_\_init\_\_(self):**

* The constructor method initializes an instance of the movies class with a pre-defined list of movies and their budgets. Each movie is stored as a tuple where:
  + The first element is the movie name (string).
  + The second element is the budget of the movie (integer).

**add\_movie(self):**

* This method allows the user to add new movies to the list.
* The program first asks how many movies the user wants to add.
* For each movie, the user is prompted to input the name and budget, which are then appended as a tuple to the self.movies list.

**average\_budget(self):**

* This method calculates the average budget of all the movies in the list.
* It sums the budgets of all movies and then divides the total by the number of movies to find the average.
* The result is returned as a formatted string.

**higest\_budget(self):**

* This method compares the budget of each movie with the average budget.
* It counts how many movies have a budget greater than the average and then prints the count.
* The method calls average\_budget() to get the average budget, but the result is converted from a string to a float for comparison.

**display\_movies(self):**

* This method simply prints out the list of movies, displaying each movie's name and budget.

**2. Main Program Logic (main() function):**

* The main() function is where the user interacts with the program.
* It runs a while loop that continuously shows the menu until the user chooses to exit.
* The menu includes the following options:
  + **1. Add movie**: This option calls the add\_movie() method to add new movies to the list.
  + **2. Calculate average budget**: This option calls the average\_budget() method to display the average budget of all movies.
  + **3. Display movies**: This option calls the display\_movies() method to show the list of all movies and their budgets.
  + **4. Display high budget movies**: This option calls the higest\_budget() method to count and display how many movies have a budget higher than the average.
  + **5. Exit**: This option breaks the loop and exits the program.
* The program repeatedly asks for the user's choice until they decide to exit by entering "5".

**3. Program Flow:**

* When the program starts, it displays the menu options to the user.
* Depending on the user's choice, the appropriate method from the movies class is called.
  + If the user wants to add movies, the program will ask for the movie name and budget.
  + If the user wants to calculate the average budget, it will compute and display the result.
  + If the user wants to see the list of movies, it will display each movie's name and budget.
  + If the user wants to know how many movies have a budget higher than the average, it will calculate and show the count.
* This process continues in a loop until the user selects the option to exit.