**1. Class Definition (student)**

1.1 Class Attributes

* **subject**: Class attribute representing the subject, set to "Fundamental of Data Structure."
* **max\_marks**: Class attribute representing the maximum marks, set to 100.

1.2 Constructor (**\_\_init\_\_**)

* Initializes an instance of the class with an optional parameter **marks** (default value is 0).
* Prints a message indicating the creation of a student for a specific subject.
* Sets the **marks** attribute of the instance.

1.3 Methods

* **setmarks(self, marks)**: Sets the **marks** attribute to the given value.
* **getmarks(self)**: Returns the **marks** attribute of the instance.

**2. Menu List (menu1)**

* A list containing menu options for the program.

**3. Function: printMenu()**

* Prints the menu options stored in the **menu1** list.

**4. Main Program**

4.1 **score** List

* Initializes an empty list **score** to store the marks of students.

4.2 Menu Interaction Loop

* Enters a loop to interact with the user until the user chooses to exit (option 6).

4.3 Menu Options

**4.3.1 Option 1**

* Allows the user to add a student to the class.
* Takes input for the student's marks and creates a **student** instance with the provided marks.
* Appends the student's marks to the **score** list.

**4.3.2 Option 2**

* Calculates and prints the average score of the class.

**4.3.3 Option 3**

* Finds and prints the highest and lowest scores in the class.
* Creates a copy of the **score** list (**sc1**) and removes any occurrences of -1 (marks indicating absence) before finding the minimum.

**4.3.4 Option 4**

* Counts and prints the number of students who were absent (marks equal to -1).

**4.3.5 Option 5**

* Finds and prints the marks with the highest frequency in the class.
* Uses a dictionary (**d**) to store the frequency of each unique mark.
* Determines the maximum frequency and prints the marks with that frequency.

4.4 Exit Option

* Breaks out of the loop if the user chooses option 6.

**Note:**

* The code assumes that the user always inputs valid integers for choices and marks, which might not always be the case in a real-world scenario.
* The code could benefit from better variable names and more comments to improve readability.
* The absence of students is represented by the mark -1.

Algorithm:

1. Define a class student with two attributes: subject and max\_marks. The subject attribute is set to "Fundamental of Data Structure", and the max\_marks attribute is set to 100. The class also has three methods: \_\_init\_\_(), setmarks(), and getmarks(). The \_\_init\_\_() method initializes a student object with a default mark of 0. The setmarks() method sets the mark for a student object, and the getmarks() method returns the mark for a student object.

2. Define a list score to store the marks of all the students in the class.

3. Define a menu with six options.

4. Use a loop to repeatedly display the menu and perform the selected operation until the user chooses to exit the program.

5. When the user selects option 1, prompt the user to enter the marks for a student. If the marks are greater than the maximum marks, print an error message. Otherwise, create a new student object with the entered marks and add the student's marks to the score list.

6. When the user selects option 2, calculate and print the average score of the class.

7. When the user selects option 3, find and print the highest and lowest scores in the class.

8. When the user selects option 4, count and print the number of students who were absent for the test.

9. When the user selects option 5, find and print the mark with the highest frequency.

10. When the user selects option 6, exit the program.