Additional Exercise 6

1. An application is required to enter marks of an assessment for a tutorial group of students. Design the following classes, followed by a console based menu that allows marks to be entered.
   a. A user defined exception InvalidMarkException that subclass from Exception.

   b. A Student class with the following requirements:
      i. A constructor that has 2 parameters that initialises the id and name. It also has an instance variable for mark set to 0.
      ii. Properties for id, name and mark.
      iii. A setter for mark. Raise InvalidMarkException with message 'Mark must be between 0 and 100' if the mark is out of the range 0 to 100.
      iv. A __str__ that returns the id, name and mark.

   c. A TutorialGroup class that consists of students and manages the search, add, reset of marks for students. It has the following requirements:
      i. A constructor with 1 parameter to initialize the tutorial group name as instance variable. It also has a dictionary of students initially set to empty.
      ii. Property for name.
      iii. Search(id) method that searches the dictionary using the id and if found, returns the Student object, else returns None.
      iv. addStudent(id, name) adds a Student object in the dictionary. The key is the id, and value is a Student object. Raise an InvalidMarkException 'id already exists', where id is the passed as parameter if a student with the id is already in the dictionary.
      v. addMark(id, mark) adds a mark only if id is in the dictionary (raise an exception if not in dictionary). A mark is assigned only when the current mark is 0. If it is not 0, that means a mark has already been entered, in which case, raise an InvalidMarkException 'Mark has already been assigned'.
      vi. resetMark(id) resets a mark to 0 only if id is in the dictionary(raise exception similar to part v). A mark is reset to 0 only when the current mark is not 0. If the mark is already 0, raise and InvalidMarkException 'Mark is already 0!'. Return the mark before it was set to 0.
      vii. getStudentMarks(option) that returns a formatted string of the student marks. If option is "Fail", return only students with marks < 50. If option is "All", return all marks. The returned string should be in the following format:

```
Id  Name    Mark
s1  John    50
s2  Peter   75
…
```

   d. Create a main function with the following:
      i. Create a tutorial Group object for 'T02'.
      ii. Add 3 students to the tutorial group:
          S1 John
          S2 Peter
          S3 Joe
      iii. Create a menu as follows:
          1. Add mark

2. Reset mark
3. Display marks
4. Quit

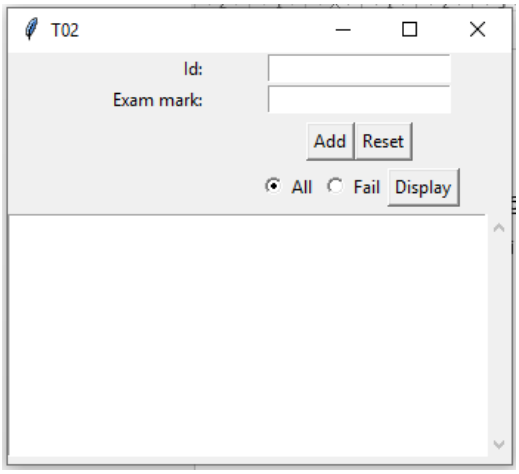
Enter option:

iv. Add option. Create a addMark(tg) function that has the tutorial group tg as parameter. Prompt the user for id, and mark and add the mark to the tutorial group. If successful, display 'Marks entered'. All exceptions must be caught and messages displayed.

v. Reset option. Create a resetMark(tg) function, and prompt for id. If reset is successful, display 'Mark reset for XXX from 99 to 0 successful!' where XXX is the name of the student, and 99 is the original mark.

vi. Display option(tg). Prompt user to enter All or Fail, and display the output.

2. Create the following AssessmentGUI to enter marks for an assessment:



Make use of the Student and TutorialGroup classes in the previous question. Requirements are as follows:

a. The constructor of the GUI has a parameter to initialise the tutorial group. Populate the tutorial group with 3 students before passing the object to the GUI.
b. The title of the GUI has the tutorial group name.
c. The Add, Reset and display buttons are self explanatory and work the same way as the requirements of the previous question.