Question 1:

Define the class Car as shown in the UML Class diagram. Then Implement a main to test your class. The main should create three cars and change their colors 2, 4, 1 times respectively and print them.

The method getNumberOfColorChanges() returns how many times the method setColor() was called.

```
Car
- model: String
- year: int
- color: String
- numberOfColorChanges: int
+ setModel(String): void
+ getModel(): String
+ setYear(int): void
+ getYear(): int
+ setColor(String): void
+ getNumberOfColorChanges(): int
+ print(): void
```

Question 2:

Update the class Car in the previous question to limit the number of color changing to three times. Then run the same main.

The method setColor() should maintain this limit before setting the new color.

This time the run should displays 2, 3, 1 as the numbers of color changes.

Question 3:

Define the class Student as shown in the UML Class diagram. Then write a program that reads the name and the mark of a student then prints his grade. The program should repeat the previous procedure (reading mark then print grade) until the user types "done" when the program prompt the name of the next student.

```
Student
- name: String
- grade: char
+ setName(String): void
+ getName(): String
+ setGrade(int): void
+ getGrade(): char
+ print(): void
```

Benefits of Encapsulation:

- Data members can be read only or write only.
  - numberOfColorChanges in the class Car is read only.
- The class has a full control over its data members.
  - We can limit the number of changes or the values range.
- The users of a class do not know how the class stores its data.