**School Management System**

* **Step 1**
  + Create a console program
  + It will ask user if he wants to add a student, view a student or delete a student.
  + User will select any of the option (using `Console.RadLine();`)
  + When the user selects an option, print message *"task student (add/delete/view) completed."*
* **Step 2**
  + Now add a student class.
  + It will have three properties. Name, Father Name, Adress
  + Write getters and setters for these properties.
* **Step 3**
  + Add a menu option to exit the menu.
  + Program will keep running until user selects exit option.
* **Step 4**
  + Implement add student menu.
  + When user will select add student, you will ask the user to input student name, etc.
  + Then create a student object using the input.
  + User should be able to create as many student objects as he wants.
  + This `AddStudent` method should be in the same class which has `main` function
* **Step 5**
  + Now add a method in `student` class
  + name the method `Display()`
  + It will display the fields of the object like this  
    Name: Ali  
    Father: Baba  
    etc
* **Step 6**
  + Add a list field that will contain students in the same class which has main method.
  + This field will be private and static.
  + In AddStudent method add the newly created student object to the list.
* **Step 7**
  + Extract `AddStudent` method and list field to a new class
  + Name the class `StudentController`  
    `StudentController` will have `AddStudent` method that you have already defined and the list field
  + the list field will be private but NOT static
  + Create an object of `StudentController` in the main method
  + When user selects add student menu, call `AddStudent` method of `StudentController` object
* **Step 8**
  + Add method view students to `StudentController` class
  + this method will iterate over the list and call display method of each student object
  + call this method when user selects view students menu option
* **Step 9**
  + Create separate files for Student and StudentController classes
  + File name should match class name  
    For example Student class will be in Student.cs file
  + Make sure your program compiles and works properly
* **Step 10**
  + Add a string field to student class `id`
  + User should enter `id` when he enters name, father name and address
  + Do not create student object if `id` is not given or if `id` is empty string `""`
  + Give warning if user does not enter the `id`
* **Step 11**
  + Show student id in `display` method along with name, father name etc
* **Step 12**
  + Inside `addStudent` method of `studentController` class, make sure **you do not add** a student if `id` has already been used  
    Hint  
    In the AddStudent method, after you make sure that the user has entered user id,  
    inside `if (id != "")`  
    iterate over all the student objects in the list  
    compare their id with the input id  
    if it matches then give error  
    otherwise add student to studentsList  
      
    Set a flag before you start the loop.  
    `matchFound = false;`  
    Inside the loop, compare the objects' id with the input user\_id  
    If match is found then set `matchFound` to true.  
    After the loop, check the value of matchFound  
    If matchFound is true then print error  
    Otherwise add student.
* **Step 13**
  + Add a private method in `StudentController`, `isStudentIDUnique`
  + this method will return a bool
  + it will take a string argument
  + extract the code written in `AddStudent` in [Step 12](https://dynalist.io/d/hllrmQt-Ob2BICKayP5VtQNl#z=e3esjGLc-iCKt3hpg5NdiJ5e) and put it inside `isStudentIDUnique` method
  + this means that `matchFound` check and loop, both will be present inside `isStudentIDUnique` method
  + this means that `matchFound` check and loop, both will NOT be present inside `AddStudent` method
  + this means that `AddStudent` method will call `isStudentIDUnique` method and pass the user ID from user input to it in arguments
  + `AddStudent` will check the return value of `isStudentIDUnique` and then it will add student if it is true, or show error if it is false
* **Step 14**
  + Exit the loop inside `isStudentIDUnique` as soon as a match is found
* **Step 15**
  + If user enters an empty ID then ask for ID again
  + print "id cannot be empty" if user does not enter the ID
  + If user enters an ID that is already used then ask for ID again
  + print "id is already used" if user enters an ID that has already been used
  + Keep asking user for ID until user enters a valid ID
* **Step 16**
  + When user select delete student from menu, ask for student ID
  + iterate over the list, find the object with matching id, store the index in a variable
  + remove the object from student list using `list.removeAt()`
  + print error if id is not found