Lab 7

# Objectives

Practice C# code writing that involves delegates and events.

# Instructions

* Install Visual Studio 2017 or Visual Studio 2019 to your computer.<https://visualstudio.microsoft.com/>
* For each question below submit your C # and also screenshots showing how your program compiles and executes (sample outputs)- You can upload your C# project but alternatively create a document with your C# code **text** copied and pasted and your screenshots pasted and then upload the document to Blackboard for submission. Prof. Aydin should be able to run your submitted code.
* Your name should appear on the screenshots for receiving full credit.
* **Academic Integrity:** If you are stuck when working in this lab you can collaborate with a couple of classmates. In that case, make sure to write/submit the name of your collaborators and any web site you used as a resource to understand the concepts and lab questions and to complete your code to prevent plagiarism and breach of academic integrity.
  + You are *not* allowed to directly copy code from the Internet, your friends, and other resources without spending any effort in completing the work. Make sure to review the academic integrity policy in the syllabus and ask for clarification, if needed.
* You will utilize event(s) and delegates to complete this lab in a student registration system. Hints:
  + Review [Week 9 (Chapter 8) slides](https://drive.google.com/drive/folders/1-voMzBRitOdXiBNl8SrMm_WM-CbnhxQD?usp=sharing) on events (publisher and subscriber classes).
  + Study the example project [chp08/EventsSample](https://drive.google.com/drive/folders/1nLBxPUxidq8TE_PMEqcMzBE5f-NRtJ9C?usp=sharing)
* **Publisher (Producer) side:** Create **class Student**
  + with Properties
    - name (string)
    - date of birth ([DateTime](https://docs.microsoft.com/en-us/dotnet/api/system.datetime?view=netcore-3.1))
    - major (string)
    - status (enum type with values FRESHMEN, SOPHOMORE, JUNIOR, SENIOR that you can create in your program)
    - registered (bool)
  + Add other appropriate methods and constructors
  + Event **newStudentArrived event will be fired** to be handled by the consumer class Registrar (depending on your design you can add this capability to class Student or to another class that you see fit)
* **Subscriber (Consumer) side:** Create a **class Registrar**
  + With Property/Field
    - student list (could be an array of Student's).
  + Add other appropriate methods and constructors
  + When a new student arrives, the registrar will take the student info, update the registered field to true and place (that is save) the studentinfo in the student list.
* **The main program**
  + will create a registrar with no students
  + will create a new student every N milliseconds where N is a random number between (100-3000). You can use [Random class](https://docs.microsoft.com/en-us/dotnet/api/system.random?view=netcore-3.1) to create a random number and [Thread.Sleep](https://docs.microsoft.com/en-us/dotnet/api/system.threading.thread.sleep?view=netcore-3.1) method to pause your main program in between a new student creations
  + Create at least 3 new students with appropriate data of your choice (name, date of birth, status, …). Note that the **registered** feature of each student is initially false. After creation, each student will fire the newStudentArrived method to be handled by the registrar.
  + Make sure to add appropriate outputs to your main program that show the user the order of events and how the events are handled.