

Assignment 1: Console Application

1. Objectives

- Create your first C# Console Application using two classes.
- Work with frequently used data types: `int`, `double`, `bool` and `string`.
- Utilize some common methods from the Console class.
- Learn user interaction: receiving input and displaying output.

2. Description

This assignment comprises a set of programming tasks, offering the flexibility to select the grade level you aim to achieve. Your grade will depend on your performance and adherence to the assignment requirements. You will have the opportunity to revise and resubmit your work for improvement. However, it is advisable to target a minimum grade of C, although achieving a pass grade of D is also possible.

Create a Console Application using either .NET 6 (or later), or .NET Framework 4.7/4.8 (if using Visual Studio 2019). Organize your project by creating a dedicated folder for this assignment. Also, back up regularly your solution to another place on your hard disk to avoid data loss.

The assignment is divided into two parts:

- **Part 1:** Completion of this part is mandatory for achieving a pass grade (D & C).
- **Part 2:** Necessary for attaining grades B and A in addition to completing Part 1.

Your application should include a main class (Program.cs) hosting the Main method to start the application. Create separate classes for each task of the assignment. User input validation is not required for this assignment. Assume all user-provided input values are valid.

The following table summarizes the tasks to be completed in this assignment:

Task to do	Grade
1. Pet: A class managing basic data for a pet animal.	D
2. TicketSeller: A class calculating entrance fees.	
3. Album: A class that handles some basic data for a music album	C
4. Own Class: Select and create a class of your choice (e.g., Album, TV, Weather, etc.) following the above requirements	B
5. Use an additional .NET object such as DateTime or Math , either as an instance variable or within the code, in Task 3.	A

Note that for Grade D, you must complete both Task 1 and Task2. To attain any grade higher than D, it is mandatory to complete all the preceding tasks. For instance, to achieve Grade C,

you must complete the **Pet**, **TicketSeller** and **Album** classes. In the same manner, to secure Grade A, it is necessary to finish all the classes outlined in the provided table.

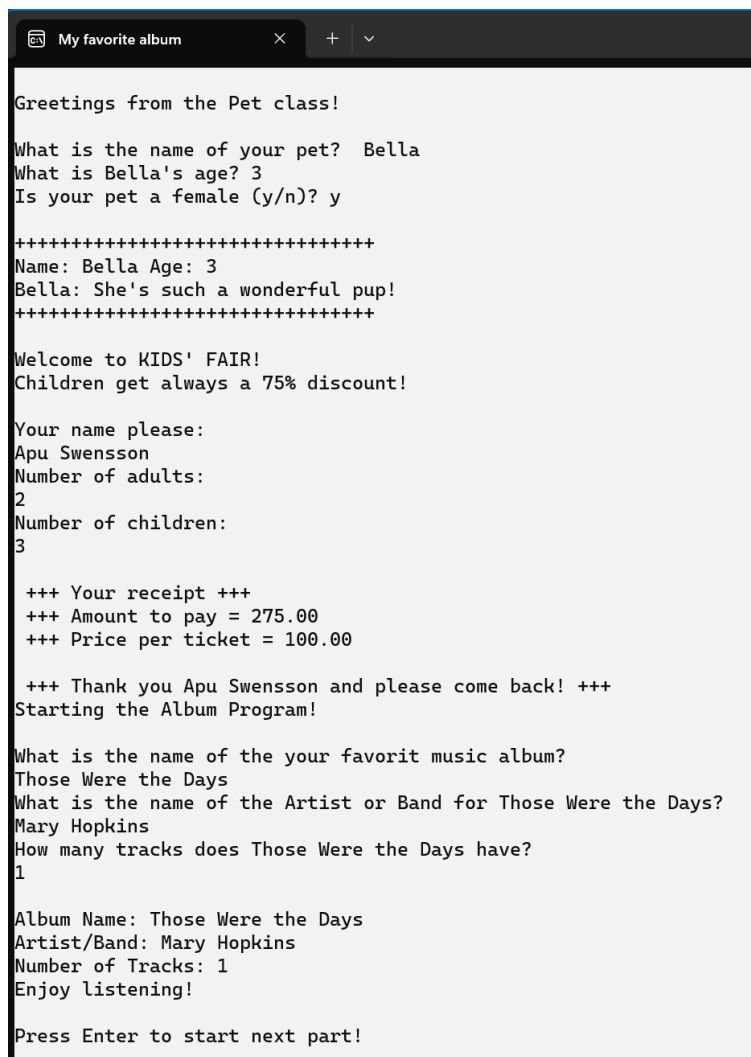
Begin writing one task at a time, run and test it before proceeding with the next one. For testing each class, you should create an instance of the class in the **Main** method within the **Program** class. Detailed instructions for this process will be provided later in the this description.

3. Part 1: Console Application for Grades D and C

Create a Console application containing the following three classes:

- **Program.cs**: The start class hosting the method Main using the other classes to make the application work.
- **Pet.cs**: Class that stores and manipulates some data about an animal.
- **TicketSeller.cs**: Class that calculates entrance fees to an event.
- **Album**: Class that stores and handles some basic data about a music album.

The following image is a screen shot of the output from the execution of a sample program. It is allowed to design the user interface using your own preferences.



```

My favorite album
Greetings from the Pet class!
What is the name of your pet? Bella
What is Bella's age? 3
Is your pet a female (y/n)? y

+++++
Name: Bella Age: 3
Bella: She's such a wonderful pup!
+++++

Welcome to KIDS' FAIR!
Children get always a 75% discount!

Your name please:
Apu Swensson
Number of adults:
2
Number of children:
3

+++ Your receipt +++
+++ Amount to pay = 275.00
+++ Price per ticket = 100.00

+++ Thank you Apu Swensson and please come back! +++
Starting the Album Program!

What is the name of the your favorit music album?
Those Were the Days
What is the name of the Artist or Band for Those Were the Days?
Mary Hopkins
How many tracks does Those Were the Days have?
1

Album Name: Those Were the Days
Artist/Band: Mary Hopkins
Number of Tracks: 1
Enjoy listening!

Press Enter to start next part!

```

In the above example, ticket prices are calculated at a rate of SEK100 per adult, with children's tickets priced at 25% of the adult fare. Additionally, the title of the Console window will change to display the name of the class being tested. The appearance of the Console window is modified from within the Main method. The code for this modification, along with the **Main** method code, will be provided later in this document.

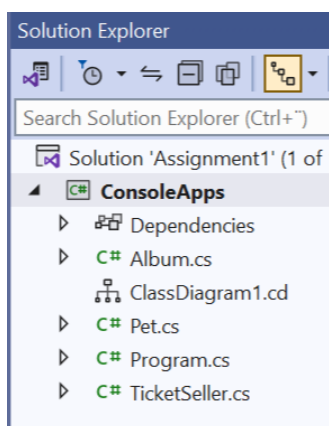
4. Solution and project

Begin by creating a new project in Visual Studio. If you are using Visual Studio 2022, select the C# Console Application option (specifically, the one without the ".NET Framework" text). Visual Studio will then automatically create a solution containing your project, along with a folder named "Dependencies" within the project.

For those working with Visual Studio 2019 and .NET Framework, VS will create a folder named **Properties** instead of Dependencies.

The image below illustrates a potential project structure. In this example, the solution is named **Assignment1**, and the project is called **ConsoleApps**, which is also the name of the namespace. If you are using Visual Studio 2022, remember to select the option indicated in the image below (Additional information).

The following images show the structure of the classes.



Additional information

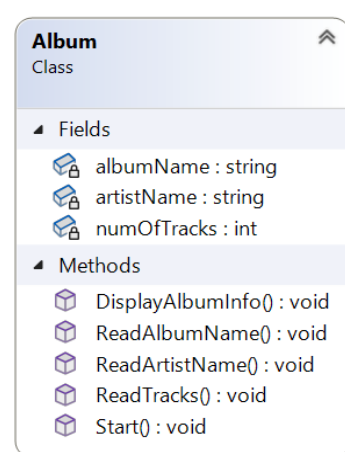
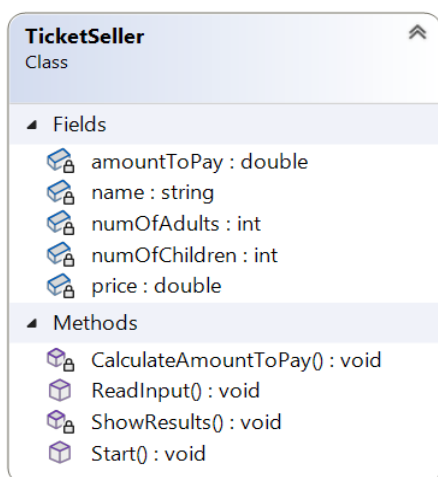
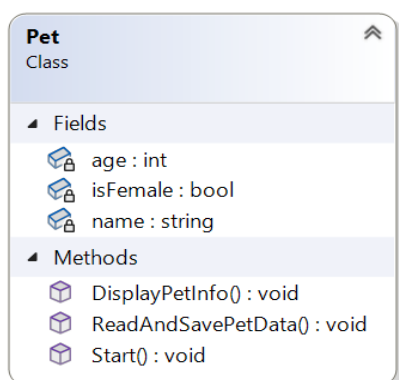
Console App

C# Linux macOS

Framework ⓘ

.NET 6.0 (Long Term Support)

☒ Do not use top-level statements ⓘ



Each of the images provided above represents a section of what is known as a class diagram. This class diagram is offered as a guide; however, you are not required to strictly follow it and may choose to employ your own design.

The type of each field is specified following the colon sign, and similarly, the return type of the methods is also indicated after the colon sign. All methods used in the above classes are void and do not return any value.

5. Requirements for Part 1, Task 1 – Grade (D)

The program should prompt the user for the name, age, and gender of a pet (or any animal). These values are to be stored in the program, which should then display the information back to the user.

5.1 Create a class with the name **Pet** and three fields (instance variables): **name** (**string**), **age** (**int**) and **isFemale** (**bool**). Save the class as **pet.cs** on your hard disk

5.2 All fields should be declared as **private**.

Note: Keep in mind that **only** private instance variables are allowed throughout all the assignments.

5.3 Write at least two methods in the **Pet** class, one for reading user input (the above three values) and one that displays the pet information as in the above figure.

5.4 Write a method **Start** in which you should call the methods of the class in an orderly manner to read input and then display output (see the above run-time example).

5.5 VS creates an empty starting class and name it "**Program**" saved as **Program.cs** in the project directory. This class contains a ready-to-use **Main** method. Create an object of the **Pet** class and call the **Start** method of this object to test its functionality.

5.6 Below is a code snippet showing the field declaration and a possible construction of the **Start** method. After this, write the two methods that are used within this method.

```
namespace ConsoleApps
{
    2 references
    class Pet
    {
        private string name; //name of the pet
        private int age;      //age as an integer
        private bool isFemale; //true if female, false otherwise

        1 reference
        public void Start()
        {
            Console.WriteLine(); //blankline
            Console.WriteLine("Greetings from the Pet class!");
            Console.WriteLine(); //blankline

            ReadAndSavePetData();
            DisplayPetInfo();
        }
    }
}
```

Note that writing a **Start** method is merely a suggested approach to organize the calling order of methods. It is not a predefined method like `Main`. It is a standard method that we choose to write to structure our code. We could use a different name instead of “Start” or choose not to use it at all. In the latter case, we would then need to call the methods one by one from the `Main` method for input reading and displaying output.

After you have finished implementing the **Pet** class, go to the `Main` method. There, write code to (1) instantiate an object of the **Pet** class and then (2) call its **Start** method to run and test the object’s features. For your convenience, the code is provided in the image below. You can use this as a template to test other classes as well. Additionally, you will find code to alter the appearance of the Console Window. Note that modifying the Console Window is optional, and you may choose not to implement this feature and use the default settings.

```
3 namespace ConsoleApps
4
5     0 references
6     class Program
7     {
8         0 references
9         static void Main(string[] args)
10        {
11            SetupConsoleWindow();
12
13            Console.Title = "My Favorite Pet";
14            //Create a pet object
15            Pet petObj = new Pet();
16
17            //Call a method of the object to run
18            petObj.Start();
19
20            //Continue with other classes as in above
21
22            Console.WriteLine("Press Enter to start next part!");
23            Console.ReadLine();
24        }
25
26        1 reference
27        static void SetupConsoleWindow()
28        {
29            //Arrange the Console Window
30            Console.BackgroundColor = ConsoleColor.White;
31            Console.Clear(); //Paint the background with above color
32            Console.ForegroundColor = ConsoleColor.Black;
33            Console.Title = "My Console Classes";
34        }
35    }
```

If you require additional help and guidance, step-by-step instructions for programming the **Pet** class are provided in a separate document titled **Assignment1Help**, which can be found under Module 1 in Canvas. After testing your application and ensuring everything functions correctly, you may then proceed to the next class.

6. Requirements for Part 1, Task 2 – The class TicketSeller - Grade (D)

6.1 Create a new class named **TicketSeller** and follow the steps as in the previous Task to complete this part. Begin by declaring some fields for the class to store values, as shown in the following image.

6.2 Implement a constant price for adult tickets. You may also define another constant for the children's discount (private double childDiscount = 0.25).

6.3 Write a **Start** method as illustrated in the image on the right side, and implement the methods specified in the method. These methods are essential for reading input from and printing output to the Command Prompt window. Then, go to the Main method and write code to create an object of **TicketSeller** and then call the object's Start method so that it performs its intended functions.

```
class TicketSeller
{
    //fields - input
    private string name;
    private double price = 10.0;
    private int numOfAdults;
    private int numOfChildren;

    //fields - output
    private double amountToPay;

    0 references
    public void Start()
    {
        ReadInput();
        CalculateAmountToPay();
        ShowResults();
    }
}
```

If your objective is to achieve a grade D, you may opt to skip the rest of this document, as it contains criteria for grades C, B, and A. However, it is strongly recommended to aim for at least a grade C in all assignments.

7. Requirements for Part 1, Task 3 – Grade (C)

7.1 Create a new class, **Album.cs** and follow the same steps as the previous tasks to complete the class.

7.2 Declare the fields of the class and write a **Start** method for setting up the order of how the class is to operate. Implement the methods specified within this Start method.

7.3 When you have completed the class, proceed to the **Main** method and as before write code to create an instance of the class. Then, invoke the Start method, run the program and test how the whole program works.

```
public class Album
{
    private string albumName;
    private string artistName;
    private int numOfTracks;

    1 reference
    public void Start()
    {
        ReadAlbumName();
        ReadArtistName();
        ReadTracks();
        DisplayAlbumInfo();
    }
}
```

If you are not aiming for a grade higher than C, you can skip Part 2 below.

8. Part 2 - Program your own object – only for A and B Grades

This section is exclusively for those aiming to achieve an A or B grade and must be completed in addition to Part 1. In this part, you are required to select any object type (a class) and program it in a manner similar to the classes in Part 1. This task involves defining at least three fields of different data types and creating methods that read user input, possibly perform some calculations or data manipulation, and finally, display the output.

Look around you, whether you are at home, at work, or elsewhere. You will notice a variety of objects like chairs, babies, TVs, houses, or cars. Select your favorite object and incorporate it into the application as a new class to represent a group of such objects. The chosen object type does not have to be a tangible item; it can also be a conceptual object, such as a Contact, Calculator, etc.

For Grade B: Task 4

Follow the same approach as in Part 1 to complete this section. Test your code from the Main method as done previously.

- 7.1 Create a class for your chosen object type. It should include at least three instance variables as fields, one of which must be a double (e.g., price, amount). Select appropriate data types for the other fields.
- 7.2 Implement methods to read user input, and if required do some calculations, and then display output.

For Grade A: Task 5

- 7.3 Utilize either the **DateTime** class, the **Math** class or any other ready-to-use class from .NET in your class created in the previous task. For instance, you might use a **DateTime** object as an instance variable or a local variable within a method to store or display a date and time. Alternatively, you can use the Math class in a method to carry out some mathematical calculation.

9. Submission

Ensure that your submission is well-organized, thoroughly tested, meeting the above requirements.

Close Visual Studio and then, using File Explorer, compress all your files using a zip, rar, or 7z format. Go to the Assignment 1 page in Module on Canvas, where you downloaded this document, and submit your compressed file as an attachment. Do not upload single files, and do not send your solution by email. Uploading only the **.sln** file is not sufficient and will not work to open the application in Visual Studio. Make sure that you include all project files.

In the event that the Canvas platform is down (which does not happen often), wait until it is up and running again (even past the deadline)

Good luck!