Assignment 1: Console Application
Part I – Kids' Fair

1. Objectives

- To create your first C# program as a Console Application with two classes
- To work with some built-in data types such as int, double, char, string and bool.
- To exercise with some useful methods of the **Console** class.
- To learn how to interact with the user to receive input and display output.

2. Description

This assignment consists of two main parts, Part 1 that is mandatory for a pass grade (C or D) and Part 2 to qualify for a higher grade, A or B. The requirements for each grade are specified down below. It is highly recommended to go for at least a C grade, which is based on minimum requirements. Thus, it is your decision as to which grade level you intend to work for, and in case of shortages in meeting the requirements no matter which you are doing, you will be given the chance to improve your solution and resubmit. This is depending on how well the code is structured and how the requirements are met. These considerations apply to all the assignments in this course.

For each application in this assignment (Part 1 and Part 2), create a class containing a Main method that starts the application. Details and guidance are given below. Control of user input is not mandatory in this assignment. Expect that users provide valid values.

For each parts of this assignment, you need to create a Console App preferably using .NET 5 or .NET Framework. You can work with either two separate solutions or two projects within the same solution. It is recommended that you create a separate folder on your computer for each assignment in this course. It is also important to take a backup of your solution.

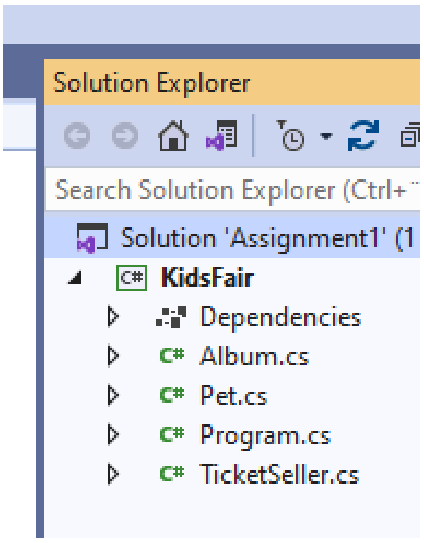
3. Part 1: Console application for grades D & C

a. **Pet** is a class that handles some basic data for a pet animal.

b. **TicketSeller** is a class that calculates the amount to pay for entrance fee

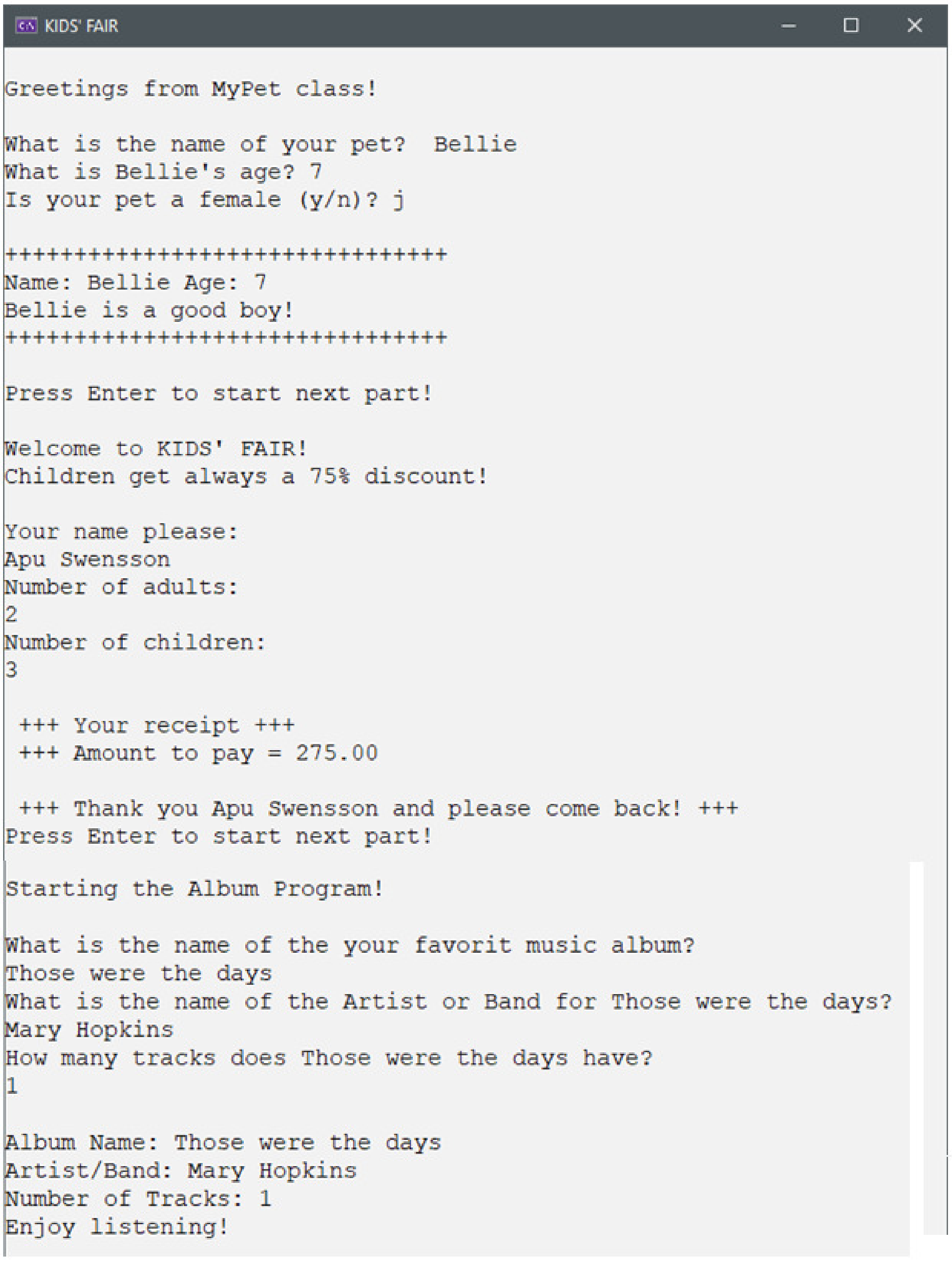
c. **Album** is a class that handles some basic data for a music album.

Each of the above options must be written as a separate class and saved in its file. Create then an object of each class in the Main method inside the class Program. When creating the solution in Visual Studio, select the option Desktop Console



Application (without ".NET Framework"), and Visual Studio will then create a folder "Dependencies". In case you have not installed the latest version of .NET, you will instead get a folder named "Properties".

For the **Pet-**class, you are given some guidance and instructions below, in addition to a step-by-step separate help document. You will then be able to solve the other two classes using your skills and following the same procedure. The following figure demonstrates a run example of Part 1.



```
Greetings from MyPet class!

What is the name of your pet?  Bellie
What is Bellie's age? 7
Is your pet a female (y/n)? j

++++++++++++++++++++++++++++++++
Name: Bellie Age: 7
Bellie is a good boy!
++++++++++++++++++++++++++++++++

Press Enter to start next part!

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

 +++ Thank you Apu Swensson and please come back! +++
Press Enter to start next part!

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!
```

The Main method for Part 1 is given later in this document.

4. Part 2: Console application for grades A and B

This part is only for those who aim to achieve a B or an A grade. In this part, you are allowed to select any object type (a class) and program it in much the same way as Part 1 but with two classes a Program.cs and your class. Develop a new application in the same solution as Part 1 or in a new solution.

5. Requirements Part 1a – Pass Grade (D)

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

5.1 Write 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 of the three 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 VS creates the starting class and names it "**Program**" saved as **Program.cs** in the project directory. This class contains a **Main** method. Create an object of **Pet** and call the methods of the object to get user input and display output as shown in the run example.

The code for the Main method starting all classes in Part 1 is also provided later in this document. If you need more help and guidance, step-by-step instructions for programming the Pet class are provided in a separate document (Assignmetn1Help) in Module 1 on Canvas.

6. Requirements Part 1b &1c: Album & TicketSeller – Pass Grade (C)

This part is to be done after you have completed and tested the **Pet** part

6.1 Create two new classes **TicketSeller** and **Album** in the same project and begin with writing the following instance variables.

```
class TicketSeller
{
        private string name;
        private double price = 100;
        private int numOfAdults;
        private int numOfChildren;

        private double amountToPay;
```

6.2 Complete the class with a **Start** method as well as other methods necessary to read input from and print output to the Console window, as in Part 1a. Go the Main method and write more code to create an object of **TicketSeller** and call the **Start** method.

6.3 Test your application and if everything goes well, start with the Album class and follow the same procedure.

```
public class Album
    {
        // Album name, artist name, and number of
tracks
        private string albumName;
        private string artistName;
        private int numOfTracks;
```

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

This part is only for grade A and B and can be skipped if you are not going for a higher grade.

This part is to be completed in addition to Part 1a and Part 1b. Using Visual Studio, you can create a new project inside the same solution of the previous part, or create a separate new project.

Look around yourself, at home, at your work, or wherever you are at this moment. You will find numerous objects, chairs, a baby, a TV, houses, or cars. Choose your **one** favorite object and program an application to represent the group of such objects. The object type does not have to be a physical thing; it can be a conceptual object such as Address, Calculator, etc.

For Grade B:

7.1 Create a class for the object type that you have chosen; include at least three instance variables as attributes, of which one of them must be a **double** (e.g. price, amount, of something). Determine an appropriate data type for each attribute.

7.2 Follow the same pattern as in Part 1 to solve the problem: Keep the Program class and the Main method as small as possible.

For Grade A:

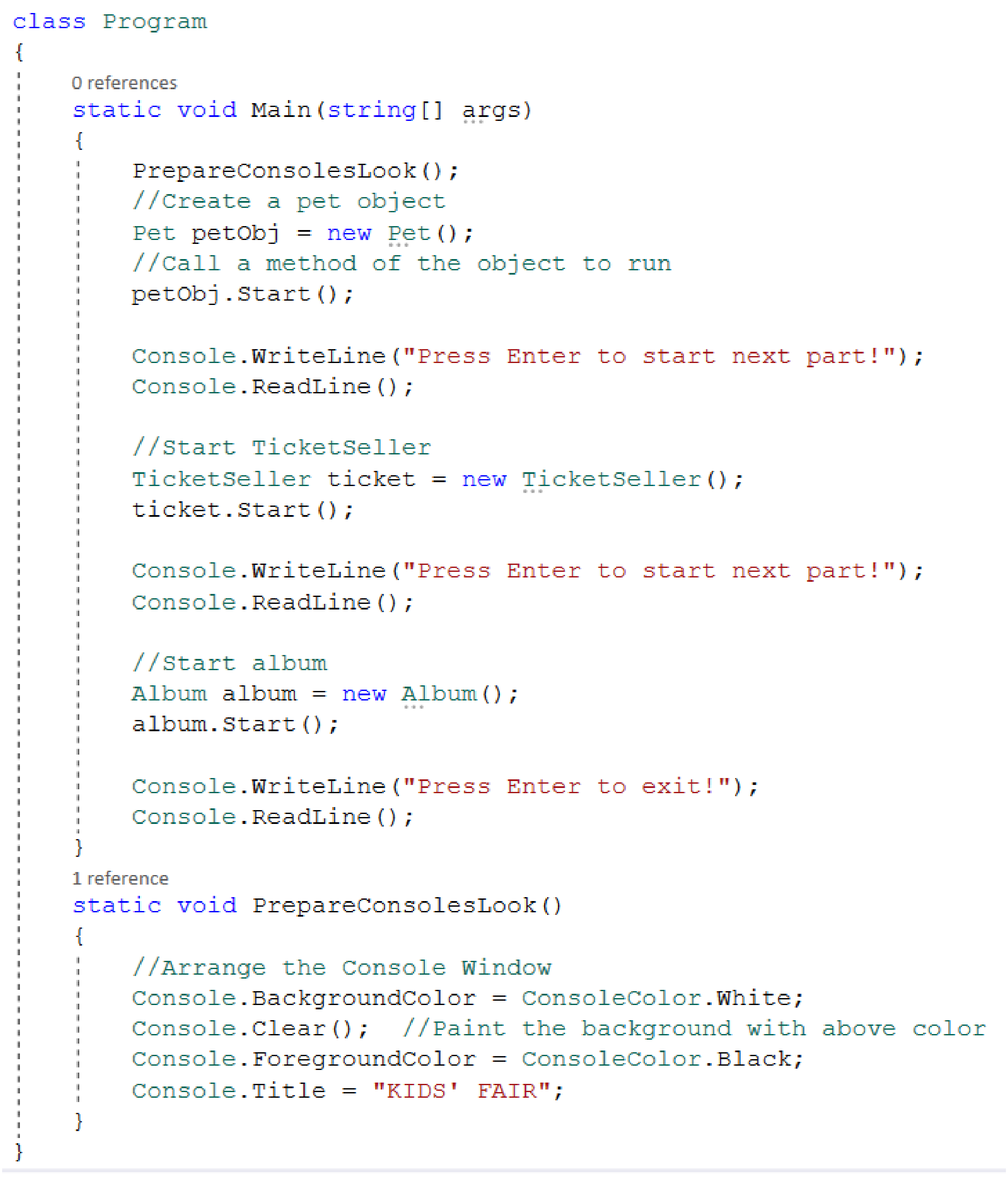
Use an instance of the **DateTime** as an extra instance variable in above class to store same date and time, or you can simply use it to show the current date and time.

8. Help and guidance - Part 1

8.1 The Program class and the Main method. You may use the code given below.

8.2 Work in steps and write one class a time. To test the class, copy that part of code from above that is necessary.

8.3 The Start method in each class should call other methods to read input, use input to calculate output and display it.



```
class Program
{
    0 references
    static void Main(string[] args)
    {
        PrepareConsolesLook();
        //Create a pet object
        Pet petObj = new Pet();
        //Call a method of the object to run
        petObj.Start();

        Console.WriteLine("Press Enter to start next part!");
        Console.ReadLine();

        //Start TicketSeller
        TicketSeller ticket = new TicketSeller();
        ticket.Start();

        Console.WriteLine("Press Enter to start next part!");
        Console.ReadLine();

        //Start album
        Album album = new Album();
        album.Start();

        Console.WriteLine("Press Enter to exit!");
        Console.ReadLine();
    }
    1 reference
    static void PrepareConsolesLook()
    {
        //Arrange the Console Window
        Console.BackgroundColor = ConsoleColor.White;
        Console.Clear();  //Paint the background with above color
        Console.ForegroundColor = ConsoleColor.Black;
        Console.Title = "KIDS' FAIR";
    }
}
```

8.4 Start Visual Studio and create a new Console Application. Visual Studio will prepare the solution and create the project structure including a start-up class, named Program. This class contains a definition for the Main method where you begin your coding. Give your application and your project appropriate names. Right-click on each item and select the Rename option.

The Pet class

Begin your coding by creating a new class. Right-click on the project name (**KidsFair**), select **Add**, and then **New Item** (or **Class** down the menu). If you go through the option **New Item**, you will find the option **Class** among the templates. Give a class name (Pet.cs), and click the button **Add**.

Declare the three variables:

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

Note: Be consequent in declaring classes either with the modifier public or without it with all your classes. For example, public class Pet or just class Pet. If you use public class with Pet then the program class must also be public. If you choose not to use public with your class names, the default access modifier internal will be applied by the compiler. This makes the class accessible for all classes in the same namespace. We will learn more about modifiers in the coming modules.

Comment your code to remember details the next time and make it easier for a reviewer to understand your thoughts. You can write comments above the variable or method declarations, but short comments can be written on the same line as in above image. Some prefer to write comments under a code line, but this is not very usual. What is important is that your programming style is consistent.

Now that you have variables to store the necessary data, the next step is to write methods. Methods are to do operations. In this class, we expect the objects of the class to perform two simple operations:

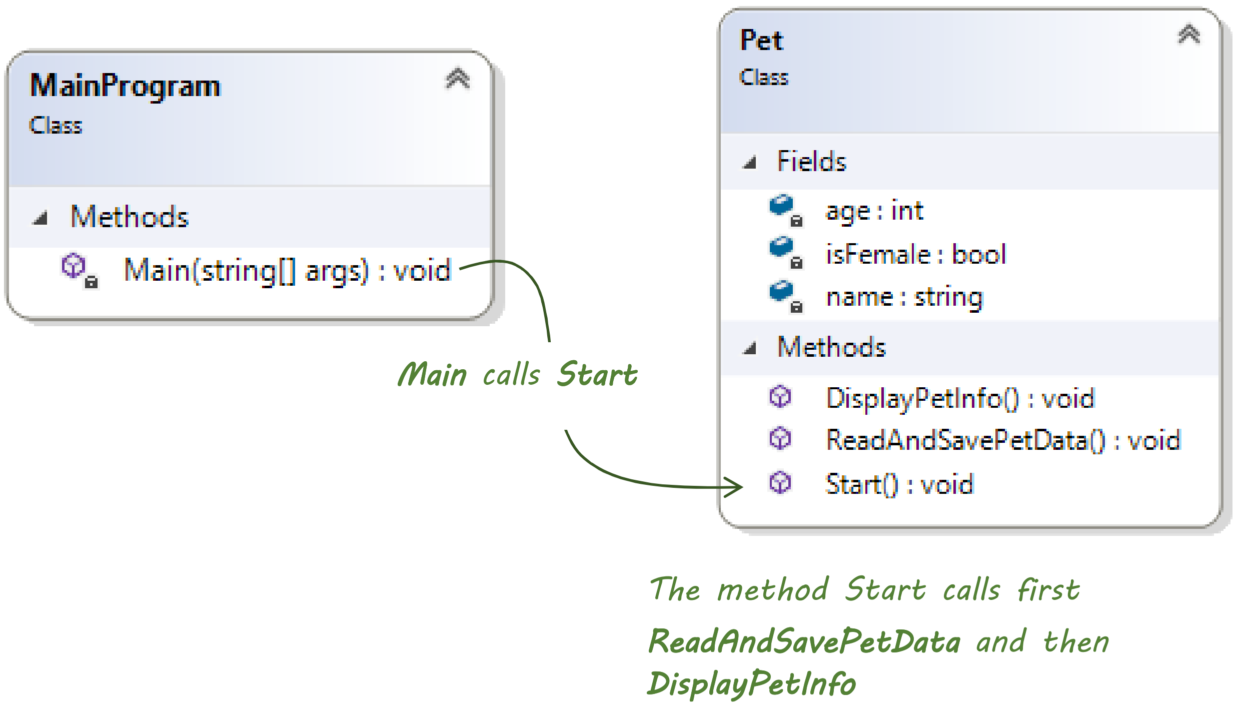
- Interact with the user and get values for the name, age and if the pet is female, and save the values in the variables of the object.
- Display the data saved in the object back to the user.

Write these two methods in the class **Pet**. Then go to the **Program** class and write code to create an object of the **Pet** class and call the above methods (see the above code snippet). .

The figure on the next page gives is a class diagram showing the instance variables and methods of the two classes (Pet and Program. You can then add a new class for the TicketSeller and Album parts to the project in next step.

You may of course call the two methods **ReadAndSavePetData** and **DisplayPetInfo** from the Main method (as in this construction), but to keep the Main method shorter and to let the **Pet** class control the flow of the program execution in a correct order, you can write a method in the Pet class that encapsulates the two calls. If we name this method **Start**, which may look like this:

```
public void Start ( )
{
    ReadAndSavePetData ( );
    DisplayPetInfo ( );
}
```



In order to give you a chance to think and try the solution by yourself, no more instruction is provided here. Instead, a step-by-step guidance is provided in a separate document. You can try solving the assignment by yourself before examining the solution in the help document.

9. Submission

Compress all your files, including the **Dependencies**, (**Properties, .NET Framework)** folder using a **zip** or **rar** format. Go the Assignments menu, Assignment 1 on the 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. In the event when the platform Canvas is down (does not happen often), wait until it is up and running again (even past the deadline).

Good Luck!

Programming is fun. Never give up. Ask for help!

Farid Naisan,
Course Responsible and Instructor