Arbeitsblatt 1. Erster Einstieg in C#

Hinweis zu den Übungsblättern: Es werden i.A. mehr Aufgaben enthalten sein, als Sie bzw. wir in der Übung schaffen werden. Der Rest ist (freiwillige) Hausaufgabe.

Nutzen Sie im Git (https://git.informatik.fh-nuernberg.de/albrecht-prog1/ws2024/Handouts) für diese Woche die beiden Handouts zu:

- Tastenkürzeln in Visual Studio
- Ein- und Ausgabe in C#

Visual Studio Projektstruktur und Einstellungen:

- Solutions, Projekte, Dateien
- Sinnvoll in der Übung: Eine Solution mit je einem Projekt pro Aufgabe.
- Auf den Laborrechnern am besten gleich das Standard-Verzeichnis für neue Projekte auf das Netzwerk-Home Z: legen (Menü Tools → Options → Projects and Solutions → Locations, dann bei Project Location "Z:\Documents\Visual Studio 2022\Projects")
- Automatische Code-Generierung abschalten (für's Lernen nicht sinnvoll!):
  - Tools → Options → Text Editor → Advanced → Responsive Code Completion
  - Tools → Options → Text Editor → Advanced → Word-based Suggestions
  - Tools → Options → IntelliCode alle Häkchen entfernen
- Debug → Options → General → Automatically close the console when debugging stops

Aufgabe 1.1. Erstes Programm: Hello, world!

a) Erstellen Sie eine neue Solution namens "Übung01" (Menü New→Project) mit einem Projekt namens "Hello".

   Tipp: Vermeiden Sie bei der Benennung von Programmdateien, Solutions und Projekten Leer- und Sonderzeichen einschließlich Punkt, Bindestricht etc. (außer "_"). Präziser: Verwenden Sie gültige C#-Bezeichner, denn aus diesen Namen werden u.U. automatisch Namespaces und Klassen-Namen generiert.

b) Geben Sie einen Text Ihrer Wahl aus, indem Sie einen Aufruf von Console.WriteLine() ergänzen bzw. den vorhandenen modifizieren.

c) Starten Sie Ihr Programm durch F5 (Debug→Start Debugging).

d) Starten Sie Ihr Programm nun durch STRG-F5 (Debug→Start without Debugging).

e) Bauen Sie als letzte Zeile den Funktionsaufruf

   ```
   Console.ReadKey();
   ```

   in Ihre Main-Funktion ein. Was bewirkt er? Was ist der Unterschied zu

   ```
   Console.ReadLine();
   ```

   Schauen Sie dafür in die Online-Dokumentation der Klasse Console (googlen "C# Console")

f) Ergänzen Sie einen sinnvollen Kommentar im Programm-Code. Hier ein Beispiel:

```
// Alles was nach den Slashes in einer Zeile steht, ist Kommentar
```

g) Erweitern Sie das Programm so, dass Sie einen Text, z.B. Ihren Namen, einlesen und wieder ausgeben! Schauen Sie im Handout für die Ein- und Ausgabe im Moodle, wie Console.ReadLine() benutzt wird oder googlen Sie das einfach.

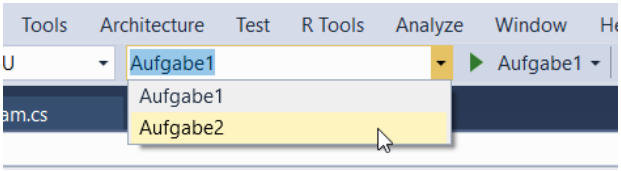
Aufgabe 1.2. Verzwickte Zeilenumbrüche

Erzeugen Sie in der vorhandenen Solution "Übung01" ein weiteres Projekt "Zeilenumbruch".

a) Kopieren Sie den hier abgebildeten Code in die geöffnete .cs-Datei:

```
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hi,\nwer geht");
        Console.Write("mit in");
        Console.Write("den\nBiergarten?");
    }
}
```

b) Schalten Sie in der Menü-Leiste das aktive Projekt um und starten Sie das Programm:



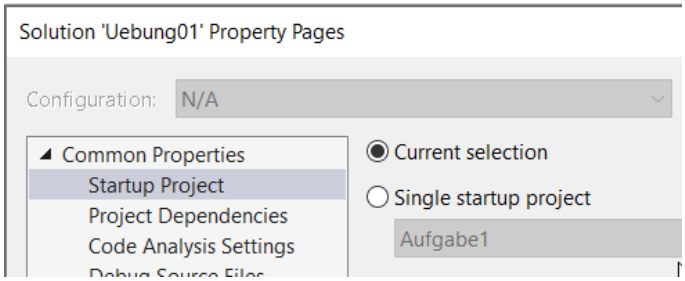
c) Was ist der Unterschied von Write und WriteLine?
Was bewirkt die Zeichenfolge \n?

d) Formulieren Sie die Anweisungen, um "Los geht's" an der aktuellen Cursor-Position (d.h. in der gleichen Zeile wie "Biergarten?") auszugeben.

e) Ändern Sie das Programm, so dass "Los geht's" zu Beginn einer neuen Zeile ausgegeben wird.

Tipp: Es empfiehlt sich, das Startverhalten in der Solution so anzupassen, dass immer die aktuelle Solution gestartet wird.

Rechtsklick auf die Solution im Solution Explorer → Properties → Startup Project



Aufgabe 1.3. Escape-Sequenzen

Schreiben Sie ein C#-Programm, das die nachfolgenden Zeilen exakt so ausgibt, wie hier angegeben. Zur Hilfe googeln Sie am besten nach "c# escape sequenzen" den entsprechenden MSDN-Link.

```
Die Datei finden Sie im Verzeichnis C:\uebung\c\neu\aufgabe1
Es gibt in C# das Hochkomma "'" und das Anführungszeichen """!
```

Aufgabe 1.4. Console-Klasse erkunden

Probieren Sie die Beispiele aus dem Handout zur Ein- und Ausgabe, um die Farbe der Ausgabe zu verändern. Mit Console.Clear() können Sie die Ausgabe löschen.

Schauen Sie in die Dokumentation der Console-Klasse:
https://docs.microsoft.com/de-de/dotnet/api/system.console

Welche Properties (Eigenschaften) können Sie neben der Farbe noch verändern?

Welche Methoden gibt es neben Console.ReadLine() und Console.WriteLine(), die Sie ausprobieren könnten?

Überlegen Sie sich weitere sinnvolle oder sinnlose Beispiele für einfache Programme und probieren Sie sie aus!