Versuchen Sie immer, Ihren Code zu kommentieren!

Aufgabe 1

Definiere ein Struct Vector3, der drei Felder X, Y und Z vom Typ double enthält, um einen 3D-Vektor zu repräsentieren. Erstelle anschließend zwei Variablen vom Typ Vector3 und fülle sie mit Werten. Gib anschließend die Werte beider Vektoren in der Konsole aus.

Aufgabe 2

Füge eine Methode namens AsString() zum Vector3 Struct hinzu, die einen String als Rückgabewert hat. Sie gibt die Werte des Vektors in folgendem Format zurück: „X, Y, Z“.
Nutze diese Methode um beide Vektoren auszugeben.

Aufgabe 3

Füge dem Vector3 Struct eine Methode namens Length hinzu, die die Länge des Vektors berechnet und zurückgibt. Die Länge eines Vektors berechnet sich wie folgt:

$$|\vec{a}| = \left| \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \right| = \sqrt{a_1^2 + a_2^2 + a_3^2}$$

Gib anschließend die Längen beider Vektoren aus.

Aufgabe 4

Füge der Struktur eine Methode mit dem Namen Multiply hinzu, die eine komponentenweise Multiplikation zweier Vektoren durchführt. Die Methode soll einen Vector3 als Parameter annehmen und einen neuen Vector3 zurückgeben.
Die Methodensignatur sieht wie folgt aus:

public Vector3 Multiply(Vector3 vector3)

Die **komponentenweise Multiplikation** zweier Vektoren bedeutet, dass die entsprechenden Komponenten der beiden Vektoren miteinander multipliziert werden

$$\mathbf{C} = \begin{pmatrix} C_x \\ C_y \\ C_z \end{pmatrix} = \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} \odot \begin{pmatrix} B_x \\ B_y \\ B_z \end{pmatrix} = \begin{pmatrix} A_x \cdot B_x \\ A_y \cdot B_y \\ A_z \cdot B_z \end{pmatrix}$$

Aufgabe 5

Füge zwei weitere Methoden zur Addition und Subtraktion von Vektoren hinzu. Die Signaturen unterscheiden sich nur im Namen von der Multiplikation. Finde selbst heraus, wie Vektoren addiert und subtrahiert werden.

Aufgabe 6

Lesen Sie die Datei **vector3.csv**, die sich im selben Ordner wie diese Aufgabe befindet, mit den Ihnen bekannten Methoden der Klasse **File** ein. Zerlegen Sie anschließend jede Zeile der CSV-Datei in die drei Koordinaten (x, y, z). Speichern Sie diese Werte in einem **Vector3**-Struct und fügen Sie das Struct in ein Array von **Vector3**-Structs ein. Das Array muss die gleiche Länge haben wie die Anzahl der Zeilen in der CSV-Datei, abzüglich der ersten Zeile (Kopfzeile), die ignoriert werden muss. Geben Sie anschließend mit der Methode **AsString()** die Werte aller **Vector3**-Structs in Ihrem Array in der Konsole aus.

Aufgabe 7

Schreiben Sie ein Programm, welches einen Jahreskalender aus der folgenden Struktur erstellt. Erstellen Sie dazu ein Array mit 365 Elementen vom Typ der Struktur Date und speichern Sie jeden Tag in dem Array. Gehen Sie davon aus, dass der Februar immer 28 Tage hat.

```
struct Date
{
        public int Day;
        public int Month;
        public int Year;
}
```

Das Programm soll z.B. folgende Ausgabe erzeugen:

```
Tag: 1 Monat: 1 Jahr: 2011
Tag: 2 Monat: 1 Jahr: 2011
...
Tag: 31 Monat: 1 Jahr: 2011
Tag: 1 Monat: 2 Jahr: 2011
Tag: 2 Monat: 2 Jahr: 2011
...
Tag: 30 Monat: 12 Jahr: 2011
Tag: 31 Monat: 12 Jahr: 2011
```

Aufgabe 8

Für ein Spiel wird eine Tabelle mit den Namen und Punktesständen von Spielern benötigt. Definieren Sie zunächst eine Struktur namens Gambler, die den Namen und die Punkte eines Spielers speichern kann und legen Sie dann ein Array für maximal 10 Spieler an.
Das Programm liest bis zu 10 Spielernamen und ihre Punkte ein. Das Array soll gemäß den Punkten absteigend sortiert sein. Anschließend werden die Spieler mit ihren Punkten tabellarisch angezeigt.

Das Programm soll z.B. folgende Ausgabe erzeugen:

```
Bitte gib deinen Namen ein: Peter
Bitte gib deine Punkte ein: 50
```

Bitte gib deinen Namen ein: Hans
Bitte gib deine Punkte ein: 21
Bitte gib deinen Namen ein: Marie
Bitte gib deine Punkte ein: 14
Bitte gib deinen Namen ein: Franzi
Bitte gib deine Punkte ein: 61
Bitte gib deinen Namen ein: Jürgen
Bitte gib deine Punkte ein: 23
Bitte gib deinen Namen ein: Abdul
Bitte gib deine Punkte ein: 51
Bitte gib deinen Namen ein: Alex
Bitte gib deine Punkte ein: 12
Bitte gib deinen Namen ein: Jule
Bitte gib deine Punkte ein: 72
Bitte gib deinen Namen ein: Phil
Bitte gib deine Punkte ein: 46
Bitte gib deinen Namen ein: Markus
Bitte gib deine Punkte ein: 9

Name	Punkte
Jule	72
Franzi	61
Abdul	51
Peter	50
Phil	46
Jürgen	23
Hans	21
Marie	14
Alex	12
Markus	9

Aufgabe 9

Schreibe ein Programm, das das Alter von zwei Personen vergleicht. Erfasse für jede Person das Geburtsdatum und berechne, wie viele Jahre, Monate und Tage sie voneinander unterscheiden. Gib das Ergebnis in einem verständlichen Format aus, das die genaue Altersdifferenz angibt.

- Erfasse die Geburtsdaten der beiden Personen als **DateTime** (DateTime.TryParse).
- Berechne die Differenz der beiden Daten (**TimeSpan**).
- Verwende die berechnete Differenz, um die Anzahl der Jahre, Monate und Tage zu bestimmen.
- Gib die Altersunterschiede mit einer präzisen Angabe der Jahre, Monate und Tage aus.

Aufgabe 10

Schreiben Sie ein Programm, das den Zeitplan einer Flugreise berechnet. Der Benutzer gibt die Abflugzeit und die Flugdauer an. Das Programm soll die Ankunftszeit berechnen und auf der Konsole ausgeben. Zeitzonenunterschiede werden dabei nicht berücksichtigt, das heißt, die Ankunftszeit wird in derselben Zeitzone wie die Abflugzeit angegeben.

- Erfassen Sie die Abflugzeit als **DateTime**
- Erfassen Sie die Flugdauer als **TimeSpan**
- Berechnen Sie die Ankunftszeit, indem Sie die Flugdauer zur Abflugzeit addieren.
- Geben Sie die Ankunftszeit auf der Konsole aus.