Spicker C#

Variablen

```
boolean flag = true/false;              //Speicher Boolean für einzelnes Bit
sbyte zahl;                             //-128 zu 127
byte zahl;                              // 0 bis 255
short zahl;                             //-32,768 bis 32,767
ushort zahl;                            // 0 bis 65,535
int zahl;                               //-2,147,483,648 bis 2,147,483,647
uint zahl;                              //0 bis 4,294,967,295
long zahl;                              //-9,223,372,036,854,775,808 bis 9,223,372,036,854,775,807
ulong zahl;                             // 0 bis 4,294,967,295

float kommazahl;                        //Speicher für Kommazahl
double kommazahl;                       //Speicher für Kommazahl mit doppelter Genauigkeit

char buchstabe;                         //Speicher Char für Buchstabe
string zeichenkette;                    //Speicher String für Wort

var zeichenketteOderZahl                //Speicher String für Wort
```

public: Auf den Typ oder Member kann von jedem anderen Code in derselben Assembly oder einer anderen Assembly, die darauf verweist, zugegriffen werden. Die Zugriffsebene öffentlicher Member eines Typs wird durch die Zugriffsebene des Typs selbst gesteuert.
private: Auf den Typ oder Member kann nur über Code in derselben Klasse oder Struktur zugegriffen werden.
protected: Auf den Typ oder Member kann nur über Code in derselben Klasse oder in einer von dieser Klasse abgeleiteten Klasse zugegriffen werden.
internal: Auf den Typ oder Member kann von beliebigem Code in derselben Assembly zugegriffen werden, jedoch nicht von einer anderen Assembly. Mit anderen Worten, auf interne Typen oder Member kann von Code aus zugegriffen werden, der Teil derselben Kompilierung ist.
protected internal: Auf den Typ oder Member kann von jedem Code in der Assembly zugegriffen werden, in der er deklariert ist, oder von innerhalb einer abgeleiteten Klasse in einer anderen Assembly.
private protected: Auf den Typ oder Member kann von Typen zugegriffen werden, die von der Klasse abgeleitet sind, die in ihrer enthaltenden Assembly deklariert sind.

```
NameObjekt namen = new NameObjekt { intVar = 1961, stringVar = "Wostock" };      //Objekt erstellen
```

Array & Liste

```
Funktion[] NameArray = new Funktion[5];         //Array erstellen
List<Funkt> NameListe = new List<Funkt>();      //Liste Erstellen
NameListe.Add(NameSpeicher);                    //Objekt oder Element in die Liste
NamenListe.Remove("Anton");                     //Entfernt ein Element
NamenListe.Clear();                             //Entfernt alle Elemente
NamenListe.Count;                               //Zählt die Anzahl Elemente
NamenListe.Insert(10, "Anton");                 //Fügt ein Element an Position X ein
NamenListe.RemoveAt(10);                        //Löscht ein Element an Position X
```

Objekt Methoden Klassen

```
NameListe obiektName = new NameListe { ParmaeterInt = 1961, ParmateString = "Wostock" };
                                                //Objekt erstellen
class Klassennamen                              //Beispiel Einer Klasse
    {
        public int Nummern;                     //Integer in Klasse
        public string Zeichenkette;             //String in Klasse
        private int einsatzbis;                 //Private Variable für Klassenintern
        public int EinsatzBis
        {
            get => einsatzbis;                  //Getter
            set                                 //Setter
            {
                einsatzbis = value;             //Eingabe wird übbergeben
                if (einsatzbis == 0)            //Abfrage
                {
                    einsatzbis = DateTime.Today.Year;
                }
            }
        }
        public Land Land;                       //Verweist auf andere Klasse
        public int Einsatzdauer => EinsatzBis - EinsatzVon;
```

```
    }                                           //Integer Klasse Rechnen

class TestKlasse                                //Klasse Schreibgeschützt
{
        readonly public string nachricht;       //Readonly Variablen
        public TestKlasse(string nachricht)
        {
                this.nachricht = nachricht;     //Nachricht kann nur einmal geschrieben werden
        }
}
```

Konsole

```
Console.ForegroundColor = ConsoleColor.Blue;    //Schriftfarbe wechseln
Console.BackgroundColor = ConsoleColor.Magenta; //Hintergrundfarbe wechseln
Console.ResetColor();                           //Farbe Reset
Console.Write("Hallo Welt");                    //Schreibe Hallo Welt
Console.WriteLine("Hallo Welt");                //Schreibe Hallo Welt auf neue Linie
Console.SetCursorPosition(20, 20);              //Cursor Position Setzen
Console.Clear();                                //Konsole löschen
zahl = int.Parse(Console.ReadLine());           //Einlesen Zahl
wort = Console.ReadLine();                      //Einlesen Wort
buchstabe = Console.ReadKey(true).KeyChar;      //Einlesen Buchstabe in char speichern
Console.ReadKey(true);                          //Beliebige Taste drücken
```

Schleifen

```
foreach(var name in namenListe)                 //Für Jedes Element in der Lise einmal ausführen
        {
        }

for (i = 1; i <= anzahl; i++)                   //Schleife Anzahl Durchläufe bekannt
        {
        }

while (flag == false)                           //Schleife Anzahl Durchläufe nicht bekannt
        {                                       //Abfrage zu Beginn der Schleife
        }

do      {                                       //Schleife Anzahl Durchläufe nicht bekannt
        }                                       //Abfrage am Ende der Schleife
while (flag == false);                          //Wird minimal 1x ausgeführt
```

Abfragen

```
if(abfrage 1)   {                               //if Abfrage
                }
else if(abfrage 2)                              //Zweite if Abfrage wird nur Ausgeführt wenn vorherige
                {                               //if Abfrage nicht ausgeführt wurde (optional)
                }
else            {                               //else wird bei dem nichtausführen der if Ausgeführt
                }                               //(optional)
```

Download

```
WebClient Client = new WebClient();             //Webclient für Downloads
Client.DownloadFile("https://www.google.com/","C:/Webseiten/nameFile.html");
                                                //Ladet Seite Herunter
string reply = Client.DownloadString ("https://www.google.com/");
                                                //Ladet Quelltext in String Herunter

try             {                               //Try um eine Seite Herunterzuladen
                }
catch (System.Net.WebException ex)              //Bei nicht gelingen des Herunterladen ausgeführt
                {
                Console.WriteLine(ex.Message);  //Gibt Fehlernachricht Aus
                }
```

Zusätzlich

```
int anzahlBuchstaben = stringName.Length;       //Wortlänge in Integer Speichern
Thread.Sleep(8000);                             //Programm wartet x Millisekunden
string string1Gross = string1.ToUpper();        //Speichert String in Grossbuschstaben
```

```
string string1Klein = string1.ToLower();        //Speichert String in Kleinbuchstaben
bool x = wortString.Contains("wort");           //Abfrage ob String Zeichenkette enthält true/false
bool x = wortString.EndsWith("wort")            //Abfrage ob String mit Zeichenkette endet true/false
bool x = wortString.StartsWith("wort")          //Abfrage ob String mit Zeichenkette endet true/false
string wort = number.ToString();                //Wandelt Zahl in Integer um
bool name string.IsNullOrEmpty("wort")          //Abfrage ob String 0 oder Leer ist
```

Zeit mit DateTime

```
DateTime dt = DateTime.Now;                     //Datum & Zeit
string zeit = dt.ToString("HH:mm:ss");          //Beispiel
d ->    //Represents the day of the month as a number from 1 through 31.
dd ->   //Represents the day of the month as a number from 01 through 31.
ddd->   //Represents the abbreviated name of the day (Mon, Tues, Wed, etc).
dddd->  //Represents the full name of the day (Monday, Tuesday, etc).
h->     //12-hour clock hour (e.g. 4).
hh->    //12-hour clock, with a leading 0 (e.g. 06)
H->     //24-hour clock hour (e.g. 15)
HH->    //24-hour clock hour, with a leading 0 (e.g. 22)
m->     //Minutes
mm->    //Minutes with a leading zero
M->     //Month number(eg.3)
MM->    //Month number with leading zero(eg.04)
MMM->   //Abbreviated Month Name (e.g. Dec)
MMMM->  //Full month name (e.g. December)
s->     //Seconds
ss->    //Seconds with leading zero
t->     //Abbreviated AM / PM (e.g. A or P)
tt->    //AM / PM (e.g. AM or PM)
y->     //Year, no leading zero (e.g. 2015 would be 15)
yy->    //Year, leading zero (e.g. 2015 would be 015)
yyy->   //Year, (e.g. 2015)
yyyy->  //Year, (e.g. 2015)
```