Klasse: Name: 09.05.17

Aufgabe 1: (20 Punkte)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace testklassenarbeit1_Aufg3
{
    class Program
    {
        // Zweidimensionales Array für die Sitzplätze
        static bool [,] Plaetze = new bool [11,50];

        //-------------------------------------------------------------------------
        // Hier wird einfach ein konstanter Wert zum Testen der Lösung zurückgegeben
        //
        // Ausserdem wird das Array mit zufälligen Werten belegt.
        // d.h. dass alle Sitzplätze mit belegt (false) und frei (true) vergeben werden
        //-------------------------------------------------------------------------
        static int hoechstePlatznummer(int Kategorie, int VeranstaltungsID)
        {
            Random rnd = new Random();

            for (int i=0; i < 11; i++)
                for (int j = 0; j < 50; j++)
                {
                    Plaetze[i,j] = Convert.ToBoolean(rnd.Next(0,2));
                }

            return 550;
        }

        //-------------------------------------------------------------------------
        // Liefert TRUE, wenn der Platz mit der übergebenen
        // Platznummer, VeranstaltungsID und Kategorie existiert und frei ist,
        // andernfalls FALSE.
        //-------------------------------------------------------------------------
        static bool istfrei(int Platznummer, int Kategorie, int VeranstaltungsID)
        {
            int Reihe, Platz;

            Reihe = Platznummer / 100 - 1;  // Ausrechnen der Reihe (Zeile ermitteln)
            Platz = Platznummer % 100 - 1;  // Ausrechnen des Sitzplatzes der Reihe
                                                // (Spalte ermitteln)
            return Plaetze[Reihe,Platz];    // aus Array ermitteln, ob Platz belegt ist
        }
```

Klasse: Name: 09.05.17

```
//-------------------------------------------------------------------------------------
// Lösung der Aufgabenstellung
// andere Lösungsmöglichkeiten nicht ausgeschlosen
//-------------------------------------------------------------------------------------
static int nZusammenhaengendeFreiePlaetze(int Kategorie, int VeranstaltungsID, int Anzahl)
{
        int Anzahl_Reihen;
        int Reihen = 0;
        int Anzahl_freiePlaetze = 0;
        int PlatzNr;

        Anzahl_Reihen = hoechstePlatznummer(1, 1) / 50;
        PlatzNr = 0;

        do
        {
            PlatzNr = 0;
            do
            {
                if (Plaetze[Reihen, PlatzNr] == true)
                {
                    Anzahl_freiePlaetze++;
                }
                else
                {
                    Anzahl_freiePlaetze = 0;
                }
                PlatzNr++;

            } while ((Anzahl_freiePlaetze < Anzahl) && (PlatzNr < 50));
            Reihen++;
        }
        while ((Reihen < Anzahl_Reihen) && (Anzahl_freiePlaetze < Anzahl));

        if (Anzahl_freiePlaetze < Anzahl)
        {
            return 0;
        }
        else
        {
            return Reihen * 100 + PlatzNr - Anzahl + 1;
        }
}

//-------------------------------------------------------------
// in Main wird getestet, ob der Lösungsansatz richtig ist.
//-------------------------------------------------------------
static void Main(string[] args)
{
    int Platzfrei;                  // erster Platz der zusammenhängenden Plätze
    int Anzahl_freiePlaetze = 6;    // 6 zusammenhängede Plätze suchen

    Platzfrei = nZusammenhaengendeFreiePlaetze(1, 1, Anzahl_freiePlaetze);
    Console.WriteLine("Ab Platznummer {0} sind {1} Plätze frei!", Platzfrei, Anzahl_freiePlaetze);
    Console.ReadKey();
}

}
}
```

Bewertungskriterien:

Zwei Schleifen (2-dim. Array) **8**
- über die Anzahl Reihen
- über die Anzahl Spalten

In der jeweiligen Spalte jeweils zusammenhängende Plätze suchen und finden **6**

Berechnung der 1. Platznummer dieser zusammenhängenden Plätze **6**

Klasse: Name: 09.05.17

Aufgabe 2.1:

```
public int Max(int[] max_Arr)                    // Syntax:       2 Punkte
{
    int Maximum = 0;

    for (int i=0; i < max_Arr.Length; i++)       // Algorithmus:8 Punkte
    {
        if (Maximum < max_Arr[i])
        {
            Maximum = max_Arr[i];
        }
    }
    return Maximum;
}
```

Aufgabe 2.2: Unterschied beschreiben: 3 Punkte

Array kann nur ein Datentyp annehmen
Es muss eine feste Länge vorgeben werden
Es kann nicht einfach ein Wert eingefügt werden, normalerweise werden Werte angehängt.

Arraylist ist ein Objekt und wie der Name schon sagt eine Liste, also eine Kette, wo der Vorgänger und wo der Nachfolger festgelegt ist (über Zeiger). Deshalb kann leicht ein Wert eingefügt werden.
Arraylist kann auch deshalb dynamisch erweitert werden.

Aufgabe 2.3: Das besondere ist der Index: 3 Punkte

Es sind beide Collections, also Listen.
Jedoch kann bei der ArrayList beliebige Datentypen und Objekte eingefügt werden.
Bei der generischen Liste List muss man sich auf einen Datentyp oder Objekt festlegen.

Klasse: Name: 09.05.17

Aufgabe 2.4:

```
namespace arraybsp
{
  class Program
  {
      static void Main(string[] args)
    {
      // initialisiert die Zufallsklasse
      Random Rnd = new Random();

      List<int> myArr = new List<int>(); // anstatt Array: 1 Punkt

      // Eingabe der Array-Größe
      Console.Write("Geben Sie die Anzahl der Elemente ein: ");
      int number = Convert.ToInt32(Console.ReadLine());

      // Initialisierung des Arrays
      // ist nicht mehr nötig myArr = new int[number];

      // jedes Element des Arrays in einer Schleife durchlaufen
      for(int i=0; i<number; i++)
      {
        // dem Array-Element einen Wert zuweisen
        myArr.Add(Rnd.Next() ); // mit Add Objekt anhängen: 1 Punkt

        // das Array-Element an der Konsole ausgeben
        Console.WriteLine("myArr[{0}] = {1}",i, myArr[i]); // *)
      }
      Console.WriteLine("Maximum: {0}", Max(myArr));
      Console.WriteLine("Mittelwert: {0}", AVR(myArr));

      Console.ReadLine(); // Nur zum Anhalten
    }
  }
}
```

*) Diese Ausgabe ist auch bei Listen möglich! ***Ausgabe der Listenwerte: 1 Punkt***

Klasse: Name: 09.05.17

Aufgabe 3: (20 Punkte)

Die Klasse Messwert muss mit dem Interface „IComparable“ (4P)versehen werden und muss mit der Methode „CompareTo() erweitert werden (16P)

```
class Messwert : IComparable
{
    public string Name
    {
        set;
        get;
    }

    public DateTime Messzeitpunkt
    {
        set;
        get;
    }

    public int Temperatur
    {
        set;
        get;
    }

    public double Windgeschwindigkeit
    {
        set;
        get;
    }

    int IComparable.CompareTo(object obj)
    {
        Messwert mw = (Messwert) obj;

        if (mw.Temperatur > this.Temperatur)
                return 1;
        if (mw.Temperatur < this.Temperatur)
            return -1;
        else
            return 0;
    }
}
```

Klasse: Name: 09.05.17

Hauptprogramm zum Testen der Methode Sort():

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Messwerte_sortieren
{
    class Program
    {
        static void Main(string[] args)
        {

            List<Messwert> messwerte = new List<Messwert>();
            Random rnd1 = new Random();
            Random rnd2 = new Random();
            DateTime meindatum = System.DateTime.Now;

            Console.WriteLine("DateTime: {0}", meindatum);

            for (int i = 0; i < 10; i++)
            {
                Messwert Messung = new Messwert();
                Messung.Name = "Messung" + i.ToString();
                meindatum = meindatum.AddDays(1);// AddHours(1);
                Messung.Messzeitpunkt = meindatum;
                Messung.Temperatur = rnd1.Next(-20, 40);
                Messung.Windgeschwindigkeit = rnd2.Next(0, 200);
                messwerte.Add(Messung);

                Console.WriteLine("Station: {0}, Datum und Uhrzeit: {1}, Temperatur: {2}, Wind:
{3}",Messung.Name,Messung.Messzeitpunkt,Messung.Temperatur,Messung.Windgeschwindigkeit);

            }
            messwerte.Sort();

            Console.WriteLine();
            foreach (Messwert Ms in messwerte)
            {
                Console.WriteLine("Station: {0}, Datum und Uhrzeit: {1}, Temperatur: {2}, Wind:
{3}", Ms.Name, Ms.Messzeitpunkt, Ms.Temperatur, Ms.Windgeschwindigkeit);
            }

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