```
1  using System.Xml.Linq;
2
3  namespace DAMLib
4  {
5      public class Set<T>
6      {
7          private T[] _set;
8          private bool _testAtribute;     // Atributo utilizado en la
             funcion EqualsDeep()
9
10
11         public bool IsEmpty => _set.Length == 0;
12         public int Count
13         {
14             get
15             {
16                 if (_set == null)
17                     return 0;
18                 else
19                     return _set.Length;
20             }
21         }
22
23
24         // Constructor sin parametros
25         public Set()
26         {
27             _set = new T[0];
28         }
29
30
31         // Funcion que añade un elemento SOLO en caso que no exista
             dentro de la coleccion.
32         public void Add(T element)
33         {
34             if (element == null)
35                 return;
36
37             if (!Contains(element))
38             {
39                 int count = _set.Length;
40                 T[] setResult = new T[count + 1];
41
42                 for (int i = 0; i < count; i++)
43                 {
44                     setResult[i] = _set[i];
45                 }
46
47                 setResult[count] = element;
48                 _set = setResult;
49             }
50         }
51
```

```
52          // Funcion que devuelve verdadero si existe el elemento dentro ↵
              de la coleccion.
53          public bool Contains(T element)
54          {
55              return IndexOf(element) >= 0;
56          }
57
58          // Funcion que devuelve el índice del elemento que le paso por ↵
              parametros.
59          public int IndexOf(T element)
60          {
61              if (element == null)
62                  return -1;
63
64              for (int i = 0; i < _set.Length; i++)
65              {
66                  if (_set[i].Equals(element))
67                      return i;
68              }
69
70              return -1;
71          }
72
73          // Funcion que elimina el elemento que le pasamos por          ↵
              parametros.
74          public void Remove(T element)
75          {
76              if (element == null)
77                  return;
78
79              int index = IndexOf(element);
80
81              if (index == -1)
82                  return;
83
84              int count = _set.Length;
85              T[] arrayResult = new T[count - 1];
86
87              // Posibilidad 1. Con dos bucles 'for'.
88              for (int i = 0; i < index; i++)
89              {
90                  arrayResult[i] = _set[i];
91              }
92
93              for (int i = index; i < count - 2; i++)
94              {
95                  arrayResult[i] = _set[i + 1];
96              }
97
98              // Posibilidad 2. Con instruccion 'continue'.
99              /*
100             for(int i = 0; i < count; i++)
101             {
```

```
102                 if (i == index)
103                     continue;
104                 arrayResult[i] = _set[i];
105             }
106             */
107
108             _set = arrayResult;
109         }
110
111
112         public void Clear()
113         {
114             _set = Array.Empty<T>();
115         }
116
117         public override int GetHashCode()
118         {
119             return 133 * 533 * 224 * _testAtribute.GetHashCode();
120         }
121
122         public override bool Equals(Object? obj)
123         {
124             return this == obj;
125         }
126
127         public bool IsEqualsInDeep(object? obj)
128         {
129             if (this == obj)
130                 return true;
131
132             if (obj is not TestCar)
133                 return false;
134
135             TestCar car = (TestCar)obj;
136
137             return (this._testAtribute == car.TestAtribute);
138         }
139
140         // Funcion que devuelve un string con todos los elementos de la
               coleccion.
141         public override string ToString()
142         {
143             string result = "";
144             int count = 0;
145
146             foreach (T element in _set)
147             {
148                 result += $"El elemento numero {count} de la coleccion
                      es: {element}.\n";
149                 count++;
150             }
151
152             return result;
```

```
153             }
154         }
155     }
```