```
 1  namespace DAMLib
 2  {
 3      public class Stack<T>
 4      {
 5          private T[] _stackArray;
 6
 7          public bool IsEmpty => _stackArray.Length == 0;
 8          public int Count
 9          {
10              get
11              {
12                  if (_stackArray == null)
13                      return 0;
14                  else
15                      return _stackArray.Length;
16              }
17          }
18
19          public Stack()
20          {
21              _stackArray = new T[0];
22          }
23
24          // Funcion que introduce un elemento generico en el Stack.
25          public void Push(T element)
26          {
27              if (element == null)
28                  return;
29
30              int count = _stackArray.Length;
31
32              T[] stackResult = new T[count + 1];
33
34              for (int i = 0; i < count; i++)
35              {
36                  stackResult[i] = _stackArray[i];
37              }
38
39             stackResult[count] = element;
40              _stackArray = stackResult;
41          }
42
43          // Funcion que extrae un elemento generico del Stack.
44          public T? Pop()
45          {
46              if (_stackArray == null)
47                  return default(T);
48
49              int count = _stackArray.Length;
50
51              T result = _stackArray[count - 1];
52              T[] stackResult = new T[count - 1];
53
```

```
54              for(int i = 0; i < count - 1; i++)
55              {
56                  stackResult[i] = _stackArray[i];
57              }
58
59              _stackArray = stackResult;
60
61              return result;
62          }
63
64          // Funcion que devuelve el elemento superior del Stack.
65          public T? Top()
66          {
67              if (_stackArray == null)
68                  return default(T);
69
70              T result = _stackArray[_stackArray.Length - 1];
71
72              return result;
73          }
74
75          public T[] Clone()
76          {
77              int size = _stackArray.Length;
78              T[] clone = new T[size];
79              for (int i = 0; i < size; i++)
80              {
81                  clone[i] = _stackArray[i];
82              }
83              return clone;
84          }
85
86          public void Clear()
87          {
88              _stackArray = new T[0];
89          }
90
91          public override string ToString()
92          {
93              string result = "";
94              int count = 0;
95
96              foreach (T element in _stackArray)
97              {
98                  result += $"El elemento {count} del Stack es: {element} ↵
                      \n";
99                  count++;
100             }
101
102             return result;
103         }
104     }
105 }
```