


```
1 using System;
2
3 namespace ResumenFunciones
4 {
5     public class Functions
6     {
7
8         public static bool IsMinor(int number1, int number2)
9         {
10             return number1 < number2;
11         }
12
13
14         // Funcion Sumatory con un bucle WHILE
15         public static int GetSumatoryWhile(int number)
16         {
17             int result = 0;
18             int i = 1;
19
20             while (i <= number)
21             {
22                 result += i;
23                 i++;
24             }
25             return result;
26         }
27
28         // Funcion Sumatory con un bucle FOR
29         public static int GetSumatoryFor(int number)
30         {
31             int result = 0;
32             for (int i = 1; i <= number; i++)
33             {
34                 result += i;
35             }
36             return result;
37         }
38
39         // Funcion auxiliar que devuelve verdadero si es un número par
40         public static bool IsEven(int number)
41         {
42             if (number % 2 == 0)
43                 return true;
44             return false;
45         }
46
47         // Funcion auxiliar que devuelve verdadero si es un numero impar
48         public static bool IsOdd(int number)
49         {
50             return (number % 2 != 0);
51         }
52
```

```
53 // Funcion Secuencia con un bucle WHILE
54 public static void GetSequenceWhile(int number)
55 {
56     int count = 0;
57     int result = 0;
58     int resultCounter = 0;
59
60     while (resultCounter < number)
61     {
62         if (IsEven(count))
63         {
64             result = count * 2 * (1);
65         }
66         else
67         {
68             result = count * 2 * (-1);
69         }
70         Console.WriteLine(result);
71         resultCounter = Math.Abs(result);
72         count++;
73     }
74 }
75
76 // Funcion Secuencia con un bucle FOR
77 public static void GetSequenceFor(int number)
78 {
79     int result;
80
81     for (int i = 1; i <= number / 2; ++i)
82     {
83         if (IsEven(i))
84         {
85             result = i * 2 * (1);
86         }
87         else
88         {
89             result = i * 2 * (-1);
90         }
91
92         Console.WriteLine(result);
93     }
94 }
95
96 // Funcion Secuencia con una funcion reservada MATH.POW
97 public static void GetSequence3(int number)
98 {
99     int result = -2, printNumber, count = 0;
100
101     while (result < number)
102     {
103         result += 2;
104         printNumber = result * (int)Math.Pow(-1, count);
105         Console.WriteLine(printNumber);
```

```
106         count++;
107     }
108 }
109
110 #region Funciones Repaso
111 // FUNCION que devuelve el menor de dos numeros
112 public static int GetMinor(int number1, int number2)
113 {
114     if (number1 <= number2)
115         return number1;
116     return number2;
117 }
118
119 // FUNCION que devuelve el menor de tres numeros
120 public static int GetMinest(int number1, int number2, int number3)
121 {
122     if (number1 <= number2 && number1 <= number3)
123         return number1;
124     if (number2 <= number3)
125         return number2;
126     else
127         return number3;
128 }
129
130 // FUNCION que imprime por consola la serie 0, -2, 4, -6, 8
131 public static void PrintSerie(int number)
132 {
133     int result = 0;
134
135     for (int i = 0; i <= number / 2; i++)
136     {
137         if (IsEven(i))
138             result = (2 * i);
139         else
140             result = (-2 * i);
141         Console.WriteLine(result);
142     }
143 }
144
145 // FUNCION que devuelve verdadero si un numero es primo
146 public static bool IsPrime(int number)
147 {
148     for (int i = 2; i < number; i++)
149     {
150         if (number % i == 0)
151             return false;
152     }
153     return true;
154 }
155
156 //FUNCION que imprime en consola los valores de la serie Collatz
```

```
157     public static void Collatz0(int number)
158     {
159         int result = number;
160         Console.WriteLine(result);
161         while (result != 1)
162         {
163             if (IsEven(result))
164                 result /= 2;
165             else
166                 result = (result * 3) + 1;
167             Console.WriteLine(result);
168         }
169     }
170
171
172     // FUNCION que devuelve el sumatorio de un numero
173     public static int GetSummatory(int number)
174     {
175         int result = 0;
176         for (int i = 1; i <= number; i++)
177         {
178             result += i;
179         }
180         return result;
181     }
182
183
184     //FUNCION que devuelve el productorio de un numero
185     public static int GetProductory(int number)
186     {
187         int result = 1;
188         for (int i = 1; i <= number; i++)
189         {
190             result *= i;
191         }
192         return result;
193     }
194     #endregion
195
196     // Funcion que devuelve si un numero es Primo
197     // TODO: Numero Primo
198     public static bool IsPrime0(int number)
199     {
200         for (int i = 2; i < number; i++)
201         {
202             if (number % i == 0)
203                 return false;
204
205             // ES UN ERROR ELIMINATORIO ESCRIBIR UN RETURN DENTRO DE UN BUCLE FOR!!!!
206             // return true;
207         }
208         return true;
```

```
209     }
210
211     // Funcion que concatena strings y devuelve series
212     public static string StringFunction1(string a, string b)
213     {
214         return a + b;
215     }
216
217     public static string StringFunction2(int number)
218     {
219         string result = "0";
220         for (int i = 0; i < number; i++)
221         {
222             result += "," + (i + 1);
223         }
224         return result;
225     }
226
227     public static string StringFunction3(int number)
228     {
229         string result = "1";
230         int aux = 1;
231         for (int i = 0; i < number; i++)
232         {
233             aux *= 2;
234             result += "," + aux;
235         }
236         return result;
237     }
238
239     public static void Concatenate(string text1, string text2, 
240         string text3)
241     {
242         Console.WriteLine("Frase1 , 2 y 3");
243         Console.WriteLine(text1 + ", " + text2 + ", " + text3);
244
245         Console.WriteLine("Frase1 y 2");
246         Console.WriteLine("{0} , {1}", text1, text2);
247
248         Console.WriteLine("Frase2 y 3");
249         Console.WriteLine($"{text2} , {text3}");
250     }
251
252     // Funcion que devuelve la serie Collatz
253     // TODO: Funcion Collatz
254     public static List<int> Collatz(int number)
255     {
256         List<int> list = new List<int>();
257         int result = number;
258
259         list.Add(number);
260     }
```

```
261         while (result != 1)
262         {
263             if (IsEven(result))
264                 result /= 2;
265             else
266                 result = (result * 3) + 1;
267             list.Add(result);
268         }
269
270         return list;
271     }
272
273     // Serie de Fibonacci con un numero determinado de resultados
274     // TODO: Serie Fibonacci
275     public static string FibonacciFOR(int number)
276     {
277         string result = "0,1";
278         int n1 = 0;
279         int n2 = 1;
280         int sumNumbers;
281
282         for (int i = 0; i < number - 2; i++)
283         {
284             sumNumbers = n1 + n2;
285             result += "," + sumNumbers;
286             n1 = n2;
287             n2 = sumNumbers;
288         }
289         return result;
290     }
291
292     // Serie de Fibonacci mientras sea menor que un numero de resultados
293     public static string FibonacciWHILE(int number)
294     {
295         string result = "0,1";
296         int n1 = 0;
297         int n2 = 1;
298         int sumNumbers = n1 + n2;
299         while (sumNumbers <= number)
300         {
301             sumNumbers = n1 + n2;
302             result += "," + sumNumbers;
303             n1 = n2;
304             n2 = sumNumbers;
305         }
306         return result;
307     }
308
309     // Funcion que ejecuta un SWAP(intercambio) entre dos elementos
310     // TODO: Funcion Swap
311     public static void Swap(List<int> list)
312     {
```

```
313         int aux;
314         aux = list[0];
315         list[0] = list[1];
316         list[1] = aux;
317     }
318
319     // Funcion que ejecuta un SORT(ordenar) una lista
320     // TODO: Funcion Sort
321     public static void Sort(List<int> list)
322     {
323         if (list == null)
324             return;
325
326         int aux;
327         int n1 = list.Count - 1;
328         int n2 = list.Count;
329
330         for (int i = 0; i < n1; i++)
331         {
332             for (int j = i + 1; j < n2; j++)
333             {
334                 if (list[i] > list[j])
335                 {
336                     aux = list[i];
337                     list[i] = list[j];
338                     list[j] = aux;
339                 }
340             }
341         }
342     }
343
344     // Funcion que devuelve los elementos pares de una lista
345     public static List<int> GetListOdd(List<int> list)
346     {
347         List<int> result = new List<int>();
348
349         for (int i = 0; i < list.Count; i++)
350         {
351             if (IsEven(list[i]))
352                 result.Add(i);
353         }
354         return result;
355     }
356
357     // Funcion que devuelve los elementos pares de un array
358     // OJO NIVEL MEDIO EXAMEN
359     public static int[] GetListOdd(int[] array)
360     {
361         if (array == null)
362         {
363             Console.WriteLine("Error de validacion. Array
364                 introducida es null");
365             return new int[0];
366         }
367     }
```

```
365     }
366
367
368     int[] result;
369     int count = 0;
370
371     // Buscamos los numeros pares y guardamos el conteo total
372     for (int i = 0; i < array.Length; i++)
373     {
374         if (IsEven(array[i]))
375             count++;
376     }
377
378     // Creamos un array cuya longitud es igual al numero de pares encontrados
379     result = new int[count];
380     int contador = 0;
381
382
383     // Introducimos los elementos pares dentro de un nuevo array
384     for (int i = 0; i < array.Length; i++)
385     {
386         if (IsEven(array[i]))
387         {
388             result[contador] = array[i];
389             contador++;
390         }
391     }
392
393     return result;
394 }
395 }
396 }
397
```