```
1  namespace Delegado5
2  {
3      internal class Program
4      {
5          static void Main(string[] args)
6          {
7              Calculations calculations = new Calculations();
8
9              // El usuario llama a la funcion Filter que imprime por
                 pantalla
10             // los numeros de la lista mayores que 100.
11             calculations.FilterMajorThan(parameter => parameter > 100);
12
13             // El usuario llama a la funcion Multiplicate que imprime
                 por pantalla
14             // los numeros de la lista multiplicados por el parametro.
15             calculations.Multiplicate(parameter => { return parameter *
                 10; });
16
17             Paragraphs paragraphs = new Paragraphs();
18
19             // El usuario llama a la funcion Filter que imprime por
                 pantalla
20             // los parrafos mayores que 20 caracteres.
21             paragraphs.FilterLengthMajorThan(parameter => parameter >=
                 20);
22         }
23     }
24
25     public class Calculations
26     {
27         public delegate bool DelegadoFuncion(int parameter);
28         List<int> list = new List<int> { 1, 3, 10, 20, 30, 5, 50, 100,
             105 };
29
30         // Funcion que imprime el elemento de la lista
31         // si el delegado es verdadero.
32         public void FilterMajorThan(DelegadoFuncion del)
33         {
34             for (int i = 0; i < list.Count; i++)
35             {
36                 int element = list[i];
37                 if (del(element))
38                     Console.WriteLine(element);
39             }
40         }
41
42
43         public delegate int DelegadoFuncion2(int parameter);
44         List<int> list2 = new List<int> { 1, 2, 3, 4, 5, 6, 7, 8, 9,
             10 };
45
46         // Funcion que imprime el numero que devuelve el delegado.
```

```
47          public void Multiplicate(DelegadoFuncion2 del)
48          {
49              for (int i = 0; i < list2.Count; i++)
50              {
51                  Console.WriteLine(del(list2[i]));
52              }
53          }
54      }
55
56      public class Paragraphs
57      {
58          public delegate bool DelegateMajor(int number);
59          public List<string> list = new List<string>();
60
61          public Paragraphs()
62          {
63              CreateList();
64          }
65
66          public void CreateList()
67          {
68              list.Add("Me encanta cuando te arreglas el pelo con
                  rulos.");
69              list.Add("La genealogia de los barbaros vikingos se basaba
                  en la caza de animales salvajes y la busqueda constante de
                   la calidad " +
70                    "de un tesoro que no resultara demasiado caro, en
                        terminos de esfuerzos físicos y económicos.");
71              list.Add("A lo largo de la tarde se espera un leve repunte
                  al alza de los valores bursatiles dedicados a la banca
                  buscando" +
72                    "alcanzar un valor soporte a las caidas generalizadas
                        del dia de hoy.");
73          }
74
75          // Funcion que imprime el string si el delegado
76          // devuelve verdadero.
77          public void FilterLengthMajorThan(DelegateMajor del)
78          {
79              foreach (string s in list)
80              {
81                  int number = s.Count();
82                  if (del(number))
83                  {
84                      Console.WriteLine(s);
85                  }
86              }
87          }
88      }
89  }
```