```
 1  namespace ResumenFunciones
 2  {
 3      public class ExerciseMedinaBook
 4      {
 5
 6          // EJERCICIO 1.
 7          // Invertir un número de dos cifras.
 8          public static int ChangeOrder(int number)
 9          {
10              string numberCharacter = number.ToString();
11              string invert = "";
12              for (int i = numberCharacter.Length - 1; i >= 0; i--)
13              {
14                  invert += numberCharacter[i];
15              }
16              return Int32.Parse(invert);
17          }
18
19          // EJERCICIO 2.
20          // Invertir un número de tres cifras.
21          public static int ChangeOrderVersion(int number)
22          {
23              string numberCharacter = number.ToString();
24              string invert = "";
25              for (int i = numberCharacter.Length - 1; i >= 0; i--)
26              {
27                  invert += numberCharacter[i];
28              }
29              return Int32.Parse(invert);
30          }
31
32          // EJERCICIO 3:
33          // Operaciones básicas.
34          public static string CalculateBasics(double number1, double
              number2)
35          {
36              string result = "";
37
38              string sum = $"La suma de los numeros es: {number1 +
                  number2}";
39              string minus = $"La resta de los numeros es: {number1 -
                  number2}";
40
41              double product = number1 * number2;
42              double division = number1 / number2;
43              string productResult = "El producto de los numeros es: " +
                  product;
44              string divisionResult = "La division de los numeros es: " +
                   division;
45
46              double module = number1 % number2;
47              string moduleResult = String.Format("El modulo de los
                  numeros es {0}", module.ToString());
```

```
48              return result;
49          }
50
51          // EJERCICIO 4:
52          // Operaciones básicas ii
53          public static double CalculateBasicsV2(double orderQuantity,
              double orderPrice)
54          {
55              return orderQuantity * orderPrice;
56          }
57
58          // EJERCICIO 4bis:
59          // Operaciones básicas iii
60
61          public static double CalculateBasicsV2bis(int hamburgers, int
              fries, int drinks)
62          {
63              double priceHamburgers = hamburgers * 9.90;
64              double priceFries = fries * 4.50;
65              double priceDrinks = drinks * 2.20;
66
67              return priceHamburgers + priceFries + priceDrinks;
68          }
69
70          // EJERCICIO 5:
71          // Operaciones básicas iv.
72          // Uso de la clase 'Math'.
73          public static double CalculateBasicsiii(double number, int
              choice)
74          {
75              double result = 0.0;
76              switch (choice)
77              {
78                  case 0:
79                      result = Math.Abs(number);
80                      break;
81                  case 1:
82                      result = Math.Pow(number, 2);
83                      break;
84                  case 2:
85                      result = Math.Floor(number);
86                      break;
87                  case 3:
88                      result = Math.Round(number);
89                      break;
90                  default:
91                      result = 0.0;
92                      break;
93              }
94              return result;
95          }
96
97          // EJERCICIO 6: Formatos de salida.
```

```
 98
 99          // EJERCICIO 7:
100          // Cálculo de operación compleja.
101          public static double CalculateOperation(double number1, double
               number2)
102          {
103              return (number1 + number2) * (number1 - number2);
104          }
105
106          // EJERCICIO 8:
107          // Devolver el mayor de dos números.
108          public static double GetMajor(double number1, double number2)
109          {
110              //  Patron del tipo if(...)else,
111              //  es mejor resolverlo con operador ternario
112
113              double result = number1 > number2 ? number1 : number2;
114              return result;
115          }
116
117          // EJERCICIO 9: Devolver el mayor de tres números.
118          // TODO: [EXAMEN] Funcion dentro de funcion.
119          public static double GetMajorSerie(double number1, double
               number2, double number3)
120          {
121              return (GetMajor(GetMajor(number1, number2), number3));
122          }
123
124          // EJERCICIO 10: Programa de máquina para realizar cambio de
               moneda.
125          // TODO: [EXAMEN] Funcion que calcula restos de divisiones.
126          public static string ChangeMoney(int money)
127          {
128              int moneyChange = money;
129              string result = "";
130              if (moneyChange >= 100)
131              {
132                  int ticket100 = moneyChange / 100;
133                  result += $"{ticket100} billete/s de 100\n";
134                  moneyChange = moneyChange % 100;
135              }
136              if (moneyChange >= 50)
137              {
138                  int ticket50 = moneyChange / 50;
139                  result += $"{ticket50} billete/s de 50\n";
140                  moneyChange = moneyChange % 50;
141              }
142              if (moneyChange >= 20)
143              {
144                  int ticket20 = moneyChange / 20;
145                  result += $"{ticket20} billete/s de 20\n";
146                  moneyChange = moneyChange % 20;
147              }
```

```
148             if (moneyChange >= 10)
149             {
150                 int ticket10 = moneyChange / 10;
151                 result += $"{ticket10} billete/s de 10\n";
152                 moneyChange = moneyChange % 10;
153             }
154             if (moneyChange >= 5)
155             {
156                 int ticket5 = moneyChange / 5;
157                 result += $"{ticket5} billete/s de 5\n";
158             }
159             else
160             {
161                 result += $"La cantidad restante de {moneyChange} euros se devuelve en monedas";
162             }
163             return result;
164         }
165 
166         // EJERCICIO 11:
167         // Cálculo de ingreso por hijo discapacitado.
168         public static double CalculateDiscapacity(int numberSons, double quantity)
169         {
170             return numberSons * quantity;
171         }
172 
173         // EJERCICIO 12:
174         // Cálculo de número intermedio.
175         public static int GetMediumValue(int number1, int number2, int number3)
176         {
177             if (number1 < number2 && number2 < number3)
178                 return number2;
179             else if (number2 < number1 && number1 < number3)
180                 return number1;
181             else
182                 return number3;
183         }
184 
185         // EJERCICIO 13: Calculo de tarifa telefonica.
186 
187         // EJERCICIO 14: Tipo de triangulo según sus lados.
188         // TODO: [EXAMEN] Definicion de tipo de triangulo.
189         public static string GetTriangleType(double side1, double side2, double side3)
190         {
191             if (side1 == side2 && side1 == side3)
192                 return "El triangulo es equilatero";
193             else if (side1 != side2 && side2 == side3)
194                 return "El triangulo es isosceles";
195             else
196                 return "El triangulo es escaleno";
```

```
197         }
198
199         // EJERCICIO 15: Ejercicio Propuesto.
200         // TODO: [EXAMEN] Operadores ternarios.
201         public static string WelcomeBack(string name)
202         {
203             // Operador ternario
204             string result = name == "Dimitri" ? "Bienvenido de
                  vuelta" : "Lo siento, le he confundido";
205
206             // Instruccion condicional
207             if (name == "Dimitri")
208                 return "Bienvenido de vuelta";
209             return "Lo siento, le he confundido";
210         }
211
212         // EJERCICIO 16:
213         // Calculas dias de la semana.
214         public static DateTime CalculateDates(DateTime date)
215         {
216             return date;
217         }
218
219         // EJERCICIO 17:
220         // Calcular el estado civil de una personen.
221         public static string GetCivilState(string name)
222         {
223             Console.WriteLine($"Bienvenido al asesor de estados
                  civiles, {name}");
224             Console.WriteLine("Por favor, responda las siguientes
                  preguntas, asi podremos indicarle cual es su estado
                  civil");
225
226             Console.WriteLine("¿Cuanto suma 5 + 10?");
227
228             int count = 0;
229             string answer1 = Console.ReadLine();
230
231
232             if (answer1 == "15")
233                 count++;
234
235             switch (count)
236             {
237                 case 0:
238                     return "Su estado civil es casado";
239             }
240
241             return "Su estado civil es soltero";
242         }
243
244         // EJERCICIO 18:
245         // Calcular la calificacion de un examen.
```

```
246         public static string CalculateQualification(int number)
247         {
248             switch (number)
249             {
250                 case 0:
251                 case 1:
252                 case 2:
253                 case 3:
254                 case 4:
255                     return "Suspendido";
256                 case 5:
257                     return "Aprobado";
258                 case 6:
259                     return "Bien";
260                 case 7:
261                 case 8:
262                     return "Notable";
263                 case 9:
264                     return "Sobresaliente";
265                 case 10:
266                     return "Matricula";
267                 default:
268                     return "Introduzca un valor valido";
269             }
270         }
271 
272         // EJERCICIO 19:
273         // Programa que indica que tipo de caracter se le pasa por parametros.
274         public static string IsVowel(char letter)
275         {
276             if (70 < letter && letter < 140)
277             {
278                 switch (letter)
279                 {
280                     case 'a':
281                     case 'e':
282                     case 'i':
283                     case 'o':
284                     case 'u':
285                         return "Se trata de una vocal";
286                 }
287             }
288 
289             else if (70 < letter && letter < 140)
290                 return "se trata de una consonante";
291 
292             else
293                 return "se trata de un caracter numerico";
294             return "No es un caracter";
295         }
296 
297         public static string IsVowelV2(char letter)
```

```
298         {
299             if (letter >= 'A' && letter <= 'z')
300 
301                 switch (letter)
302                 {
303                     case 'a':
304                     case 'e':
305                     case 'i':
306                     case 'o':
307                     case 'u':
308                     case 'A':
309                     case 'E':
310                     case 'I':
311                     case 'O':
312                     case 'U':
313                         return "Se trata de una vocal";
314 
315                     default:
316                         return "Se trata de una consonante";
317                 }
318 
319             return "No es un caracter alfabetico";
320         }
321         // EJERCICIO 20:
322         // Calcular las tablas de multiplicar y las potencias de un
              número dado.
323         public static void GetProductTable(int number)
324         {
325             for (int i = 0; i < 10; i++)
326             {
327                 Console.WriteLine("{0} x {1} = {2}", number, i, number
                      * i);
328             }
329 
330             double result = number;
331             for (int i = 0; i < 10; i++)
332             {
333                 result = Math.Pow(number, i);
334                 Console.WriteLine("{0}^{1} = {2}", number, i, result);
335             }
336         }
337 
338         // EJERCICIO 21: Áreas de un hospital.
339 
340         // EJERCICIO 22:
341         // Suma de números pares e impares.
342         public static void CalculateSumEvenOdd(int number)
343         {
344             int evenNumber = 0;
345             int oddNumber = 1;
346 
347             for (int i = 2; i <= number; i += 2)
348                 evenNumber += i;
```

```
349
350             for (int i = 3; i <= number; i += 2)
351                 oddNumber += i;
352
353             Console.WriteLine("La suma de los numeros pares es: {0}\n" +
354                 "La suma de los numeros impares es: {1}", evenNumber, oddNumber);
355         }
356
357         // EJERCICIO 23:
358         // Calcular las tablas de multiplicar de una lista de enteros.
359         public static void GetMultipleProduct(List<int> list)
360         {
361             if (list == null || list.Count == 0)
362                 return;
363
364             for (int i = 0; i < list.Count; i++)
365             {
366                 GetProductTable(list[i]);
367             }
368         }
369
370         // EJERCICIO 24:
371         // Suma de los números que contiene una lista de enteros.
372         public static int GetNSum(List<int> list)
373         {
374             if (list == null || list.Count == 0)
375                 return 0;
376
377             int totalSum = 0;
378             for (int i = 0; i < list.Count; i++)
379             {
380                 totalSum += list[i];
381             }
382
383             return totalSum;
384         }
385
386         // EJERCICIO 25:
387         // Averiguar el número mayor y el número menor de una lista de enteros.
388         public static (double, double) GetMajorMinor(List<double> list)
389         {
390             if (list == null || list.Count == 0)
391                 return (double.NaN, double.NaN);
392
393             double aux = list[0];
394             for (int i = 0; i < list.Count - 1; i++)
395                 for (int j = 1 + i; j < list.Count; j++)
396                 {
397                     if (list[i] > list[j])
398                     {
```

```
399                         aux = list[i];
400                         list[i] = list[j];
401                         list[j] = aux;
402                     }
403                 }
404             return (list[0], list[list.Count - 1]);
405         }
406 
407         // EJERCICIO 26: Serie de Fibonacci.
408 
409         // EJERCICIO 27:
410         // Calcular las notas medias de una lista de notas de examen.
411         public static void CalculateAverageClass(List<double> list)
412         {
413             Random r;
414             r = new Random();
415             int randomNumber = r.Next(0, 11);
416 
417             if (list == null || list.Count == 0)
418                 return;
419 
420             double average = 0;
421             int count = 0;
422             for (int i = 0; i < list.Count; i++)
423             {
424                 CalculateQualification(randomNumber);
425                 average += randomNumber;
426                 count++;
427             }
428         }
429 
430         // EJERCICIO 28: Series de números y carácteres ASCII.
431         // TODO: [EXAMEN] Series de characters y strings.
432         public static void GetSerieAscii(int number)
433         {
434             Random r;
435             r = new Random();
436 
437             string characterString = "";
438             char characterLetter = ' ';
439             int totalAscii = 255;
440 
441             for (int i = 0; i <= number; i++)
442             {
443                 int randomNumber = r.Next(totalAscii);
444                 characterString = "" + randomNumber;
445                 characterLetter = (char)randomNumber;
446 
447                 Console.WriteLine(characterString);
448                 Console.WriteLine(characterLetter);
449             }
450         }
451 
```

```
452         // EJERCICIO 29: Funciones de cadenas de texto.
453
454         // EJERCICIO 30:
455         // Simulación de un reloj digital.
456         public static void DigitalClock()
457         {
458             Random r;
459             r = new Random();
460
461             int seconds, minutes, hours;
462
463             seconds = r.Next(60);
464             minutes = r.Next(60);
465             hours = r.Next(24);
466
467             Console.WriteLine("Su hora " +
468                 "es:\n        {0}    :   {1}   :   {2}",
469                 hours, minutes, seconds);
470         }
471
472         // EJERCICIO 31:
473         // Cantidad de vocales 'o' de un texto dado.
474         public static int GetVowels(string text)
475         {
476             int count = 0;
477             for (int i = 0; i < text.Length; i++)
478             {
479                 if (text[i] == 'o')
480                     count++;
481             }
482             return count;
483         }
484
485         // EJERCICIO 32:
486         // Programa que devuelve la estadística de vocales en un texto.
487         public static void GetVowelsPercent(string text)
488         {
489
490             double letterA = 0, letterE = 0, letterI = 0, letterO = 0,  ↵
                  letterU = 0;
491             double totalLetter = 0;
492
493             for (int i = 0; i < text.Length; i++)
494             {
495                 char letter = text[i];
496                 switch (letter)
497                 {
498                     case 'a':
499                         letterA++;
500                         break;
501                     case 'e':
502                         letterE++;
503                         break;
```

```
504                     case 'i':
505                         letterI++;
506                         break;
507                     case 'o':
508                         letterO++;
509                         break;
510                     case 'u':
511                         letterU++;
512                         break;
513                     default:
514                         totalLetter++;
515                         break;
516                 }
517             }
518 
519             double totalA = Math.Floor((letterA / totalLetter) * 100);
520             double totalE = Math.Floor((letterE / totalLetter) * 100);
521             double totalI = Math.Floor((letterI / totalLetter) * 100);
522             double totalO = Math.Floor((letterO / totalLetter) * 100);
523             double totalU = Math.Floor((letterU / totalLetter) * 100);
524 
525             Console.WriteLine("El porcentaje de vocales de su texto es ➥
                  el siguiente:\n" +
526                 "Vocales A: {0}\n" +
527                 "Vocales E: {1}\n" +
528                 "Vocales I: {2}\n" +
529                 "Vocales O: {3}\n" +
530                 "Vocales U: {4}\n", totalA, totalE, totalI, totalO,     ➥
                    totalU);
531         }
532 
533         // EJERCICIO 33: Factorial de un número.
534 
535         // EJERCICIO 34: Series.
536 
537         // EJERCICIO 35: Sucesión de N cuadros.
538 
539         // EJERCICIO 36: Varios.
540 
541         // EJERCICIO 50:
542         // Calcular el número de posiciones de un número dado.
543 
544         public static void GetPositions(int number)
545         {
546             int count = 0;
547             while (number > 0)
548             {
549                 number /= 10;
550                 count++;
551             }
552             Console.WriteLine("El numero tiene {0} posiciones", count);
553         }
554 
```

```
555         // EJERCICIO PAG 80:
556         // Responder a una suma.
557         public static void AnswerSum()
558         {
559             string userQuestion = "";
560             bool endingApp = false;
561             int questionNumber;
562             int resultNumber = 0;
563 
564             while (!endingApp)
565             {
566                 Console.WriteLine("Introduzca un numero:");
567                 userQuestion = Console.ReadLine();
568                 questionNumber = Int32.Parse(userQuestion);
569                 resultNumber += questionNumber;
570 
571                 if (questionNumber <= 0)
572                     endingApp = !endingApp;
573                 Console.WriteLine("La suma de los numeros es: {0}",
                      resultNumber);
574             }
575         }
576 
577         // EJERCICIO PAG 86:
578         // Añadir valores a un array.
579         public static int[] AddValuesArray()
580         {
581             bool endingApp = false;
582             int[] result = new int[4];
583             int userAnswer;
584             int contador = 0;
585             while (!endingApp)
586             {
587                 contador++;
588                 result = new int[contador];
589                 Console.WriteLine("Introduzca un numero:");
590                 userAnswer = Int32.Parse(Console.ReadLine());
591                 result[contador] = userAnswer;
592                 Console.WriteLine("Pulse [0] para salir");
593 
594                 userAnswer = Int32.Parse(Console.ReadLine());
595 
596                 if (userAnswer == 0)
597                     endingApp = true;
598             }
599             return result;
600         }
601     }
602 }
```