57:

```
* @file diccionario.cpp
 2:
      * @brief Programa para listar un diccionario tras barajar
 3:
 4:
 5:
      * @author Fulanito...
     * @date Diciembre-2020
 7:
      ^{\star} El programa es un ejemplo de uso de string y vectores-C con números aleatorios
 8:
 9:
10:
     * El programa no usará el tipo vector de char, aunque sí el vector-C con objetos de
     * tipo string. Deberá crear un vector con el siguiente contenido:
11:
              "caballero",
                             "Dulcinea", "historia",
                                                          "escudero",
12:
              "rocinante",
                            "adelante",
                                           "gobernador", "andantes",
13:
                            "capitulo",
                                           "menester", "doncella"
              "voluntad",
14:
15:
              "caballeria", "castillo",
                                                          "finalmente",
                                           "Fernando",
                            "hermosura", "palabras",
                                                         "gobierno",
              "aventura",
16:
                            "cardenio", "pensamientos", "Luscinda",
"aposento", "aventuras", "quisiera",
              "intencion",
17:
              "lagrimas",
18:
              "libertad",
                             "desgracia", "entendimiento", "pensamiento",
19:
                                           "semejantes", "silencio",
              "licencia",
                             "Mercedes",
20:
                                          "semejance",
"labrador",
              "valeroso",
                             "doncellas",
                                                         "caballerias",
21:
              "cristiano",
                            "cristianos", "discreto",
                                                          "hicieron",
22:
              "llegaron",
                             "quisiere",
                                           "espaldas",
23:
                                                         "muestras"
                                           "grandeza",
              "escuderos",
                            "discurso",
24:
                                                         "altisidora",
              "princesa",
                             "haciendo",
                                           "renegado",
25:
                                                          "provecho",
26:
              "quedaron",
                             "resolucion", "presente",
                                                          "encantadores",
              "enamorado",
                             "valiente",
                                           "encantado",
                                                         "molino",
27:
              "licenciado", "necesidad",
                                           "responder", "discrecion",
28:
              "ejercicio", "hacienda", "posadero", "presencia", "historias", "presentes",
29:
                                                         "rocinante",
                                                         "verdadero"
30:
31:
      ^{\star} A continuación, deberá barajar las entradas del vector. Para ello, puede hacer tantos
32:
     * intercambios como palabras tiene el diccionario. Un intercambio consiste en seleccionar
33:
     * dos posiciones e intercambiar los contenidos de las dos casillas del vector.
34:
35:
36:
     * Finalmente, tendrá que listar el nuevo diccionario, con las mismas palabra pero en
      * distinto orden. Además, para cada palabra, debe dibujar tantos guiones como letras tenga
37:
     * debajo. Por ejemplo, el listado puede comenzar así:
38:
39:
             1.- caballero
40:
41:
             2.- Dulcinea
42:
43:
              3.- valiente
44:
45:
              4.- valeroso
46:
47:
     * y terminar:
48:
            74.- licenciado
49:
50:
51:
              75.- escudero
52:
                   _____
53:
              76.- provecho
55:
56:
```

```
* @file diccionario.cpp
 2:
      * @brief Programa para listar un diccionario tras barajar
 3:
 4:
      * @author Don Oreo
 5:
      * @date Diciembre-2020
 7:
      * El programa es un ejemplo de uso de string y vectores-\mathcal C con numeros aleatorios
 8:
 9:
10:
      * El programa no usara el tipo vector de char, aunque si el vector-C con objetos de
     * tipo string. Debera crear un vector con el siguiente contenido:
11:
               "caballero",
                             "Dulcinea",
                                            "historia",
                                                           "escudero",
12:
              "rocinante",
                             "adelante",
                                            "gobernador", "andantes"
13:
                                                         "doncella"
              "voluntad",
                             "capitulo",
                                            "menester",
14:
                                                           "finalmente"
              "caballeria", "castillo",
                                            "Fernando",
15:
                             "hermosura", "palabras",
              "aventura",
                                                          "gobierno",
16:
                             "cardenio", "pensamientos", "Luscinda", "aposento", "aventuras", "quisiera",
              "intencion",
17:
              "lagrimas",
18:
19:
              "libertad",
                             "desgracia",
                                           "entendimiento", "pensamiento",
                                            "semejantes", "silencio",
20:
              "licencia",
                             "Mercedes",
                             "doncellas",
                                            "labrador",
                                                           "caballerias",
21:
              "valeroso",
              "cristiano",
                             "cristianos", "discreto",
                                                           "hicieron",
22:
              "llegaron",
                             "quisiere",
                                            "espaldas",
23:
                                                           "muestras"
              "escuderos",
                             "discurso",
                                            "grandeza",
24:
                                                          "altisidora"
25:
              "princesa",
                             "haciendo",
                                            "renegado",
                                                           "provecho",
                                                           "encantadores",
26:
              "quedaron",
                             "resolucion", "presente",
              "enamorado",
                             "valiente",
                                            "encantado",
                                                           "molino",
27:
              "licenciado", "necesidad",
                                            "responder",
28:
                                                           "discrecion"
29:
              "ejercicio", "hacienda",
                                            "posadero",
                                                           "rocinante",
                             "hacienda", "posadero", "historias", "presentes",
30:
               "presencia",
                                                          "verdadero"
31:
      ^{\star} A continuacion, debera barajar las entradas del vector. Para ello, puede hacer tantos
32:
     * intercambios como palabras tiene el diccionario. Un intercambio consiste en seleccionar
33:
     * dos posiciones e intercambiar los contenidos de las dos casillas del vector.
34:
      * Finalmente, tendra que listar el nuevo diccionario, con las mismas palabra pero en
36:
      * distinto orden. Ademas, para cada palabra, debe dibujar tantos guiones como letras tenga
37:
      * debajo. Por ejemplo, el listado puede comenzar asi:
38:
39:
             1.- caballero
40:
41:
             2.- Dulcinea
42:
43:
              3.- valiente
44:
45:
              4.- valeroso
46:
47:
48:
     * y terminar:
49:
            74.- licenciado
50:
51:
              75.- escudero
52:
                    _____
53:
              76.- provecho
54:
55:
56:
57: #include <iostream>
58: #include<time.h>
59: #include<stdlib.h>
60: using namespace std:
61:
62: int main() {
     srand(time(NULL));
63:
        int n1, numero2, palabras=75;
        char separador = '-';
65:
66:
       string intercambio;
                                                        "historia",
67:
       string diccionario[palabras] = {"Dulcinea",
                                                                       "escudero".
            "rocinante", "adelante", "gobernador", "andantes", "voluntad", "capitulo", "menester", "doncella",
68:
                                          "Fernando",
70:
            "caballeria", "castillo",
                                                        "finalmente",
            "aventura",
                           "hermosura",
                                                        "gobierno"
71:
                                          "palabras",
                          "cardenio",
                                         "pensamientos", "Luscinda",
            "intencion",
72:
                                          "aventuras", "quisiera",
73:
            "lagrimas",
                           "aposento",
74:
            "libertad",
                          "desgracia",
                                          "entendimiento", "pensamiento",
            "licencia",
                           "Mercedes",
75:
                                          "semejantes", "silencio",
                                          "labrador",
                           "doncellas",
                                                        "caballerias"
            "valeroso",
76:
            "cristiano", "cristianos", "discreto",
                                                        "hicieron",
77:
                                          "espaldas",
78:
            "llegaron",
                           "quisiere".
                                                        "muestras",
            "escuderos", "discurso",
                                          "grandeza",
                                                        "altisidora",
            "princesa",
                                          "renegado",
                                                         "provecho",
80:
                           "haciendo",
            "quedaron",
                           "resolucion", "presente",
                                                         "encantadores",
81:
            "enamorado",
                          "valiente",
                                          "encantado",
82:
                                                        "molino",
            "licenciado", "necesidad",
83:
                                          "responder",
                                                        "discrecion".
            "ejercicio", "hacienda", "posadero", "rocinante",
"presencia", "historias", "presentes", "verdadero");
84:
85:
86:
```

```
palabras = sizeof(diccionario)/sizeof(string);
        for (int i=0 ; i < palabras ; i++) {
    n1=rand()%palabras;</pre>
 88:
 89:
 90:
             numero2=rand()%palabras;
 91:
             intercambio=diccionario[n1];
             diccionario[n1]=diccionario[numero2];
 93:
              diccionario[numero2]=intercambio;
        }
 94:
 95:
         for (int i=0 ; i < palabras ; i++) {</pre>
 96:
 97:
             int tamanio = diccionario[i].size();
 98:
              cout << i+1 << " " << diccionario[i] << endl;</pre>
              cout << _:.
if (i+1 >= 10) {
    ...+ << " ";</pre>
 99:
100:
                  cout << "
101:
102:
             else{
103:
                  cout << " ";
104:
             for (int i=0 ; i < tamanio ; i++) {</pre>
105:
106:
                  cout << separador;</pre>
107:
108:
109: }
             cout << endl;
110: }
```

```
2: //
 3: // Fundamentos de Programación
 4: // ETS Informática y Telecomunicaciones
 5: // Universidad de Granada
 6: // // Don Oreo
 7: //
 9 .
10: #include <iostream>
11: #include <cmath>
12:
13: using namespace std;
14:
15: int main() {
16:
17:
      int dorsal;
      int hora, minutos;
18:
19:
     char DOS_PUNTOS;
20:
      const int TERMINADOR=0;
21:
      int ganador, hora_ganador, minutos_ganador;
22:
      int participantes=0;
23:
24:
      /*la variables ganador guarda el dorsal del ganador
        las variables hora_ganador y minutos_ganador guarda el tiempo(seria más facil con un vector...)
25:
26:
27:
      cout << "Introduce el dorsal del corredor(termine con el dorsal 0 o menor): ";</pre>
28:
29:
     cin >> dorsal;
30:
31:
      /*El programa terminará cuando el dorsal sea menor o igual al terminador */
32:
      while (dorsal>TERMINADOR) {
33:
         participantes++;
34:
         cout << "Introduce el tiempo del corredor: ";</pre>
35:
         cin >> hora >> DOS_PUNTOS >> minutos;
36:
37:
     /*si la hora es menor a la ganadora, automaticamente se convierte en el nuevo ganador
38:
        si la hora es igual, compara los minutos*/
39:
40:
         if(hora < hora_ganador |  (hora==hora_ganador && minutos_minutos_ganador)){</pre>
41:
            ganador = dorsal;
            hora_ganador = hora;
42:
43:
            minutos_ganador = minutos;
44:
45:
         cout << "Introduce otro dorsal(termine con el dorsal 0 o menor): ";</pre>
46:
         cin >> dorsal:
47:
     }
48:
49:
     if(participantes>0){
         cout << "-
50:
         cout << "\nEl ganador tiene el dorsal: " << ganador
51:
             << " con un tiempo de: " << hora_ganador << DOS_PUNTOS << minutos_ganador
52:
53:
              << "\nEl numero de participantes es: " << participantes << endl;</pre>
54:
55:
      else
         cout << "No hay ganador.";
56:
57: }
58:
59:
```

```
* @file maratonistas.cpp
 2:
      ^{\star} @brief Procesa tiempos de llegada y determina el ganador
 3:
 4:
     * @author Fulanito...
 5:
      * @date Noviembre-2020
 7:
      * Suponga una serie de datos correspondientes al resultado de una maratón. Los datos
 8:
     * consisten en varios valores por maratonista:
 9:
10:
           - Número de dorsal. Los dorsales de corredores son número positivos.
11:
           - Tiempo obtenido. El tiempo que ha necesitado para acabar la maratón. Está compuesto
12:
            de dos valores: horas y minutos.
13:
14:
     * Escriba un programa que lea los resultados de una carrera e indique quién es el ganador.
15:
     * El formato de entrada de valores será un valor de dorsal más el tiempo que ha necesitado
16:
     * para terminar. El tiempo se especifica con un formato horas:minutos (vea el ejemplo más
17:
      * abajo).
18:
19:
20:
     * Una ejemplo de ejecución es el siguiente:
21:
            Introduzca los dorsales y tiempos correspondientes. Termine con el dorsal 0:
             5 1:37
22:
            2 0:59
23:
24:
            9 1:04
25:
26:
            Número de participantes: 3
            Primera posición corresponde al dorsal 2 con un tiempo de 0:59
27:
28:
29:
     * donde puede ver que los datos terminan cuando se introduce un dorsal de número 0.
     * El resultado del programa corresponde a las dos últimas líneas, donde aparece el número
30:
      * de participantes seguido por el dorsal ganador y el tiempo correspondiente.
31:
32:
     * Debe tener en cuenta que también puede haber cero participantes; en este caso, el
33:
34:
     * programa responde como sigue:
35:
36:
             Introduzca los dorsales y tiempos correspondientes. Termine con el dorsal 0:
37:
38:
            Número de participantes: 0
39:
            No hay ganador
40:
41:
```

```
* @file pascua.cpp
 2:
      * @brief Calcula la fecha del domingo de Pascua de un año
 3:
 4:
     * @author Fulanito...
 5:
      * @date Octubre-2020
 7:
      * La fecha del domingo de Pascua corresponde al primer domingo después de la primera
 8:
     * luna llena que sigue al equinoccio de primavera. El algoritmo que se presenta a
9:
     * continuación (denominado Cómputus) permite calcular esta fecha y es válido para años
10:
     * comprendidos entre 1900 y 2100. Para un determinado año, los cálculos que hay que
11:
      * realizar son:
12:
          - A = año mod 19
13:
           -B = a\tilde{n}o \mod 4
14:
          -C = \tilde{a}no \mod 7
15:
          -D = (19 * A + 24) \mod 30
16:
17:
          -E = (2 * B + 4 * C + 6 * D + 5) \mod 7
          -N = (22 + D + E)
18:
     * donde mod significa módulo (resto de dividir el primero entre el segundo).
19:
20:
21:
      * El valor de N corresponde al día de marzo en el que se sitúa el domingo de Pascua.
      * En el caso de que sea mayor que 31, el valor se refiere a un día de abril. Además, hay
22:
     * dos excepciones:
23:
24:
         1.- Si la fecha obtenida es el 26 de abril, entonces la Pascua caerá en el 19
25:
              de abril.
          2.- Si es el 25 de abril, con D = 28, E = 6 y A > 10, entonces la Pascua es el 18
26:
27:
             de abril.
      ^{\star} Escriba un programa que lea un año y muestre el día y mes en el que se celebró o
28:
29:
     * celebrará el domingo de pascua para ese año.
      * En la siguiente lista tiene algunos datos para verificar que el ejercicio es correcto:
30:
31:
          - Año 2005 -> Pascua el 27 de marzo
          - Año 2011 -> Pascua el 24 de abril
32:
          - Año 2049 -> Pascua el 18 de abril
33:
34:
          - Año 2076 -> Pascua el 19 de abril
35:
36:
```

```
* @file pascua.cpp
 2:
      * @brief Calcula la fecha del domingo de Pascua de un año
 3:
 4:
 5:
     * @author Fulanito...
     * @date Octubre-2020
 7:
      * La fecha del domingo de Pascua corresponde al primer domingo después de la primera
 8:
     * luna llena que sigue al equinoccio de primavera. El algoritmo que se presenta a
 9:
10:
     * continuación (denominado Cómputus) permite calcular esta fecha y es válido para años
     * comprendidos entre 1900 y 2100. Para un determinado año, los cálculos que hay que
11:
      * realizar son:
12:
13:
          -A = a\tilde{n}o \mod 19
14:
           -B = a\tilde{n}o \mod 4
15:
          -C = \tilde{ano} \mod 7
          -D = (19 * A + 24) \mod 30
16:
17:
           -E = (2 * B + 4 * C + 6 * D + 5) \mod 7
          -N = (22 + D + E)
18:
     * donde mod significa módulo (resto de dividir el primero entre el segundo).
19:
20:
21:
      * El valor de N corresponde al día de marzo en el que se sitúa el domingo de Pascua.
     * En el caso de que sea mayor que 31, el valor se refiere a un día de abril. Además, hay
22:
     * dos excepciones:
23:
24:
         1.- Si la fecha obtenida es el 26 de abril, entonces la Pascua caerá en el 19
25:
               de abril.
26:
           2.- Si es el 25 de abril, con D = 28, E = 6 y A > 10, entonces la Pascua es el 18
27:
              de abril.
     * Escriba un programa que lea un año y muestre el día y mes en el que se celebró o
28:
29:
     * celebrará el domingo de pascua para ese año.
     * En la siguiente lista tiene algunos datos para verificar que el ejercicio es correcto:
30:
        - Año 2005 -> Pascua el 27 de marzo
31:
          - Año 2011 -> Pascua el 24 de abril
32:
          - Año 2049 -> Pascua el 18 de abril
33:
34:
          - Año 2076 -> Pascua el 19 de abril
35:
36:
      */
37: #include <iostream>
38: using namespace std;
39:
40: int main()
41: {
42:
        int anio, a, b, c, d, e, dia;
43:
        cout <<"Este programa calcula el dia de domingo de Pascua" << endl;</pre>
44:
45:
            cout <<"Introduzca el anio: ";</pre>
46:
           cin >> anio;
47:
48:
        //Cálculo
49:
50:
        a = anio % 19;
       b = anio % 4;
51:
52:
        c = anio % 7;
53:
        d = (19 * a + 24) % 30;
        e = (2 * b + 4 * c + 6 * d + 5) % 7;
54:
55:
       dia = (22 + d + e);
56:
57:
       //Salida de Datos...
58:
       cout << "\nLa Fecha del Domingo de Pascua de " << anio <<" es el ";
59:
60:
       if (dia<=31)
61:
            cout << dia << " de Marzo ";
62:
        else if(dia==31 + 25)
63:
           cout << "18 de Abril ";
64:
        else if (dia==31 + 26 && d == 28 && e == 6 && a > 10)
65:
           cout << "19 de Abril ";
66:
67:
68:
           cout << dia - 31 << " de Abril ";
        return 0;
70: }
```

```
* @file media_movil.cpp
 2:
      ^{\star} @brief Calcula la media movil de una secuencia de temperaturas
 3:
 4:
     * @author Fulanito...
 5:
      * @date Diciembre-2020
 7:
      ^{\star} Escriba un programa que procesa una secuencia de valores de temperatura hasta que
 8:
      * se introduce una temperatura menor que el cero absoluto (-273.15 grados).
 9:
10:
      * Como resultado, escribirá una secuencia de datos que corresponde a la media móvil
11:
      * con tamaño N. Cada valor de la secuencia de esta media móvil corresponde a:
12:
13:
14:
           - La media de los primeros N datos desde el 0 al N-1,
15:
          - la media de los N siguientes desde el 1 al N,
          - la media de los N siguientes desde el 2 al N+1,
16:
17:
           - etc.
18:
     * Por tanto, si hay D datos, la secuencia resultado tendrá D-(N-1) valores. Esta
19:
20:
     * secuencia, además, estará también terminada con un valor centinela.
21:
      ^{\star} El problema se puede resolver cargando toda la secuencia de datos y luego
22:
     * calculando la media móvil para cada N datos, aunque también se podría limitar el
23:
     * tamaño de la memoria ocupada evitando tener toda la secuencia, pues sólo es
24:
      * necesario almacenar los últimos N valores.
25:
26:
     * Nota: En el problema, puede suponer que N es fijo y tiene un valor
* predeterminado. Así, evita tener que introducirlo; tanto la entrada como la
27:
28:
29:
     * salida serán una simple secuencia.
30:
31:
      * Un ejemplo de ejecución, con N valiendo 5, es:
           1 2 3 4 5 6 7 8 9 -300
32:
            3 4 5 6 7 -300
33:
34:
     * donde la primera línea es la entrada y la segunda la salida.
35:
36:
      * Otra ejemplo, ahora con N valiendo 3, es:
            -0.04 -0.05 -0.09 -0.06 -0.07 -0.01 0.09 0.07 0.02 0.12 0.15 -300
37:
            38:
39:
40:
```

```
2: //
 3: // Fundamentos de Programación
 4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
 7: //
 9:
10: //media movil
11:
     /* Un ejemplo de ejecución, con N valiendo 5, es:
12:
         1 2 3 4 5 6 7 8 9 -300
3 4 5 6 7 -300
13:
13: ^ 125450,05
14: * 3 4 5 6 7 -300
15: * donde la primera línea es la entrada y la segunda la salida.
16:
17:
     * Otra ejemplo, ahora con N valiendo 3, es:
           -0.04 -0.05 -0.09 -0.06 -0.07 -0.01 0.09 0.07 0.02 0.12 0.15 -300
18:
19:
           -0.06 -0.0666667 -0.0733333 -0.0466667 0.00333333 0.05 0.06 0.07 0.0966667 -300
20:
21:
22:
23: #include <iostream>
24: using namespace std;
26: int main() {
     const int MAX_DATOS = 1000;
27:
      const double TERMINADOR = -273.15;
28:
29:
     double v[MAX_DATOS];
     double datos = 0;
int N;
30:
31:
     int util = 0;
double suma = 0;
32:
33:
34:
     double media[MAX_DATOS];
35:
     cin >> datos;
36:
     while (datos > TERMINADOR) {
37:
38:
        cin >> datos;
39:
         v[util] = datos;
40:
         util++;
     }
41:
42:
      cout << "Introduce el N: ";
43:
44:
     cin >> N;
45:
46:
      //Computo
47:
48:
      int iter;
49:
      for(iter = 0; N + iter < util; iter++){</pre>
50:
       for(int i = iter; i < N + iter; i++)</pre>
51:
            suma += v[i];
52:
53:
54:
        media[iter] = suma / N;
55:
         suma = 0;
56:
57:
58:
      media[iter] = datos;
59:
60:
      //Salida de Datos
61:
     for(int i=0; i <= iter; i++)
  cout << media[i] << " ";</pre>
62:
63:
64: }
65:
66:
67:
68:
```