```
/**
 * @file diccionario.cpp
 * @brief Programa para listar un diccionario tras barajar
 *
 * @author Fulanito...
 * @date Diciembre-2020
 *
 * El programa es un ejemplo de uso de string y vectores-C con números aleatorios
 *
 * 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:
 *       "caballero",  "Dulcinea",    "historia",    "escudero",
 *       "rocinante",  "adelante",    "gobernador", "andantes",
 *       "voluntad",   "capitulo",    "menester",    "doncella",
 *       "caballeria", "castillo",    "Fernando",    "finalmente",
 *       "aventura",   "hermosura",   "palabras",    "gobierno",
 *       "intencion",  "cardenio",    "pensamientos", "Luscinda",
 *       "lagrimas",   "aposento",    "aventuras",   "quisiera",
 *       "libertad",   "desgracia",   "entendimiento", "pensamiento",
 *       "licencia",   "Mercedes",    "semejantes", "silencio",
 *       "valeroso",   "doncellas",   "labrador",    "caballerias",
 *       "cristiano",  "cristianos",  "discreto",    "hicieron",
 *       "llegaron",   "quisiere",    "espaldas",    "muestras",
 *       "escuderos",  "discurso",    "grandeza",    "altisidora",
 *       "princesa",   "haciendo",    "renegado",    "provecho",
 *       "quedaron",   "resolucion",  "presente",    "encantadores",
 *       "enamorado",  "valiente",    "encantado",   "molino",
 *       "licenciado", "necesidad",   "responder",   "discrecion",
 *       "ejercicio",  "hacienda",    "posadero",    "rocinante",
 *       "presencia",  "historias",   "presentes",   "verdadero"
 *
 * A continuación, deberá barajar las entradas del vector. Para ello, puede hacer tantos
 * intercambios como palabras tiene el diccionario. Un intercambio consiste en seleccionar
 * dos posiciones e intercambiar los contenidos de las dos casillas del vector.
 *
 * 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
 * debajo. Por ejemplo, el listado puede comenzar así:
 *       1.- caballero
 *           ---------
 *       2.- Dulcinea
 *           --------
 *       3.- valiente
 *           --------
 *       4.- valeroso
 *           --------
 *
 * y terminar:
 *       74.- licenciado
 *            ----------
 *       75.- escudero
 *            --------
 *       76.- provecho
 *            --------
 *
 */

```

```cpp
/**
 * @file diccionario.cpp
 * @brief Programa para listar un diccionario tras barajar
 *
 * @author Don Oreo
 * @date Diciembre-2020
 *
 * El programa es un ejemplo de uso de string y vectores-C con numeros aleatorios
 *
 * 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:
 *       "caballero",  "Dulcinea",    "historia",    "escudero",
 *       "rocinante",  "adelante",    "gobernador", "andantes",
 *       "voluntad",   "capitulo",    "menester",    "doncella",
 *       "caballeria", "castillo",    "Fernando",    "finalmente",
 *       "aventura",   "hermosura",   "palabras",    "gobierno",
 *       "intencion",  "cardenio",    "pensamientos", "Luscinda",
 *       "lagrimas",   "aposento",    "aventuras",   "quisiera",
 *       "libertad",   "desgracia",   "entendimiento", "pensamiento",
 *       "licencia",   "Mercedes",    "semejantes", "silencio",
 *       "valeroso",   "doncellas",   "labrador",    "caballerias",
 *       "cristiano",  "cristianos",  "discreto",    "hicieron",
 *       "llegaron",   "quisiere",    "espaldas",    "muestras",
 *       "escuderos",  "discurso",    "grandeza",    "altisidora",
 *       "princesa",   "haciendo",    "renegado",    "provecho",
 *       "quedaron",   "resolucion",  "presente",    "encantadores",
 *       "enamorado",  "valiente",    "encantado",   "molino",
 *       "licenciado", "necesidad",   "responder",   "discrecion",
 *       "ejercicio",  "hacienda",    "posadero",    "rocinante",
 *       "presencia",  "historias",   "presentes",   "verdadero"
 *
 * A continuacion, debera barajar las entradas del vector. Para ello, puede hacer tantos
 * intercambios como palabras tiene el diccionario. Un intercambio consiste en seleccionar
 * dos posiciones e intercambiar los contenidos de las dos casillas del vector.
 *
 * Finalmente, tendra que listar el nuevo diccionario, con las mismas palabra pero en
 * distinto orden. Ademas, para cada palabra, debe dibujar tantos guiones como letras tenga
 * debajo. Por ejemplo, el listado puede comenzar asi:
 *       1.- caballero
 *           ---------
 *       2.- Dulcinea
 *           --------
 *       3.- valiente
 *           --------
 *       4.- valeroso
 *           --------
 *
 * y terminar:
 *       74.- licenciado
 *            ----------
 *       75.- escudero
 *            --------
 *       76.- provecho
 *            --------
 *
 */
#include <iostream>
#include<time.h>
#include<stdlib.h>
using namespace std;

int main(){
    srand(time(NULL));
    int n1,numero2,palabras=75;
    char separador = '-';
    string intercambio;
    string diccionario[palabras] = {"Dulcinea",   "historia",   "escudero",
        "rocinante",  "adelante",    "gobernador", "andantes",
        "voluntad",   "capitulo",    "menester",    "doncella",
        "caballeria", "castillo",    "Fernando",    "finalmente",
        "aventura",   "hermosura",   "palabras",    "gobierno",
        "intencion",  "cardenio",    "pensamientos", "Luscinda",
        "lagrimas",   "aposento",    "aventuras",   "quisiera",
        "libertad",   "desgracia",   "entendimiento", "pensamiento",
        "licencia",   "Mercedes",    "semejantes", "silencio",
        "valeroso",   "doncellas",   "labrador",    "caballerias",
        "cristiano",  "cristianos",  "discreto",    "hicieron",
        "llegaron",   "quisiere",    "espaldas",    "muestras",
        "escuderos",  "discurso",    "grandeza",    "altisidora",
        "princesa",   "haciendo",    "renegado",    "provecho",
        "quedaron",   "resolucion",  "presente",    "encantadores",
        "enamorado",  "valiente",    "encantado",   "molino",
        "licenciado", "necesidad",   "responder",   "discrecion",
        "ejercicio",  "hacienda",    "posadero",    "rocinante",
        "presencia",  "historias",   "presentes",   "verdadero"};

```

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

```cpp
////////////////////////////////////////////////////////////////////////
//
// Fundamentos de Programación
// ETS Informática y Telecomunicaciones
// Universidad de Granada
// // Don Oreo
//
////////////////////////////////////////////////////////////////////////

#include <iostream>
#include <cmath>

using namespace std;

int main(){

   int dorsal;
   int hora,minutos;
   char DOS_PUNTOS;
   const int TERMINADOR=0;
   int ganador,hora_ganador,minutos_ganador;
   int participantes=0;

   /*la variables ganador guarda el dorsal del ganador
     las variables hora_ganador y minutos_ganador guarda el tiempo(seria más facil con un vector...)
   */

   cout << "Introduce el dorsal del corredor(termine con el dorsal 0 o menor): ";
   cin >> dorsal;

   /*El programa terminará cuando el dorsal sea menor o igual al terminador */
   while(dorsal>TERMINADOR){
      participantes++;
      cout << "Introduce el tiempo del corredor: ";
      cin >> hora >> DOS_PUNTOS >> minutos;

   /*si la hora es menor a la ganadora, automaticamente se convierte en el nuevo ganador
     si la hora es igual, compara los minutos*/

      if(hora < hora_ganador || (hora==hora_ganador && minutos<minutos_ganador)){
         ganador = dorsal;
         hora_ganador = hora;
         minutos_ganador = minutos;
      }
      cout << "Introduce otro dorsal(termine con el dorsal 0 o menor): ";
      cin >> dorsal;
   }

   if(participantes>0){
      cout << "----------------------------------------";
      cout << "\nEl ganador tiene el dorsal: " << ganador
           << " con un tiempo de: " << hora_ganador << DOS_PUNTOS << minutos_ganador
           << "\nEl numero de participantes es: " << participantes << endl;
   }
   else
      cout << "No hay ganador.";
}
```

```
/**
 * @file maratonistas.cpp
 * @brief Procesa tiempos de llegada y determina el ganador
 *
 * @author Fulanito...
 * @date Noviembre-2020
 *
 * Suponga una serie de datos correspondientes al resultado de una maratón. Los datos
 * consisten en varios valores por maratonista:
 *
 *    - Número de dorsal. Los dorsales de corredores son número positivos.
 *    - Tiempo obtenido. El tiempo que ha necesitado para acabar la maratón. Está compuesto
 *      de dos valores: horas y minutos.
 *
 * Escriba un programa que lea los resultados de una carrera e indique quién es el ganador.
 * El formato de entrada de valores será un valor de dorsal más el tiempo que ha necesitado
 * para terminar. El tiempo se especifica con un formato horas:minutos (vea el ejemplo más
 * abajo).
 *
 * Una ejemplo de ejecución es el siguiente:
 *      Introduzca los dorsales y tiempos correspondientes. Termine con el dorsal 0:
 *      5 1:37
 *      2 0:59
 *      9 1:04
 *      0
 *      Número de participantes: 3
 *      Primera posición corresponde al dorsal 2 con un tiempo de 0:59
 *
 * 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
 * de participantes seguido por el dorsal ganador y el tiempo correspondiente.
 *
 * Debe tener en cuenta que también puede haber cero participantes; en este caso, el
 * programa responde como sigue:
 *
 *      Introduzca los dorsales y tiempos correspondientes. Termine con el dorsal 0:
 *      0
 *      Número de participantes: 0
 *      No hay ganador
 *
 */
```

```
/**
 * @file pascua.cpp
 * @brief Calcula la fecha del domingo de Pascua de un año
 *
 * @author Fulanito...
 * @date Octubre-2020
 *
 *  La fecha del domingo de Pascua corresponde al primer domingo después de la primera
 * luna llena que sigue al equinoccio de primavera. El algoritmo que se presenta a
 * 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
 * realizar son:
 *    - A = año mod 19
 *    - B = año mod 4
 *    - C = año mod 7
 *    - D = (19 * A + 24) mod 30
 *    - E = (2 * B + 4 * C + 6 * D + 5) mod 7
 *    - N = (22 + D + E)
 * donde mod significa módulo (resto de dividir el primero entre el segundo).
 *
 * 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
 * dos excepciones:
 *    1.- Si la fecha obtenida es el 26 de abril, entonces la Pascua caerá en el 19
 *        de abril.
 *    2.- Si es el 25 de abril, con D = 28, E = 6 y A > 10, entonces la Pascua es el 18
 *       de abril.
 * Escriba un programa que lea un año y muestre el día y mes en el que se celebró o
 * celebrará el domingo de pascua para ese año.
 * En la siguiente lista tiene algunos datos para verificar que el ejercicio es correcto:
 *    - Año 2005 -> Pascua el 27 de marzo
 *    - Año 2011 -> Pascua el 24 de abril
 *    - Año 2049 -> Pascua el 18 de abril
 *    - Año 2076 -> Pascua el 19 de abril
 *
 */
```

```cpp
/**
  * @file pascua.cpp
  * @brief Calcula la fecha del domingo de Pascua de un año
  *
  * @author Fulanito...
  * @date Octubre-2020
  *
  *  La fecha del domingo de Pascua corresponde al primer domingo después de la primera
  * luna llena que sigue al equinoccio de primavera. El algoritmo que se presenta a
  * 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
  * realizar son:
  *    - A = año mod 19
  *    - B = año mod 4
  *    - C = año mod 7
  *    - D = (19 * A + 24) mod 30
  *    - E = (2 * B + 4 * C + 6 * D + 5) mod 7
  *    - N = (22 + D + E)
  * donde mod significa módulo (resto de dividir el primero entre el segundo).
  *
  * 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
  * dos excepciones:
  *    1.- Si la fecha obtenida es el 26 de abril, entonces la Pascua caerá en el 19
  *        de abril.
  *    2.- Si es el 25 de abril, con D = 28, E = 6 y A > 10, entonces la Pascua es el 18
  *       de abril.
  * Escriba un programa que lea un año y muestre el día y mes en el que se celebró o
  * celebrará el domingo de pascua para ese año.
  * En la siguiente lista tiene algunos datos para verificar que el ejercicio es correcto:
  *    - Año 2005 -> Pascua el 27 de marzo
  *    - Año 2011 -> Pascua el 24 de abril
  *    - Año 2049 -> Pascua el 18 de abril
  *    - Año 2076 -> Pascua el 19 de abril
  *
  */
#include <iostream>
using namespace std;

int main()
{
    int anio,a,b,c,d,e,dia;
    cout <<"Este programa calcula el dia de domingo de Pascua" << endl;

        cout <<"Introduzca el anio: ";
        cin >> anio;

    //Cálculo

    a = anio % 19;
    b = anio % 4;
    c = anio % 7;
    d = (19 * a + 24) % 30;
    e = (2 * b + 4 * c + 6 * d + 5) % 7;
    dia = (22 + d + e);

    //Salida de Datos...

    cout << "\nLa Fecha del Domingo de Pascua de " << anio <<" es el ";

    if (dia<=31)
        cout << dia << " de Marzo ";
    else if(dia==31 + 25)
        cout << "18 de Abril ";
    else if(dia==31 + 26 && d == 28 && e == 6 && a > 10)
        cout << "19 de Abril ";
    else
        cout << dia - 31 << " de Abril ";
    return 0;
}
```

```
/**
 * @file media_movil.cpp
 * @brief Calcula la media movil de una secuencia de temperaturas
 *
 * @author Fulanito...
 * @date Diciembre-2020
 *
 * Escriba un programa que procesa una secuencia de valores de temperatura hasta que
 * se introduce una temperatura menor que el cero absoluto  (-273.15 grados).
 *
 * Como resultado, escribirá una secuencia de datos que corresponde a la media móvil
 * con tamaño N. Cada valor de la secuencia de esta media móvil corresponde a:
 *
 *    - La media de los primeros N datos desde el 0 al N-1,
 *    - la media de los N siguientes desde el 1 al N,
 *    - la media de los N siguientes desde el 2 al N+1,
 *    - etc.
 *
 * Por tanto, si hay D datos, la secuencia resultado tendrá D-(N-1) valores. Esta
 * secuencia, además, estará también terminada con un valor centinela.
 *
 * El problema se puede resolver cargando toda la secuencia de datos y luego
 * calculando la media móvil para cada N datos, aunque también se podría limitar el
 * tamaño de la memoria ocupada evitando tener toda la secuencia, pues sólo es
 * necesario almacenar los últimos N valores.
 *
 * 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
 * salida serán una simple secuencia.
 *
 * Un ejemplo de ejecución, con N valiendo 5, es:
 *     1 2 3 4 5 6 7 8 9 -300
 *     3 4 5 6 7 -300
 * donde la primera línea es la entrada y la segunda la salida.
 *
 * 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
 *     -0.06 -0.0666667 -0.0733333 -0.0466667 0.00333333 0.05 0.06 0.07 0.0966667 -300
 *
 */
```

```cpp
////////////////////////////////////////////////////////////////////////
//
// Fundamentos de Programación
// ETS Informática y Telecomunicaciones
// Universidad de Granada
// // Don Oreo
//
////////////////////////////////////////////////////////////////////////

//media movil

  /* Un ejemplo de ejecución, con N valiendo 5, es:
  *     1 2 3 4 5 6 7 8 9 -300
  *     3 4 5 6 7 -300
  * donde la primera línea es la entrada y la segunda la salida.
  *
  * 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
  *     -0.06 -0.0666667 -0.0733333 -0.0466667 0.00333333 0.05 0.06 0.07 0.0966667 -300
  *
  */

#include <iostream>
using namespace std;

int main(){
   const int MAX_DATOS = 1000;
   const double TERMINADOR = -273.15;
   double v[MAX_DATOS];
   double datos = 0;
   int N;
   int util = 0;
   double suma = 0;
   double media[MAX_DATOS];

   cin >> datos;
   while(datos > TERMINADOR){
      cin >> datos;
      v[util] = datos;
      util++;
   }

   cout << "Introduce el N: ";
   cin >> N;

   //Computo

   int iter;

   for(iter = 0; N + iter < util; iter++){
      for(int i = iter; i < N + iter; i++)
         suma += v[i];

      media[iter] = suma / N;
      suma = 0;
   }

   media[iter] = datos;

   //Salida de Datos

   for(int i=0; i <= iter ; i++)
      cout << media[i] << " ";
}
```