```
    1: ////////////////////////////////////////////////////////////////////////
    2: //
    3: // Fundamentos de Programación
    4: // ETS Informática y Telecomunicaciones
    5: // Universidad de Granada
    6: // // Don Oreo
    7: //
    8: ////////////////////////////////////////////////////////////////////////
    9:
   10: /*
   11:
   12: . [Contiene débil] (Examen Enero 2018) Dados dos vectores grande y peque de
   13: tipo char, queremos comprobar si el primero contiene al segundo de la siguiente forma: todos los caracte
res de peque tienen que aparecer en grande en el
   14: mismo orden, aunque no tienen por qué estar consecutivos. Por ejemplo, el vector grande = {'d','e','s','
t','i','n','o'} contiene débilmente al vector
   15: peque = {'s','i'} pero no a peque = {'i','s'}.
   16: Construya un programa que lea desde teclado los caracteres del vector grande, parando la entrada cuando
se introduzca el carácter #. Haga lo mismo para introducir los
   17: caracteres del vector peque. El programa indicará si el vector grande contiene o no al vector peque.
   18: Si lo desa puede usar el esbozo del programa que se encuentra en el siguiente enlace:
   19: http://decsai.ugr.es/jccubero/FP/III_ContieneDebilEsbozo.cpp
   20: Ejemplo de entrada: destino#si#
   21:  Salida correcta: Sí lo contiene
   22: Ejemplo de entrada: destino#is#
   23:  Salida correcta: No lo contiene
   24: Ejemplo de entrada: destino#no#
   25:  Salida correcta: Sí lo contiene
   26: Finalidad: Recorrido de las componentes de un vector. Dificultad Media.
   27:
   28: */
   29:
   30: #include <iostream>
   31: using namespace std;
   32:
   33: int main(){
   34:    const char TERMINADOR = '#';
   35:    const int MAX_NUM_CARACT = 200;
   36:    char grande[MAX_NUM_CARACT],
   37:         peque[MAX_NUM_CARACT];
   38:    char car;
   39:    int util_grande, util_peque;
   40:    int num_leidos;
   41:    bool encontrado;
   42:
   43:
   44:    cout << "Búsqueda -débil- de un vector de caracteres dentro de otro\n"
   45:         << "Introduzca los carácteres del vector grande, con terminador "
   46:         << TERMINADOR << "\n"
   47:         << "A continuación introduzca  los caracteres del vector pequeño,"
   48:         << " usando el mismo terminador.\n\n";
   49:
   50:    // Lectura
   51:
   52:    car = cin.get();
   53:    num_leidos = 0;
   54:
   55:    while (car != TERMINADOR && num_leidos < MAX_NUM_CARACT){
   56:       grande[num_leidos] = car;
   57:       car = cin.get();
   58:       num_leidos++;
   59:    }
   60:
   61:    util_grande = num_leidos;
   62:
   63:    car = cin.get();
   64:    num_leidos = 0;
   65:
   66:    while (car != TERMINADOR && num_leidos < MAX_NUM_CARACT){
   67:       peque[num_leidos] = car;
   68:       car = cin.get();
   69:       num_leidos++;
   70:    }
   71:
   72:    util_peque = num_leidos;
   73:
   74:
   75:
   76:    ////////////////////////////////////////////////////////
   77:    encontrado=false;
   78:    num_leidos=0;
   79:    int j=0;
   80:
   81:       for(int i=0;i<util_grande && encontrado==false;i++){
   82:          if(grande[i]==peque[j]){
   83:             num_leidos++;
```

```
84:             j++;
85:
86:          if(num_leidos==util_peque)
87:             encontrado=true;
88:          }
89:       }
90:    ////////////////////////////////////////////////////////
91:
92:
93:
94:    cout << "\n";
95:
96:    if (encontrado)
97:       cout << "\nEl vector pequeño está dentro del grande";
98:    else
99:       cout << "\nEl vector pequeño NO está dentro del grande";
100:
101: /*
102: aaabbbccc#abc#
103: Si
104:
105: abc#a#
106: Si
107:
108: cba#a#
109: Si
110:
111: azbzcz#abc#
112: Si
113:
114: abz#abc#
115: No
116: */
117: }
118:
119:
120:
```

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

// Elimina ocurrencias de una competencia (versión eficiente)
// La entrada de cin de este programa es un .txt con todo el Quijote

#include <iostream>
using namespace std;

int main (){
        const char TERMINADOR = '#';
        const long long MAX_NUM_CARACTERES = 25e+5;
        char v[MAX_NUM_CARACTERES];
        char a_borrar;
        char caracter;
        int pos_escritura, pos_lectura, utilizados_v, utilizados_final;

   caracter = cin.get();
   utilizados_v = 0;

   while (caracter != TERMINADOR){
      v[utilizados_v] = caracter;
      caracter = cin.get();
      utilizados_v++;
   }

   a_borrar = cin.get();

        utilizados_final = 0;
        pos_escritura = 0;
        pos_lectura = 0;

        for (int i = pos_escritura ; i < utilizados_v ; i++){
                if (v[pos_lectura] == a_borrar){
                        while (v[pos_lectura] == a_borrar){
                                pos_lectura++;
                        }

                        v[pos_escritura] = v[pos_lectura];
                        utilizados_final++;
                }
                else{
                        v[pos_escritura] = v[pos_lectura];
                }

                pos_escritura++;
                pos_lectura++;
        }

        for (int i = 0 ; i <= utilizados_final ; i++){
                cout << v[i];
        }

}
```

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

// top_k: Calcula los k mayores valores de un vector - versión ineficiente -

/*
Se dispone de una serie de enteros enteros positivos y se
quiere calcular los k mayores, ordenados de mayor a menor. Construya un programa
que vaya leyendo enteros desde teclado hasta que se introduzca -1. A continuación lea
el número k y aplique el siguiente algoritmo:
Vector original: v
Vector que contendrá los k mayores valores: topk
   Copiar v en topk
   Ordenar topk de MAYOR a MENOR <-- Atención!!!
      (se recomienda modificar el algoritmo de ordenación
      por inserción)
   Seleccionar los k primeros elementos de topk
Finalmente, imprima los k primeros valores del vector topk en pantalla.
Ejemplo de entrada: 2 0 3 2 12 -1 2
-- Salida correcta: 12 3
*/

#include <iostream>
using namespace std;

int main(){
   const int TERMINADOR = -1;
   int entero;
   const int TAMANIO                =  1e6;
   int vector[TAMANIO], topk[TAMANIO];
   int utilizados_vector, k;

   ////////////////////////////////////////////////////////////////
   // Lectura de los datos:


   cout << "Topk.\n\n"
        << "Introduzca enteros con terminador "
        << TERMINADOR << "\n"
        << "Luego introduzca el valor de k.\n\n";

   utilizados_vector = 0;
   cin >> entero;

   while (entero != TERMINADOR && utilizados_vector < TAMANIO){
      vector[utilizados_vector] = entero;
      utilizados_vector++;
      cin >> entero;
   }

   cin >> k;

   /*
   Algoritmo ineficiente:
      Copiar el vector en topk
      Ordenar topk
      Seleccionar los k primeros de topk
   */


   for (int i = 0; i < utilizados_vector; i++)
      topk[i] = vector[i];

   int i;
   double a_insertar;
        int j;

        for (i = 1; i < utilizados_vector; i++){
      a_insertar = topk[i];

      for (j = i; j > 0 && a_insertar > topk[j-1]; j--)   // Ordenación de mayor a menor
         topk[j] = topk[j-1];

      topk[j] = a_insertar;
   }

   for (int i = 0; i < k; i++){
      cout << topk[i] << " ";
   }
}
```

```
87:
```

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

#include <iostream>
using namespace std;

int main(){
   const int MAX_FIL = 10, MAX_COL = 10;
   double matrA[MAX_FIL][MAX_COL],matrB[MAX_FIL][MAX_COL], multpl[MAX_FIL][MAX_COL];
   double trasp[MAX_COL][MAX_FIL];
   int n, m,k;
   int a_insertar;

        //Entrada de datos

        cout << "Introduce el numero de filas y columnas de la primera matriz con un máximo de "
        << MAX_FIL << " filas" << " y " << MAX_COL << " columnas." << endl;

        //matrA--> n x m
   cin >> n;
   cin >> m;

   for (int i=0; i<n; i++)
      for (int j=0; j<m; j++)
         cin >> matrA[i][j];

        //matrB--> m x k
   cout << "Introduce el numero columnas de la segunda matriz con un máximo de "
        << MAX_COL << " columnas." << endl;

        cin >> k;

        for (int i=0; i<m; i++)
      for (int j=0; j<k; j++)
         cin >> matrB[i][j];

   ////////////////////////////////////////////////////////////////////////

   for(int l=0; l<k; l++){
      for(int j=0; j<n; j++){
         a_insertar=0;
         for(int i=0; i<m; i++){
            a_insertar = a_insertar + matrA[j][i] * matrB[i][l];
         }
         multpl[j][l]= a_insertar;
      }
   }

        //Salida de Datos

        cout << "\n\n";
   cout << "Matriz primera:\n";

   for (int i=0; i<n; i++){
      cout << "\n";

      for (int j=0; j<m; j++)
         cout << matrA[i][j] << '\t';
   }

        cout << "\n\n";
   cout << "Matriz segunda:\n";

   for (int i=0; i<m; i++){
      cout << "\n";

      for (int j=0; j<k; j++)
         cout << matrB[i][j] << '\t';
   }

        cout << "\n\n";
   cout << "Matriz multiplicada:\n";

   for (int i=0; i<n; i++){
      cout << "\n";

      for (int j=0; j<k; j++)
         cout << multpl[i][j] << '\t';
   }
}
```

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

// Sistema de D'Hondt

#include <iostream>
using namespace std;

int main (){
        const int NUM_MAX_PARTIDOS = 10;
        int total_escanios, total_partidos, posicion_mayor_cociente, i, j;
        double mayor_cociente = -1;
        double numero_votos[NUM_MAX_PARTIDOS];
        double numero_escanios[NUM_MAX_PARTIDOS] = {0};
        double cociente_dhondt[NUM_MAX_PARTIDOS];

        cout << "¿Número total de escaños a distribuir?: ";
        cin >> total_escanios;
        cout << "¿Cúantos partidos han participado en las elecciones?: ";
        cin >> total_partidos;
        cout << "Introduzca por orden el número de votos que ha obtenido cada partido: ";

        for (int i = 0 ; i < total_partidos ; i++){
                cin >> numero_votos[i];
        }

        for (i = 0 ; i < total_escanios ; i++){
                for (j = 0 ; j < total_partidos ; j++){
                        cociente_dhondt[j] = numero_votos[j] / (numero_escanios[j] + 1);

                        if (cociente_dhondt[j] > mayor_cociente){
                                mayor_cociente = cociente_dhondt[j];
                                posicion_mayor_cociente = j;
                        }

                }
                mayor_cociente = -1;
                numero_escanios[posicion_mayor_cociente]++;
        }

        for (i = 0 ; i < total_partidos ; i++){
                cout << " " << numero_escanios[i];
        }

}
```