```
    1: #include <iostream>
    2: #include <cmath>
    3: double g;                             //g son los grados.
    4: const double PI=6*asin(0.5);         //Se calcula pi usando el arcoseno ya que pi/6=arcsen(0.5), despejan
do queda pi=6*arcsen(0.5)
    5: int main()
    6: {
    7:     cout << "Este programa convierte los grados en radianes.\n\n";
    8:
    9:     cout << "Introduzca los grados(dos numeros enteros):";
   10:     cin >> g;
   11:
   12:     cout <<"\n    |" << g << " grados son " << g*(PI/180) << " radianes \n" ;
   13: }
```

```
    1: #include <iostream>
    2: #include <cmath>
    3: double valor_inicial;
    4: double valor_final;
    5:
    6: int main()
    7:  {
    8:       cout <<"Este programa calcula la variacion porcentual de dos valores(inicial y final).\n\n";
    9:      cout <<"Introduce el valor inicial: ";
   10:      cin >> valor_inicial;
   11:      cout <<"Introduce el valor final: ";
   12:      cin >> valor_final;
   13:
   14:      cout <<"\nLa variacion porcentual es de: " << abs(100*((valor_inicial - valor_final)/valor_inicial))
 <<" %\n" ;
   15:  }
```

```cpp
#include <iostream>
#include <cmath>
double PI;
int main()
{
    cout << 6*asin(0.5) ;
}
```

```cpp
#include <iostream>
#include <cmath>

using namespace std;

double numero ;
double cifra ;

int main()
{

    cout <<"\nEscribe el numero que quiere redondear y la cifra decimal:" ;
    cin >> numero >> cifra ;
    numero= round(numero * pow(10, cifra));
    numero = numero/ pow (10, cifra);
    cout << "\nEl numero aproximado es: " << numero ;
}
//Este codigo no compila en codeblocks pero si en dev c++, no entiendo muy bien por qué.

```

```
    1: #include <iostream>
    2:
    3: double x1,x2;           //Coordenadas de P1.
    4: double y1,y2;           //Coordenadas de P2.
    5:
    6: int main()
    7: {
    8:     cout << "Este programa calcula la distancia entre dos puntos dados.\n\n";
    9:
   10:     cout << " Coordanada x del P1: ";
   11:     cin >> x1;
   12:     cout << " Coordanada y del P1: ";
   13:     cin >> y1;
   14:
   15:     cout << "\n Coordanada x del P2: ";
   16:     cin >> x2;
   17:     cout << " Coordanada y del P2: ";
   18:     cin >> y2;
   19:
   20:     cout << "\n    -La distancia euclidea entre ambos puntos es de " << sqrt((x2-x1)*(x2-x1) + (y2-y1)*(
y2-y1)) << "\n";
   21: }
```

```
    1: #include <iostream>
    2:
    3: double capital;
    4: double interes;
    5: double total;
    6:
    7: int main()
    8: {
    9:     cout <<"Este programa calcula el capital total dado un interes bancario\n\n";
   10:
   11:     cout << "Introduzca su capital: ";
   12:     cin >> capital;
   13:     cout << "introduzca el interes: ";
   14:     cin >> interes;
   15:
   16:     total = capital + capital*(interes/100);
   17:     cout << "Su capital total es de " << total;
   18:
   19:     /*En la asignación que calcula la variable total, ¿se podría sustituir dicha variable por capital?*/
   20:
   21:     /**Si cambiamos la variable por capital, el código NO funcionará. Si queremos ahorrarnos la variable
total,
   22:     *debemos introducir los calculos dentro del cout.*/
   23:
   24: }
```

```cpp
#include <iostream>

int opcion;

double salario_base;              //En este ejercicio se utilizarán las tres opciones para calcular el salario final.
double salario_final;             //Al final del codigo se responde a la cuestion de cual es la alternativa más adecuada.

int main()
{
    cout <<"Este programa calcula el salario final tras aplicar un aumento del 2%, de tres formas distintas(internamente).\n\n";

    cout << " Elige una opcion:";
    cin >> opcion;

    //En la opcion 1 se utiliza unicamente la variable salario_base y se le multiplica por 1.02 dentro del cout.

    if(opcion==1)
    {
    cout << "\nIntroduce el salario:";
    cin >> salario_base;

    cout << "\n -El salario final es " <<1.02*salario_base <<"\n";
    }

    //En la opcion 2 se declara una nueva variable llamada salario_final que será el producto del salario inicial por 1.02.

    if(opcion==2)
    {
        cout << "\n Introduce el salario:";
        cin >>salario_base;

        salario_final=salario_base*1.02;
        cout << "\n -El salario final es " <<salario_final <<"\n";
    }

    //En la opcion 3 tenemos una única variable de nuevo que se modifica(multiplicando por 1.02) tras conocer su valor.

    if(opcion==3)
    {
        cout << "\nIntroduce el salario:";
        cin >> salario_base;

        salario_base = salario_base*1.02;
        cout << "\n -El salario final es " <<salario_base <<"\n";
    }

    /*a) Directamente hacer el cómputo 1.02 * salario_base dentro de la sentencia cout
      b) Introducir una variable salario_final, asignarle la expresión anterior y mostrar su contenido en la sentencia cout
        c) Modificar la variable original salario_base con el resultado de incrementarla un 2%.

        Indique qué alternativa elige y justifíquela.*/

        //La mejor alternativa es la primera ya que requiere de menos variables y menos lineas de codigo.

}
```

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main(){

    double x,y,z;
    double contenedor;

    cout << "Este programa intercambia el valor de tres variables.\n\n";

    cout <<"Introduce la variable x:";
    cin >> x;
    cout <<"Introduce la variable y: ";
    cin >> y;
    cout <<"Introduce la variable z: ";
    cin >> z;

    contenedor = x + y + z;
    x = contenedor - x - y;
    y = contenedor - y - z;
    z = contenedor - x - y;

    cout << " " << x << " " << y << " " << z;

}
```

```
    1: #include <iostream>
    2: using namespace std;
    3:
    4: double caja_izda;
    5: double caja_dcha;
    6: double intercambio;      //variable auxiliar
    7:
    8: int main()
    9: {
   10:     cout << "Este programa intercambia el valor de dos cajas, sin confundir al usuario.\n\n";
   11:
   12:     cout << "Introduzca el valor de la caja izquierda: ";
   13:     cin >> caja_izda;
   14:     cout << "Introduzca el valor de la caja derecha: ";
   15:     cin >> caja_dcha;
   16:
   17:     intercambio = caja_dcha;        //guardamos el valor de una de las cajas en la variable intercambio
   18:     caja_dcha = caja_izda;
   19:     caja_izda = intercambio;        //usando la variable intercambio, intercambiamos el valor de las caj
as
   20:
   21:     cout << "\nLa caja izquierda vale " << caja_izda << "\n";
   22:     cout << "La caja derecha vale " << caja_dcha;
   23: }
   24:
```

```cpp
#include <iostream>
using namespace std;

double h1, m1,s1;
double h2,m2,s2;
int main()
{
    cout << "Este programa calcula los segundos entre dos horas.\n\n";

    cout << "Introduce una hora(formato _h _m _s): ";
    cin >> h1 >> m1 >> s1;

    cout << "Introduce una hora distinta(formato _h _m _s): ";
    cin >> h2 >> m2 >> s2;

    cout << "Los segundos entre ambas horas son: " << abs(3600*(h1 - h2) + 60*(m1 - m2) + (s1 - s2));
}
```

```cpp
#include <iostream>
using namespace std;

double billete;
const double desc_puntos=0.96;           //4% descuento.
const double desc_vuelo_largo=0.98;      //2% descuento.
int main()
{
    cout << "Este programa calcula el precio de su billete aplicando un descuento.\n\n";

    cout << "Introduzca el precio de su billete:";
    cin >> billete;

    cout << "El precio de su billete es " << billete*desc_puntos << " " << billete*desc_vuelo_largo ;
}
```

```cpp
#include <iostream>
using namespace std;

const double tarifa_fija=150;
double km;

int main()
{
    cout << "Este programa calcula el precio final de un vuelo convencional.\n\n";

    cout <<"Introduce la distancia a su destino(en km):";
    cin >>km;

    cout <<"El precio de su vuelo es de " <<tarifa_fija+km*0.10 << " euros";
}
```

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main(){

    double x,y,z;

    cout << "Este programa intercambia el valor de tres variables.\n\n";

    cout <<"Introduce la variable x:";
    cin >> x;
    cout <<"Introduce la variable y: ";
    cin >> y;
    cout <<"Introduce la variable z: ";
    cin >> z;

    x = x + y + z; //aplicando un "truquillo" de restas y sumas, podemos intercambiar las variables sin añadir auxiliares
    y = x - y - z;
    z = x - y - z;
    x = x - y -z;


    cout << " " << x << " " << y << " " << z;


}
```

```
    1: #include <iostream>
    2: #include <cmath>
    3: using namespace std;
    4:
    5: double x,esperanza,desviacion,abcs_x;
    6: const double PI = 6 * asin(0.5);
    7:
    8: int main(){
    9:
   10:     cout <<"Este progama calcula una gausiana\n\n";
   11:
   12:     cout <<"Introduce la esperanza: ";
   13:     cin >> esperanza;
   14:     cout <<"\nIntroduce la desviacion: ";
   15:     cin >> desviacion;
   16:     cout <<"\nIntroduce el valor de abcisa: ";
   17:     cin >> abcs_x;
   18:
   19:     x = (1/(desviacion*sqrt(2*PI)))*exp(-0.5*((abcs_x - esperanza)/desviacion)*((abcs_x - esperanza)/des
viacion));
   20:     cout << x;
   21: }
   22:
```

```cpp
#include <iostream>
using namespace std;

double metros;                                  //variable de entrada
double pulgada, pie, yarda,milla,m_marina;  //variables de salida

int main()
{
    cout <<"Este programa convierte metros en pulgadas, pies, yardas y millas.\n";

    cout <<"Introduce la distancia en metros: ";
    cin >> metros;

    pulgada = metros/0.0254;                //Conversión metros a pulgadas
    pie = metros/0.3048;                    //Conversión metros a pies
    yarda = metros/0.9144;                  //Conversión metros a yardas
    milla = metros/1609.344;                //Conversión metros a millas
    m_marina = metros/1852;                 //Conversión metros a millas náuticas


    cout << "\nLa distancia en pulgadas es: " << pulgada;
    cout << "\nLa distancia en pies es: " << pie;
    cout << "\nLa distancia en yardas es: " << yarda;
    cout << "\nLa distancia en millas es: " << milla;
    cout << "\nLa distancia en millas náuticas es: " << m_marina;
}

```

```
    1: #include <iostream>
    2: #include <cmath>
    3: using namespace std;
    4: int main()
    5: {
    6:     //...
    7:     cout <<"\n  Ejercicio 14.4\n";
    8:     cout <<"  Este programa calcula el valor de PI\n\n";  /**Se calcula pi usando el arcoseno ya que pi/
6=arcsen(0.5),
    9:                                                          *despejando queda pi=6*arcsen(0.5)*/
   10:     const double PI = 6*asin(0.5);
   11:
   12:     cout << "  " << PI << "\n\n";
   13:
   14:     //-------------------------------------------------------------------------------------------------
-----------------------------
   15:     cout <<"\n  Ejercicio 14.3\n";
   16:     cout << "  Este programa calcula el area y la longitud de una circunferencia.\n\n";
   17:
   18:     double radio_cir;
   19:
   20:     cout <<"   Introduce el radio de la circunferencia: ";
   21:     cin >> radio_cir;
   22:
   23:     cout <<"\n  -El area de la circunferencia es: " << PI*radio_cir*radio_cir <<" cm cuadrados.";
   24:     cout <<"\n  -La longitud de la circunferencia es: " << 2*PI*radio_cir <<" cm.\n";
   25:
   26:     //-------------------------------------------------------------------------------------------------
-----------------------------
   27:     cout <<"\n  Ejercicio 14.9\n";
   28:     cout << "  Este programa convierte los grados en radianes.\n\n";
   29:
   30:     double g;                           //g son los grados.
   31:
   32:     cout << "  Introduzca los grados(dos numeros enteros):";
   33:     cin >> g;
   34:
   35:     cout <<"\n   |" << g << " grados son " << g*(PI/180) << " radianes \n" ;
   36:
   37:     //-------------------------------------------------------------------------------------------------
-----------------------------
   38:     cout <<"\n  Ejercicio 14.11\n";
   39:     cout << "  Este programa calcula el precio final de un vuelo convencional.\n\n";
   40:
   41:     const double tarifa_fija=150;
   42:     const double tarifa_variable=0.10;
   43:     double km;
   44:
   45:     cout <<"  Introduce la distancia a su destino(en km):";
   46:     cin >>km;
   47:
   48:     cout <<"\n  El precio de su vuelo es de " <<tarifa_fija + km*tarifa_variable << " euros \n";
   49:
   50:     //-------------------------------------------------------------------------------------------------
-----------------------------
   51:     cout <<"\n  Ejercicio 14.12\n";
   52:     cout << "  Este programa calcula el precio de su billete aplicando dos descuentos.\n\n";
   53:
   54:     double billete;
   55:     const double desc_puntos=0.96;          //4% descuento.
   56:     const double desc_vuelo_largo=0.98;     //2% descuento.
   57:
   58:     cout << "  Introduzca el precio de su billete:";
   59:     cin >> billete;
   60:
   61:     cout << "\n  El precio de su billete es " << billete*desc_puntos << " euros y " << billete*desc_vuel
o_largo << " euros\n" ;
   62: }
   63:
   64:
```

```cpp
#include <iostream>
#include <cmath>

using namespace std;

int main()
{
    int edad_persona;       //La edad de una persona no va a superar nunca un número como 2^32, así que un int o un short es suficiente

    long long pib_pais;     /*El PIB de un país si que puede superar el limite de 2^32,
                            por eso usamos el long long que llega hasta 2^64 sin perder precision como el double.*/

    bool primo;             //Para saber si es primo o no, usaremos un bool que nos dirá si lo es o no.

    string estado_civil;    //Para diferenciar y guardar correctamente variables de palabras, se usa string.

    bool sexo;              //Como son 2 opciones solo, se puede asignar un sexo a un estado. Ej. Hombre=1 y mujer=0.
}


```

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

#include <iostream>
#include <cmath>

using namespace std;

int main()
{

    int edad, adivine, anio, velocidad;
    char car, vocal;
    bool b_car, b_edad, b_adivine, b_anio, b_velocidad, b_vocal;

    cout << "Este programa comprueba si la letra es minuscula o no\n\n";

    cout << "Introduce una letra minuscula: ";
    cin >> car;

    b_car = false;

    if(car>=97 && car<=124)
        b_car = true;

    cout << b_car << endl;

    //...

    cout << "\nEste programa comprueba si la edad es menor 18 o mayor que 65\n\n";

    cout << "Introduce la edad: ";
    cin >> edad;

    b_edad = false;

    if(edad<=18 | edad>=65)
        b_edad=true;

    cout << b_edad << endl;

    //...

    cout << "\nEste programa comprueba si el numero esta entre 1 y 100\n\n";

    cout << "Introduce un numero del 1 al 100: ";
    cin >> adivine;

    b_adivine = false;

    if(adivine>=1 && adivine<=100)
        b_adivine = true;

    cout << b_adivine;

    //...

    cout << "\nEste programa comprueba si es un anio bisiesto o no\n\n";

    cout << "Introduce un anio: ";
    cin >> anio;

    b_anio = false;

    if(anio%4==0)
    {
        b_anio = true;

        if(anio%100==0)
            b_anio=false;
    }
    if(anio%400==0)
        b_anio = true;

    cout << b_anio << endl;

    //...

    cout << "\nEste programa comprueba si vas a mas de 100 km/h: ";

```

```
87:     cout << "Introduce una velocidad en km/h: ";
88:     cin >> velocidad;
89:
90:     b_velocidad = false;
91:
92:     if(velocidad>=100)
93:         b_velocidad = true;
94:
95:     cout << b_velocidad << endl;
96:
97:     //...
98:
99:     cout << "Este programa comprueba si es vocal o consonante\n\n";
100:
101:    cout << "Introduce una vocal: ";
102:    cin >> vocal;
103:
104:    b_vocal = false;
105:
106:    if(vocal=='a'| vocal=='e' | vocal=='i' | vocal=='o' | vocal=='u')
107:        b_vocal = true;
108:
109:    cout << b_vocal << endl;
110: }
```

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main(){

    char letra;

    cout << "\nIntroduzca una letra a continuacion:";
    cin >> letra;

    if('A' <= letra && letra <= 'Z')
    {
    cout << "\nLa entrada es mayuscula";
    letra = letra + 32;
    }
    else
    {
    cout <<"\nLa entrada es minuscula";
    letra = letra - 32;
    }
    cout << "\nEl resultado es: " << letra;
}

```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11: #include <cmath>
12:
13: using namespace std;
14:
15: int main()
16: {
17:    char tipo_cota_inf, coma_sep, tipo_cota_sup;
18:    int cota_inf,cota_sup;
19:
20:    cout <<"Introduce un intervalo: ";
21:    cin >> tipo_cota_inf >> cota_inf >> coma_sep >> cota_sup >> tipo_cota_sup;
22:
23:    cout << "\n" << tipo_cota_inf << cota_inf << coma_sep << cota_sup << tipo_cota_sup;
24:
25: }
```

```cpp
#include <iostream>
#include <cmath>

using namespace std;

int main(){

    cout << "Este programa aproxima decimales con la funcion trunc\n\n";

    double r,n;
    double dsplz, num_redondeado;

    cout << "Introduce el numero a redondear: ";
    cin >> r;
    cout << "\nIntroduce el numero de decimales a truncar: ";
    cin >> n;

    dsplz = pow(10,n);
    num_redondeado = trunc(r*dsplz)/dsplz;

    cout << num_redondeado;

    //igual que en el ejercicio 10, no compila en C pero si en dev C++...
}
```

```cpp
#include <iostream>
#include <cmath>

using namespace std;

int main()
{
    int a,b;
    bool div;

    cout << "Este programa comprueba si cualquiera de los dos numeros son divisibles\n\n";
    cout << "Introduce dos numeros: ";
    cin >> a >> b;

    div = a%b==0 || b%a==0;

    if(div)
    {
     cout << "Los numeros son divisibles.";
    }
    else
        cout << "No son divisibles.";
}


```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: /* Aprovechando el ejercicio 13 que ya hicimos, simplemente comprobamos si la diferencia es negativa
11: o no para saber cual va primero.
12:
13:     cout << "Este programa calcula los segundos entre dos horas.\n\n";
14:
15:     cout << "Introduce una hora(formato _h _m _s): ";
16:     cin >> h1 >> m1 >> s1;
17:
18:     cout << "Introduce una hora distinta(formato _h _m _s): ";
19:     cin >> h2 >> m2 >> s2;
20:
21:     diferencia=3600*(h1 - h2) + 60*(m1 - m2) + (s1 - s2);
22:
23:     if(diferencia<0)
24:         cout << "\nEl primero SI es anterior." << endl;
25:     else
26:         cout << "\nEl primero NO es anterior." << endl;
27: */
28:
29: #include <iostream>
30: using namespace std;
31:
32: int main()
33: {
34:     double h1, m1,s1;
35:     double h2,m2,s2;
36:     bool COMPARADOR;
37:
38:     cout << "Este programa calcula los segundos entre dos horas.\n\n";
39:
40:     cout << "Introduce una hora(formato _h _m _s): ";
41:     cin >> h1 >> m1 >> s1;
42:     cout << "Introduce una hora distinta(formato _h _m _s): ";
43:     cin >> h2 >> m2 >> s2;
44:
45:     //Proceso...
46:
47:     if(h1!=h2)
48:     {
49:         if(h1<h2)
50:             COMPARADOR=true;
51:         if(h1>h2)
52:             COMPARADOR=false;
53:     }
54:     else
55:         {
56:             if(m1!=m2)
57:             {
58:                 if(m1<m2)
59:                     COMPARADOR=true;
60:                 if(m1>m2)
61:                     COMPARADOR=false;
62:             }
63:             else
64:                 {
65:                     if(s1!=s2)
66:                     {
67:                         if(s1<s2)
68:                             COMPARADOR=true;
69:                         if(s1>s2)
70:                             COMPARADOR=false;
71:                     }
72:                     else
73:                         COMPARADOR=false;
74:                 }
75:         }
76:
77:
78:     //Salida de datos...
79:
80:     if(COMPARADOR==true)
81:         cout << "\nEl primero SI es anterior." << endl;
82:     else
83:         cout << "\nEl primero NO es anterior." << endl;
84: }
85:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11: #include <cmath>
12:
13: using namespace std;
14:
15: int main()
16: {
17:     int anio;
18:     bool b_anio;
19:
20:     cout << "\nEste programa comprueba si es un anio bisiesto o no\n\n";
21:
22:     cout << "Introduce un anio: ";
23:     cin >> anio;
24:
25:     b_anio = false;
26:
27:     if(anio%4==0)
28:     {
29:         b_anio = true;
30:
31:         if(anio%100==0)
32:             b_anio=false;
33:     }
34:
35:     if(anio%400==0)
36:         b_anio = true;
37:
38:     if(b_anio)
39:         cout << "Es un anio bisiesto.";
40:     else
41:         cout << "No es un anio bisiesto.";
42:
43: }
44:
```

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

#include <iostream>
#include <cmath>
using namespace std;

double x,media,desviacion,minim,maxim, incremento;
const double PI = 6 * asin(0.5);

int main(){

    cout <<"ESTE PROGRAMA CALCULA LA GAUSSIANA\n";
    cout <<"----------------------------------\n\n";

    cout <<"Introduce la Media: ";
    cin >> media;

    do{
        cout <<"\nIntroduce la Desviacion: ";
        cin >> desviacion;
    }while(desviacion<0);

    cout <<"\nIntroduce el valor Minimo de abcisa: ";
    cin >> minim;
    cout <<"\nIntroduce el valor Maximo de abcisa: ";
    cin >> maxim;
    cout <<"\nIntroduce el Incremento: ";
    cin >> incremento;

    cout << "\n        SOLUCIONES\n";
    cout << "       ------------" << endl << endl;
    while(minim<=maxim){
    x = (1/(desviacion*sqrt(2*PI)))*exp(-0.5*((minim - media)/desviacion)*((minim - media)/desviacion));
    cout <<"Gaussiana(" << minim << ") es " << x << endl;
    minim = minim + incremento;
    }
}
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11: #include <cmath>
12:
13: using namespace std;
14:
15: int main()
16: {
17:     char RADAR;
18:     double vel;
19:     long double vel_imputada;
20:
21:     cout << "Este programa calcula la velocidad imputada de un vehiculo: \n\n";
22:
23:     cout << "\n Introduzca el tipo de radar (F para fijo y otra letra para movil): ";
24:     cin >> RADAR;
25:     cout << " Introduzca la velocidad del vehiculo: ";
26:     cin >> vel;
27:
28:     if(RADAR =='F')
29:     {
30:     vel_imputada = vel*0.95;
31:     }
32:     else
33:         vel_imputada = vel*0.93;
34:
35:     cout << "\n   La velocidad imputada es " << vel_imputada << endl;
36:
37:     return 0;
38: }
39:
40:
```

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

#include <iostream>
#include <cmath>

using namespace std;

int main()
{
    const double tarifa_base=150;
    double tarifa_variable,tarifa_final;
    int puntos;
    double km;

    cout << "Este programa calcula el precio final de un vuelo convencional.\n\n";

    cout <<"Introduce la distancia a su destino(en km):";
    cin >>km;
    cout << "Introduzca sus puntos: ";
    cin >>puntos;

    tarifa_variable=(km-300)*0.1;

    if(km>300)
        tarifa_final=tarifa_base + tarifa_variable;
    else
        tarifa_final=tarifa_base;
    if(puntos>=100)
    {
        if(puntos>=200)
            tarifa_final = tarifa_final*0.96;       //descuento 4%
        else
            tarifa_final = tarifa_final*0.97;       //descuento 3%
    }
    if(km>=700)
        tarifa_final = tarifa_final*0.98;           //desuento 2%

    cout << "El precio final es: " << tarifa_final << endl;
}
```

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

#include <iostream>
#include <cmath>

using namespace std;

int main()
{
    // a)

    char tipo_radar;
    cin >> tipo_radar;
    if (tipo_radar == 'F' && tipo_radar == 'f')
    .....

    /* No se puede hacer igualdades con char de esa manera, es con un = solo.*/

    // b)

    double velocidad;
    cin >> velocidad;
    if (velocidad > 100 && velocidad < 70)
    cout << "\nVelocidad fuera del rango";

    /* El problema de este programa es que no existe una velocidad mayor a 100 y a la vez sea menor que
70
    en todo caso debe ser o uno u otro, usando "||".*/

    // c)

    double velocidad;
    cin >> velocidad;
    if (velocidad > 100 || velocidad > 110)
    cout << "Velocidad excesiva";

    /*Se repite código innecesariamente, lo correcto es poner "velocidad > 100" y yasta.*/
}
```

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


#include <iostream>
#include <cmath>

using namespace std;

int main()
{
    const double tarifa_base=150;
    double tarifa_variable,tarifa_final;
    bool UMBRAL_KM;
    double km;

    cout << "Este programa calcula el precio final de un vuelo convencional.\n\n";

    cout <<"Introduce la distancia a su destino(en km):";
    cin >>km;

    UMBRAL_KM=km>=300;
    tarifa_variable=(km-300)*0.1;

    if(UMBRAL_KM)
        tarifa_final=tarifa_base + tarifa_variable;
    else
        tarifa_final=tarifa_base;

    cout << "El precio final es: " << tarifa_final << endl;
}


```

```cpp
#include <iostream>
#include <cmath>
using namespace std;

/*17. [Variación porcentual: lectura de varios valores] Recupere la solución del ejercicio 5
[Variación porcentual] de la Relación de Problemas I. Modifíquelo para realizar una
lectura de múltiples pares de valores. La entrada de datos se interrumpirá cuando
se introduzca cualquier valor negativo. Para simplificar, supondremos que si el primer
valor introducido es positivo, el usuario también introducirá un positivo como segundo
valor.
Por cada par de valores, el programa mostrará la correspondiente variación porcentual.
En este ejercicio puede mezclar entradas de datos con salidas y cómputos, dentro
del mismo bucle.
*/

int main()
{
   double valor1,valor2, variacion;
   cout << "Este programa calcula la variacion porcentual de dos valores"
        << "(si el primero es positivo, el segundo tambien)\n hasta introducir un valor negativo\n" << e
ndl;
   do{
      cout << "Introduce el primer valor: ";
      cin >> valor1;
      if(valor1>=0){
         cout << "Ahora introduce el segundo: ";
         cin >> valor2;
         variacion = abs(100*((valor1 - valor2)/valor1));
         cout << "La variacion porcentual es " << variacion << "%" << endl;
         cout << "------------------------------" << endl;
      }
   }
   while(valor1>=0);


}
```

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main() {

   enum class tipo
   {mayuscula, minuscula, otro};

   char letra_introducida;
   char letra_convertida;
   const int DISTANCIA_MAY_MIN = 'a'-'A';
   tipo caracter;

//Entrada de Datos...

   cout << "\nIntroduzca una letra --->";
   cin >> letra_introducida;

//Calculo de Datos...

   if ((letra_introducida >= 'A') && (letra_introducida <= 'Z')){
      letra_convertida = letra_introducida + DISTANCIA_MAY_MIN;
      caracter = tipo::mayuscula;
   }
   else if ((letra_introducida >= 'a') && (letra_introducida <= 'z')){
      letra_convertida = letra_introducida - DISTANCIA_MAY_MIN;
      caracter = tipo::minuscula;
   }
   else
      caracter = tipo::otro;

//Salida de Datos...

   if(caracter == tipo::mayuscula)
      cout << "La letra convertida es: " << letra_convertida;
   else if(caracter == tipo::minuscula)
      cout << "La letra convertida es: " << letra_convertida;
   else
            cout << "El caracter no era una letra";
   return 0;
}
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: // Coranovirus
11:
12: /*
13: Por todos es conocido el gran daño humano y económico que ha producido
14: el coranovirus ARS-CoV-2 al provocar la enfermedad denominada COVID-19.
15: En Argentina, la web gubernamental https://coronavirus.argentina.gob.
16: ar/ diseñó un test aproximado para su identificación. El código fuente estaba escrito
17: en JavaScript y parte de él se muestra en la siguiente figura (como puede apreciar,
18: Java es un lenguaje con una sintaxis similar a C++)
19:
20: Suponemos que la lógica del código es correcta y por tanto identifica adecuadamente
21: la enfermedad del COVID-19. Sin embargo, este código tiene varios problemas
22: (a los pocos días de publicarlo arreglaron parte de ellos) en el sentido de
23: que incumple algunas normas que un buen programador ha de seguir. Uno de los
24: problemas más sencillos de detectar es la aparición duplicada de las comprobaciones
25: sobre respiratoryDisease. En cualquier caso, para simplificar el problema
26: nos vamos a fijar únicamente en cuatro síntomas, a saber, bodyTemperature,
27: difficultyBreathing, diabetes y cancer, de forma que el código anterior se
28: simplificaría en el siguiente:
29:
30: if((bodyTemperature >= 38 && difficultyBreathing) ||
31:    (bodyTemperature >= 38 && difficultyBreathing && diabetes) ||
32:    (bodyTemperature >= 38 && difficultyBreathing && cancer) ||
33:    (bodyTemperature >= 38 && diabetes) ||
34:    (bodyTemperature >= 38 && cancer))
35:
36:    cout << "Consulte autoridades locales";
37: else
38:    if (bodyTemperature >= 38)
39:       cout << "Cuarentena";
40:    else
41:       if (bodyTemperature < 38)
42:          cout << "Test negativo";
43:       else
44:          cout << "Test negativo";
45:
46: Identifique los problemas que pueda haber y proponga una solución modificando el
47: código del programa que puede encontrar en el siguiente enlace:
48: http://decsai.ugr.es/jccubero/FP/II_CoranovirusEsbozo.cpp
49:
50: Ejemplo de entrada: 37 S S S
51: .... Salida correcta: Test negativo
52: Ejemplo de entrada: 39 S N N
53: .... Salida correcta: Consulte autoridades locales
54: Ejemplo de entrada: 39 N N N
55: .... Salida correcta: Cuarentena en su casa
56: */
57:
58: #include <iostream>
59: #include <cctype>
60: using namespace std;
61:
62: int main(){
63:    double bodyTemperature;
64:    bool difficultyBreathing, diabetes, cancer;
65:    char opcion;
66:
67:    cout << "Detección Aproximada de COVID-19\n\n\n"
68:         << "Introduzca la temperatura y a continuacion conteste S/N a las siguientes preguntas:\n\n"
69:         << "- Tiene dificultades para respirar?\n"
70:         << "- Es diabetico?\n"
71:         << "- Tiene algun tipo de cancer?\n\n";
72:
73:    cin >> bodyTemperature;
74:    cin >> opcion;
75:    difficultyBreathing = toupper(opcion) == 'S';
76:    cin >> opcion;
77:    diabetes = toupper(opcion) == 'S';
78:    cin >> opcion;
79:    cancer = toupper(opcion) == 'S';
80:
81:    /*
82:    El siguiente código es el que aparecía en la página web.
83:    Arréglelo
84:    */
85:
86:    if(bodyTemperature >= 38 && (difficultyBreathing || diabetes ||cancer))
```

```
87:       cout << "Consulte autoridades locales";
88:    else
89:       if (bodyTemperature >= 38)
90:          cout << "Cuarentena en su casa";
91:       else
92:          if (bodyTemperature < 38)
93:             cout << "Test negativo";
94:          else
95:             cout << "Test negativo";
96: }
```

```
    1: ////////////////////////////////////////////////////////////////////////
    2: //
    3: // Fundamentos de Programación
    4: // ETS Informática y Telecomunicaciones
    5: // Universidad de Granada
    6: // // Don Oreo
    7: //
    8: ////////////////////////////////////////////////////////////////////////
    9:
   10: #include <iostream>
   11: #include <cmath>
   12:
   13: using namespace std;
   14:
   15: int main()
   16: {
   17:     char tipo_cota_inf, coma_sep, tipo_cota_sup;
   18:     double cota_inf,cota_sup;
   19:     double valor;
   20:     bool intervalo;
   21:
   22:     //entrada de datos...
   23:     cout <<"Introduce un intervalo: ";
   24:     cin >> tipo_cota_inf >> cota_inf >> coma_sep >> cota_sup >> tipo_cota_sup;
   25:     cout <<"Introduce un valor: ";
   26:     cin >> valor;
   27:
   28:     //proceso de datos...
   29:
   30:     if(valor==cota_inf && tipo_cota_inf=='[')
   31:         intervalo=true;
   32:     else if(valor==cota_sup && tipo_cota_sup==']')
   33:         intervalo=true;
   34:     else if(valor>cota_inf && valor<cota_sup)
   35:         intervalo=true;
   36:     else
   37:         intervalo=false;
   38:
   39:     //Salida de datos...
   40:
   41:     if(intervalo)
   42:         cout << "El valor " << valor << " esta dentro del intervalo " << tipo_cota_inf << cota_inf << co
ma_sep << cota_sup << tipo_cota_sup << endl;
   43:     else
   44:         cout << "El valor " << valor << " NO esta dentro del intervalo " << tipo_cota_inf << cota_inf <<
 coma_sep << cota_sup << tipo_cota_sup << endl;
   45:
   46: }
   47:
```

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

//Este programa calcula si un numero es narcicista o no.

#include <iostream>
#include <cmath>

using namespace std;

int main(){
   long int num_digitos = 0;
   long int suma = 0;
   long int n, digito;
   long int guardado;
   cout << "Introduce un entero: ";
   cin >> n;

//computo...

   guardado = n;

   while(n>0){
      n = n/10;
      num_digitos++;
   }

   n=guardado;     //restauramos n

   for(int i=0;i<num_digitos;i++){
      digito = n%10;
      suma = suma + pow(digito,num_digitos);
      n = n / 10;
   }
   n=guardado;     //restauramos n

   if(suma==n)
      cout << "El numero es narcicista.";
   else
      cout << "El numero no es narcicista.";

}
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11: #include <cmath>
12:
13: using namespace std;
14:
15: int main(){
16:
17:    double capital, interes;
18:
19:    cout << "Introduce su capital: ";
20:    cin >> capital;
21:    cout << "Introduce el interes: ";
22:    cin >> interes;
23:    double anios=0;
24:    double tope_cap=(2*capital);
25:
26:    do{
27:          capital = capital + capital*(interes/100);
28:          anios++;
29:    }
30:    while(capital<=tope_cap);
31:
32:    cout << "\nPara doblar la cantidad inicial han de pasar " << anios << " anios" << endl;
33:    cout << "Al finalizar, se obtendra un total de " << capital << " euros" << endl;
34:
35: }
36:
37:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11: #include <cmath>
12:
13: using namespace std;
14:
15: int main(){
16:
17: double C,I,tope_N;
18: double elev;
19: double N,M;
20:
21: /*donde: C es el dinero original
22: I es el interes
23: M es el interes compuesto*/
24:
25:    cout << "Introduce el capital: ";
26:    cin >> C;
27:    cout << "Introduce el interes: ";
28:    cin >> I;
29:    cout << "Introduce los anios a invetir: ";
30:    cin >> tope_N;
31:
32:    for(N=0;N<tope_N;N++)
33:       {
34:       elev = pow((1 + I/100),N+1);
35:       M = C*elev;
36:       cout << "Capital obtenido transcurrido el anio numero " << N
37:            << " = " << M << endl;
38:    }
39:
40: }
41:
42:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: /* ENUNCIADO:
11:    29. [Secuencia de temperaturas] Construya un programa que calcule cuándo se produjo
12:    la mayor secuencia de días consecutivos con temperaturas crecientes. El programa
13:    leerá una secuencia de reales representando temperaturas. Una temperatura es correcta
14:    si su valor está en el intervalo [-90, 60] (los extremos representan las temperaturas
15:    extremas registradas en la Tierra). La entrada de datos terminará cuando
16:    se introduzca una temperatura fuera del rango anterior. El programa debe calcular la
17:    subsecuencia de números ordenada, de menor a mayor, de mayor longitud.
18:    El programa nos debe decir la posición donde comienza la subsecuencia y su longitud.
19:    Por ejemplo, ante la entrada siguiente:
20:
21:    17.2 17.3 16.2 16.4 17.1 19.2 18.9 100
22:
23:    El programa nos debe indicar que la mayor subsecuencia empieza en la posición 3 (en
24:    el 16.2) y tiene longitud 4 (termina en 19.2)
25:    Considere los siguientes consejos:
26:       -Tendrá que leer sobre dos variables anterior y actual, para así poder comparar
27:       el valor actual con el anterior.
28:       -Se recomienda que use la técnica de lectura anticipada, por lo que tendrá que
29:       leer un primer valor y comprobar si está en el rango adecuado:
30:
31:    cin >> anterior;
32:    final_entrada_datos = anterior < MIN_TEMP
33:                                   ||
34:                                   anterior > MAX_TEMP;
35:    while (! final_entrada_datos){
36:       cin >> actual;
37:       ......
38:    }
39:
40:    Dentro del cuerpo del bucle tendrá que comparar el valor actual con los extremos
41:    del rango de temperaturas, tal y como se hizo antes de entrar al bucle
42:    con el valor anterior. Esto hace que repitamos un código muy parecido. Lo
43:    resolveremos cuando veamos las funciones.
44:    Ejemplo de entrada: 17.2 17.3 16.2 16.4 17.1 19.2 18.9 100
45:       -Salida correcta: Inicio: 3 Longitud: 4
46:    Ejemplo de entrada: 17.2 17.3 16.2 16.4 17.1 19.2 100
47:       -Salida correcta: Inicio: 3 Longitud: 4
48:    Ejemplo de entrada: 17.2 17.3 100
49:       -Salida correcta: Inicio: 1 Longitud: 2
50:    Ejemplo de entrada: 17.2 15.3 100
51:       -Salida correcta: Inicio: 2 Longitud: 1
52:    Ejemplo de entrada: 17.2 100
53:       -Salida correcta: Inicio: 1 Longitud: 1
54:    Ejemplo de entrada: 100
55:       -Salida correcta: Inicio: 1 Longitud: 0
56:    Finalidad: Trabajar con bucles que comparan un valor actual con otro anterior. Dificultad
57:    Media.
58:    */
59:
60: #include <iostream>
61: #include <cmath>
62:
63: using namespace std;
64:
65: int main(){
66:    /*Intervalo válido de temperatura [-90,60]
67:    Límite de registro de temperaturas de 100
68:    El terminador será true y false en función del intervalo [-90,60]
69:    */
70:    double temperatura;
71:    const int MIN_TEMP=-90, MAX_TEMP=60;
72:    const int LIM_TEMP=100;
73:    double registro_temp[LIM_TEMP];
74:    bool terminador;
75:    double minimo=MAX_TEMP;
76:
77:
78: //Entrada de datos...
79:
80:    cout << "Introduce las temperaturas: ";
81:
82:    /*sale del bucle cuando el numero de temperaturas excede el limite del vector o cuando una temperatur
a sale del rango [-90,60]
83:    Cada vez que metemos una temperatura, comprobamos que sea valida([-90,60]) con el bool terminador
84:    */
85:    int i=0;
```

```
   86:     int contador=-1;
   87:     int posicion;
   88:
   89:     while(contador<=LIM_TEMP && terminador==false){
   90:        cin >> temperatura;
   91:        terminador = (temperatura< MIN_TEMP || temperatura > MAX_TEMP);
   92:        registro_temp[i]=temperatura;
   93:        contador++;                        //Cuenta el numero de temperaturas para guardarlo en la variable
util después
   94:        i++;
   95:        if(temperatura<minimo){            //con el condicional guardamos la menor temperatura y su posicion
 gracias al contador
   96:           minimo = temperatura;
   97:           posicion = contador + 1 ;
   98:        }
   99:     }
  100:
  101:
  102: //Cálculo de datos...
  103:     /*guardamos el valor del contador en la variable util*/
  104:
  105:     int util = contador;
  106:     int longitud=1;
  107:     int anterior, actual;
  108:     //anterior y actual son los valores que utilizamos para comparar las variables del vector(en el bucle
 posterior)
  109:
  110:     //si util es 0 es porque hemos metido una temperatura invalida y entonces su longitud y posicion son
0
  111:     if(util>0){
  112:        for(i=posicion;i<=util;i++){
  113:           anterior=registro_temp[i];
  114:           actual=registro_temp[i+1];
  115:           if(anterior<actual){
  116:              longitud++;
  117:           }
  118:        }
  119:     }
  120:     else
  121:     {
  122:        longitud=0;
  123:        posicion=0;
  124:     }
  125:
  126: //Salida de datos...
  127:
  128:     cout << endl << "Inicio: " << posicion << " Longitud: " << longitud << endl;
  129:
  130:
  131:
  132:
  133: }
  134:
  135:
```

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

   /*[Aproximación de PI por Madhava sin usar pow] En el siglo XIV el matemático indio
   Madhava of Sangamagrama calculó el arco tangente a través de un desarrollo de
   Taylor (este tipo de desarrollos se ve en la asignatura de Cálculo)

   arctan(x) = ((-1)^i•x^(2i+1))/(2i+1)

   Usando como x el valor 1, obtenemos la serie de Leibniz vista en el ejercicio 26:

   arctan(x)= pi/4 = ((-1)^i)/(2i+1)

   Usando como x el valor 1/sqrt(3), obtenemos:

   arctan(1/sqrt(3))= pi/6 = ((-1)^i•(1/sqrt(3))^(2i+1))/(2i+1)

   Por lo tanto, podemos usar la siguiente aproximación:

   pi/6 = ((-1)^i•(1/sqrt(3))^(2i+1))/(2i+1)

      Construya un programa que lea el valor tope obligando a que esté entre 1 y cien
   mil, calcule la aproximación de PI mediante la anterior serie e imprima el resultado en
   pantalla.
   Importante: En la implementación de esta solución NO puede usar la función pow ni
   ningún condicional if. Se le pide expresamente que para el cómputo de cada término,
   intente aprovechar los cómputos realizados en la iteración anterior.
   Ejemplo de entrada: 1000 -- Salida correcta: 3.14159265358979
   Ejemplo de entrada: 100000 -- Salida correcta: 3.14159265358979
   */

#include <iostream>
#include <cmath>

using namespace std;

int main(){
   int tope;
   int i=0;
   int signo = 1;
   double numerador,denominador;
   double serie = 0;
   double pi_aprox;
   const int CAMBIO_SIGNO=-1;
   const double raiz= (1/sqrt(3));

   do{
   cout << "Introduce el tope: ";
   cin >> tope;
   }
   while(tope<0 || tope>1e+5);

   numerador= raiz;
   denominador=1.0;

   for(i=0;i<=tope;i++){

         serie += signo*numerador/denominador;

         signo *= CAMBIO_SIGNO;
         denominador+= 2;
         numerador*=raiz*raiz;
   }
   pi_aprox= 6*serie;

   cout.precision(15);
   cout << "La aproximacion de PI por madhava es: " << pi_aprox << endl;
}
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: /* [Aproximación de PI por Wallis] Otra aproximación de PI introducida en el siglo XVII por
11: el matemático inglés John Wallis viene dada por:
12:
13:    pi/2 =2/1 • 2/3 • 4/3 • 4/5 • 6/5 • 6/7 ...
14:
15:    Construya un programa que lea el valor tope obligando a que esté entre 1 y cien mil,
16:    calcule la aproximación de PI mediante la anterior fórmula (multiplicando un total de
17:    tope fracciones) e imprima el resultado en pantalla.
18:    Debe resolver este problema de dos formas distintas, a saber:
19:       --Observe que el numerador y el denominador varían de forma alternativa (aunque
20:    ambos de la misma forma, a saltos de 2). Cuando a uno le toca cambiar, el otro
21:    permanece igual. Este comportamiento se puede controlar con una única variable
22:    de tipo de dato bool.
23:       --Otra forma de implementar los cambios en el numerador y denominador es observando
24:    que en cada iteración, el numerador es el denominador de la iteración
25:    anterior más 1 y el denominador es el numerador de la iteración anterior más 1.
26:    Ejemplo de entrada: 1000 -- Salida correcta: 3.1400238186006
27:    Ejemplo de entrada: 100000 -- Salida correcta: 3.14157694582286
28: */
29:
30: #include <iostream>
31: #include <cmath>
32:
33: using namespace std;
34:
35: int main(){
36:    int tope;
37:    int i = 0.0;
38:    double numerador = 0.0;
39:    double denominador = 1.0;
40:    double serie;
41:    double acumulador = 1.0;
42:    double pi_aprox;
43:    bool cambio;
44:
45:    do{
46:       cout << "Introduce el tope de calculo: ";
47:       cin >> tope;
48:    }
49:    while(tope<0 || tope>1e+5);
50:
51:    for(i=0;i<tope;i++){
52:       cambio = (i%2==0);
53:       if(cambio){
54:          numerador = numerador + 2;
55:          serie = numerador/denominador;
56:       }
57:       else{
58:          denominador = denominador + 2;
59:          serie = numerador/denominador;
60:       }
61:    acumulador = serie * acumulador;
62:
63:    }
64:
65:    pi_aprox = 2*acumulador;         //Es porque se iguala a pi/2 por lo que para obtener pi, se le multi
plica por 2
66:    cout.precision(15);
67:    cout << "Pi aproximado segun wallis es: " << pi_aprox << endl;
68:
69: }
70:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10:    /* [Aproximación de PI por Gregory-Leibniz] En el siglo XVII el matemático alemán Gottfried
11:    Leibniz y el matemático escocés James Gregory introdujeron una forma de calcular
12:    PI a través de una serie, es decir, de una suma de términos:
13:
14:    pi/4 = pow(-1,i)/(2*n +1)
15:
16:    Esta es una serie infinita, pues realiza la suma de infinitos términos. Como en Programación
17:    no podemos realizar un número infinito de operaciones, habrá que parar en
18:    un índice dado, llamémosle tope, obteniendo por tanto una aproximación al valor de
19:    PI. Usaremos el símbolo "aprox" para denotar esta aproximación:
20:
21:    pi/4 aprox pow(-1,n)/(2*n +1) ... + pow(-1,tope)/(2*tope +1)
22:
23:    Construya un programa que lea el valor tope obligando a que esté entre 1 y cien mil,
24:    calcule la aproximación de PI mediante la anterior serie e imprima el resultado en
25:    pantalla.
26:    Resuelva este problema de tres formas distintas (no hace falta que entregue tres ejercicios:
27:    baste con que incluya las dos primeras soluciones dentro de un comentario):
28:
29:    a) Use la función pow (potencia) de cmath para implementar (-1)^n
30:
31:    b) Para cada valor de n, calcule (-1)^n con un bucle, tal y como hizo en el ejercicio
32:    de la potencia (problema 18 [Factorial y Potencia] )
33:
34:    c) De una forma más eficiente que las anteriores. Por ejemplo, observe que el valor
35:    de (-1)^n es 1 para los valores pares de i y -1 para los impares
36:
37:    Recuerde que, para visualizar 15 cifras decimales, por ejemplo, debe incluir la sentencia
38:    cout.precision(15); antes de realizar la salida en pantalla.
39:
40:    Ejemplo de entrada: 1000 -- Salida correcta: 3.14259165433954
41:    Ejemplo de entrada: 100000 -- Salida correcta: 3.14160265348972
42:    */
43: #include <iostream>
44: #include <cmath>
45:
46: using namespace std;
47:
48: int main(){
49:
50:
51:    int tope;
52:    int n;
53:    double sumando, suma;
54:    double pi_aprox;
55:    do
56:    {
57:      cout << "Introduce el tope de calculo: ";
58:      cin >>tope;
59:    }
60:     while(tope<0 || tope>1e+5);
61:
62:    /* //Metodo (a):
63:    suma = 0;
64:    n = 0;
65:
66:    while(n<=tope){
67:
68:       sumando = pow(-1,n)/(2*n +1);
69:       // o directamente suma = suma + pow(-1, n) / (2*n + 1);
70:       suma = suma + sumando;
71:       n++;
72:    }
73:
74:    pi_aprox = 4 * suma;             //Es porque se iguala a pi/4 por lo que para obtener pi, se le multi
plica por 4
75:    cout.precision(15);
76:    cout << pi_aprox << endl;
77:
78:    return 0;
79:    */
80:
81:
82:    /* //METODO (B)
83:    suma = 0;
84:    int signo = 1;
85:    const int CAMBIO_SIGNO = -1;
```

```
   86:    for(n=0;n<tope;n++){
   87:       sumando = signo/(2.0*n +1);
   88:       suma = suma + sumando;
   89:       signo= signo * CAMBIO_SIGNO;
   90:    }
   91:    pi_aprox = 4*suma;                //Es porque se iguala a pi/4 por lo que para obtener pi, se le mult
iplica por 4
   92:    cout.precision(15);
   93:    cout << pi_aprox << endl;
   94:
   95:    return 0;
   96:    */
   97:    //METODO (c)
   98:    suma = 0;
   99:    int signo;
  100:
  101:    for(n=0;n<tope;n++){
  102:       if(n%2==0)
  103:          signo=1;
  104:       else
  105:          signo=-1;
  106:
  107:       sumando = signo/(2.0*n +1);
  108:       suma = suma + sumando;
  109:    }
  110:    pi_aprox = 4*suma;                //Es porque se iguala a pi/4 por lo que para obtener pi, se le mult
iplica por 4
  111:    cout.precision(15);
  112:    cout << pi_aprox << endl;
  113:
  114:    return 0;
  115: }
  116:
  117:
  118:
  119:
  120:
  121:
  122:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: /*
11: 24. [Mínimo de varios valores] Realice un programa que lea enteros desde teclado y calcule
12: cuántos se han introducido y cual es el mínimo de dichos valores (pueden ser
13: positivos o negativos). Se dejará de leer datos cuando el usuario introduzca el valor 0.
14: Realice la lectura de los enteros dentro de un bucle sobre una única variable llamada
15: dato. Es importante controlar los casos extremos, como por ejemplo, que el primer
16: valor leído fuese ya el terminador de entrada (en este caso, el cero).
17: Ejemplo de entrada: 0
18:    Salida correcta: Introducidos: 0. Mínimo: 0
19: Ejemplo de entrada: 1 3 -1 2 0
20:    Salida correcta: Introducidos: 4. Mínimo: -1
21: Ejemplo de entrada: 1 3 1 2 0
22:    Salida correcta: Introducidos: 4. Mínimo: 1
23: Una vez hecho el programa, indique qué cambios debería realizar si los valores a leer
24: son enteros negativos y el final de la entrada de datos lo marca la introducción de
25: cualquier valor positivo.
26: */
27: #include <iostream>
28: #include <cmath>
29:
30: using namespace std;
31:
32: int main(){
33:
34:    int dato, min_dato, i=-1;
35:    const int terminador = 0;
36:
37:    cout << "Introduce una serie de numeros: ";
38:    cin >> dato;
39:    i++;
40:    min_dato = dato;
41:    do{
42:       cout << " ";
43:       cin >> dato;
44:       if(dato < min_dato && dato != 0)
45:          min_dato = dato;
46:       i++;
47:    }
48:    while(dato != terminador);
49:
50:    cout << "---------------------------------------" << endl;
51:    cout << "\tEl numero mas pequenio es el " << min_dato << endl
52:         << "\t y hay " << i << " numeros" << endl;
53: }
54:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10:
11: /*[Mayor nota media] (Examen Prácticas Noviembre 2019)
12: Se quiere calcular la máxima nota media de evaluación continua de un conjunto de
13: alumnos. Para ello, se anota en un fichero un número entero con el código del alumno
14: y las notas que ha conseguido. El número de notas puede variar de un alumno a otro,
15: por lo que se terminará la introducción de las notas con un -1. La entrada de datos
16: finaliza con el código de alumno 0.
17: Cree un programa que lea las notas desde la entrada por defecto, y calcule el alumno
18: con mayor nota media. Puede suponer que los datos de entrada son siempre correctos.
19: Por ejemplo, para el siguiente registro de entradas, el alumno con máxima nota es el
20: que tiene identificador 17 con una nota media de 9.5
21:
22: 11    8 7 6 -1
23: 14    3 -1
24: 7     9 9 8 7 -1
25: 17    10 9 -1
26: 8     9 9 -1
27: 15    6 7 5 -1
28: 5     8 -1
29: 0
30:
31: */
32:
33: #include <iostream>
34: #include <cmath>
35:
36: using namespace std;
37:
38: int main(){
39:
40:    int identificador=1;
41:    int i;
42:    const int terminador_identificador=0;
43:    const int terminador_notas=-1;
44:    double notas;
45:    double acumulador;
46:    double media;
47:    double mejor_alumno=0;
48:    double mejor_nota=0;
49:
50:    cout << "Introduce el identificador del alumno: ";
51:    cin >> identificador;
52:
53:    while(identificador!= terminador_identificador){
54:       //inicializamos los datos
55:       acumulador=0.0;
56:       i=-1;
57:       notas=1.0;
58:
59:       cout << "Introduce las notas del alumno: ";
60:       cin >> notas;
61:
62:       while(notas!=terminador_notas){
63:          cin >> notas;
64:          acumulador+= notas;
65:          i++;
66:       }
67:
68:       media = acumulador/i;
69:       cout << identificador << "\t" << media << endl;
70:
71:       if(media>mejor_nota){
72:          mejor_nota=media;
73:          mejor_alumno=identificador;
74:       }
75:       cout << "Introduce el identificador del alumno: ";
76:       cin >> identificador;
77:    }
78:    cout << "---------------------------------------" << endl;
79:    cout << mejor_alumno << "\t" << mejor_nota << endl;
80: }
81:
82:
83:
84:
85:
```

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

/* 2. [Divisores] Recupere la solución del ejercicio 15 [Divisores de un entero] de la Relación
de Problemas II que puede encontrar en el siguiente enlace:
http://decsai.ugr.es/jccubero/FP/II_Divisores.cpp
Modifíquelo para separar los cómputos de las entradas y salidas de datos. Para ello,
se pide que cada vez que encuentre un divisor lo guarde en un vector divisores.
Una vez construido el vector, en un bucle aparte, debe imprimir sus componentes en
pantalla.
Ejemplo de entrada: 16 -- Salida correcta: 2, 4, 8
*/

#include <iostream>
using namespace std;

int main(){
   int entero, ultimo_divisor,divisor,util;
   const int LIMITE_DIVISORES=1000;
   int divisores[LIMITE_DIVISORES];

   cout << "Divisores de un entero\n\n";

//Entrada de datos...

   do{
      cout << "Introduce un numero entero mayor estricto que 0: ";
      cin >> entero;
   }while (entero <= 0);

//Cálculo de datos...

   ultimo_divisor = entero / 2;
   int i=0;

   for(divisor=2;divisor <= ultimo_divisor;divisor++){
      if(entero%divisor==0){
         divisores[i]= divisor;
         i++;
      }
   }
   util=i;

//Salida de datos...
   cout << "Los divisores son: ";
   for(int i=0;i<util;i++)
      cout << divisores[i] << " ";



}
```

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

#include <iostream>
using namespace std;

int main(){
   /*
   Esquema alumnos:

      Filtro para la distancia d (debe ser d >= 0):

      do{
         cin >> d;
      }while (d < 0);

      Filtro para el número de ptos n:

      do{
         cin >> n;
      }while (n < MIN  ||  MAX < n);
   */

   const int    LIM_KMS_RECARGO = 300;
   const double RECARGO_KM = 0.1;
   const double MIN_KMS_DSCTO = 700.0;
   const int    MIN_PTOS_DSCTO_BAJO = 100;
   const int    MIN_PTOS_DSCTO_ALTO = 200;
   const int    DSCTO_KMS = 2;
   const int    DSCTO_BAJO_PTOS = 3;
   const int    DSCTO_ALTO_PTOS = 4;
   const double TRF_BASE = 150.0;
   const int    MAX_NUM_PTOS = 400;
   const int    MIN_NUM_PTOS = 0;
   const char NUEVO_BILLETE='N';
   const char TERMINADOR='#';

   double dscto;
   double trf = 0;
   int dist;
   int ptos_fideliz;
   char billete;

   //Damos por hecho que minimo vas a comprar 1 billete

   cout << "Tarifa aerea."
        << "\nIntroduzca 'N' para un nuevo billete o '#' para finalizar la compra:" << endl;

   while(billete!=NUEVO_BILLETE && billete!=TERMINADOR)
      cin >> billete;

   while(billete!=TERMINADOR){

      cout << "\nIntroduzca la distancia del recorrido del viaje (> 0) y el "
           << "número de puntos de la tarjeta de fidelización (entre 0 y "
           << MAX_NUM_PTOS << ":\n";

      do{
         cin >> dist;
      }while (dist < 0);

      do{
         cin >> ptos_fideliz;
      }while (ptos_fideliz < MIN_NUM_PTOS
              ||
              MAX_NUM_PTOS < ptos_fideliz);


   /*
   Algoritmo:
      Inicializar la tarifa a la tarifa base

      Según sea la longitud del trayecto
         Actualizar la tarifa

      Según sea la longitud del trayecto
         Inicializar el dscto

      Según sea el número de puntos
         Actualizar el dscto
```

```
87:
88:       Aplicar el dscto calculado anteriormente a la tarifa
89:    */
90:
91:
92:       trf = TRF_BASE;
93:
94:       if (dist > LIM_KMS_RECARGO)
95:          trf = trf + RECARGO_KM*(dist - LIM_KMS_RECARGO) ;
96:
97:
98:       if (dist > MIN_KMS_DSCTO)
99:          dscto = DSCTO_KMS;
100:      else
101:         dscto = 0;
102:
103:      if (ptos_fideliz > MIN_PTOS_DSCTO_ALTO)
104:         dscto = dscto + DSCTO_ALTO_PTOS;
105:      else if (ptos_fideliz > MIN_PTOS_DSCTO_BAJO)
106:         dscto = dscto + DSCTO_BAJO_PTOS;
107:
108:      trf = trf * (1 - dscto / 100.0);
109:
110:      cout << "\n\nTarifa final aplicando los dsctos: ";
111:      cout << trf;
112:
113:      billete=0;
114:      cout <<"\n-----------------------------------------------------" << endl;
115:      cout << "\nIntroduzca 'N' para un nuevo billete o '#' para finalizar la compra:" << endl;
116:      while(billete!=NUEVO_BILLETE && billete!=TERMINADOR){
117:         cin >> billete;
118:      }
119:   }
120: }
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9: /*
10: 3. [Sustituir carácter por vector (con vector auxiliar)] Tenga en cuenta la observación al
11: inicio de esta relación de problemas sobre la lectura de los caracteres (ver página RPIII.1).
12: Para poder leer caracteres, incluyendo los espacios en blanco, hay que usar caracter = cin.get(),
13: en vez de cin >> caracter.
14:
15: Dado un vector de caracteres, queremos sustituir todas las apariciones de un carácter
16: y poner en su lugar el contenido de otro vector.
17:
18: Por ejemplo, si tenemos el vector [u n o a d o s a a], el resultado de
19: sustituir las apariciones del carácter 'a' por el nuevo vector [T T U] sería
20: [u n o T T U d o s T T U T T U]
21:
22: Resolveremos este problema de varias formas a lo largo de esta Relación de Problemas.
23: En este ejercicio, se construirá un tercer vector sustituido con el resultado
24: pedido.
25:
26: Construya un programa que lea caracteres hasta que se introduzca # lo que formará
27: el primer vector (v). A continuación lea el carácter a_borrar que se va a eliminar de
28: v. Finalmente, el programa leerá los caracteres que formarán el vector a_insertar
29: que sustituirán cada aparición de a_borrar. El terminador de entrada de caracteres
30: para el vector a_insertar es también el carácter #
31:
32: El programa construirá e imprimirá en pantalla un tercer vector sustituido que
33: contendrá los caracteres de v pero reemplazando todas las apariciones de a_borrar
34: por los caracteres del vector a_insertar. Si el vector a_insertar contuviese el
35: carácter a_borrar, dichas apariciones no se eliminan, tal y como puede apreciarse
36: en el último ejemplo que aparece al final de este enunciado.
37:
38: Para realizar la tarea pedida, se recomienda que implemente el siguiente algoritmo:
39:
40: Recorrer las componentes -i- del vector v
41:    Si v[i] == a_borrar
42:       Añadir a sustituido todas las componentes del vector a_insertar
43:    si no
44:       Añadir a sustituido la componente v[i]
45:
46: Puede utilizar el esbozo del programa disponible en el siguiente enlace:
47: http://decsai.ugr.es/jccubero/FP/III_SustituyeCaracterVectorEsbozo.cpp
48:
49: Ejemplo de entrada: u n o a d o s a a # T T U # a
50: -- Salida correcta: u n o T T U d o s T T U T T U
51: Ejemplo de entrada: u n o a d o s a a # T a U # a
52: -- Salida correcta: u n o T a U d o s T a U T a U
53: Finalidad: Trabajar con vectores auxiliares. Dificultad Baja.
54: */
55: #include <iostream>
56: #include <cmath>
57:
58: using namespace std;
59:
60: int main(){
61:    int k=0;
62:    int util_v = 0;
63:    int util_a_insertar = 0;
64:    int util_total;
65:    const int TAMANIO=100;
66:    char v[TAMANIO];
67:    char a_insertar[TAMANIO];
68:    char sustituido[TAMANIO];
69:    char caracter;
70:    char caracter_a_borrar;
71:    const char TERMINADOR = '#';
72:
73: //Entrada de Datos...
74:
75:    cout << "Introduce el vector inicial: ";
76:    caracter = cin.get();
77:
78:    while (caracter != TERMINADOR && util_v < TAMANIO){
79:       v[util_v] = caracter;
80:       util_v++;
81:       caracter = cin.get();
82:    }
83:    cout << "Introduce el caracter a borrar: ";
84:    cin >> caracter_a_borrar;
85:    caracter = cin.get();
86:
```

```
   caracter=0;   //o cualquier otra cosa diferente a TERMINADOR

   cout << "Introduce el vector a insertar: ";
   caracter = cin.get();

   while (caracter != TERMINADOR && util_a_insertar < TAMANIO){
      a_insertar[util_a_insertar] = caracter;
      util_a_insertar++;
      caracter = cin.get();
   }

//Cómputo...

   util_total=util_v;

   for(int i=0;i<util_total;i++){
      if(v[i]!=caracter_a_borrar){
         sustituido[k]=v[i];
         k++;
      }
      else{
         util_total+=util_a_insertar-1;
         for(int j=0;j<util_a_insertar;j++){
            sustituido[k]=a_insertar[j];
            k++;
         }
      }
   }
//Salida de Datos...

   for (int i = 0; i < util_total; i++)
      cout  << sustituido[i];




}
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11:
12: using namespace std;
13:
14: int main(){
15:    int min;
16:    int max;
17:    int k;
18:    int divisor;
19:    double tope;
20:
21:    divisor = 0;
22:    k = 0;
23:    max = 0;
24:
25: //Entrada de Datos...
26:
27:    cout << "Introduce un numero minimo: ";
28:    cin >> min;
29:
30:    cout << "Introduce un numero maximo: ";
31:    while(max<min)
32:       cin >> max;
33:
34:    cout <<"Introduce el limite de divisores: ";
35:    while(k<1)
36:       cin >> k;
37:
38: //Cómputo...
39:
40:    for(int i=min;i<=max;i++){
41:       tope=i/2;
42:       for(int j=2;j<tope;j++){
43:          if(i%j==0)
44:             divisor++;
45:       }
46:    if(divisor>=k)
47:       cout << i << "  ";
48:
49:    divisor=0;
50:    }
51:
52: }
53:
54:
```

```
    1: ////////////////////////////////////////////////////////////////////////
    2: //
    3: // Fundamentos de Programación
    4: // ETS Informática y Telecomunicaciones
    5: // Universidad de Granada
    6: // // Don Oreo
    7: //
    8: ////////////////////////////////////////////////////////////////////////
    9:
   10: #include <iostream>
   11: using namespace std;
   12:
   13: int main(){
   14:     int computo=0,util=0,i=0,j=0,solucion;
   15:     int pos=0; /*Inicializamos a 0 para comparar al final si no ha sido modificado e imprimir que "no ha
y solucion" */
   16:
   17:     //Introduciion de datos
   18:     cout << "Introduce el valor de util: ";
   19:     cin >> util;
   20:     int v[util];
   21:
   22:     cout << "Introduce el vector: ";
   23:     for (i=0;i<util;i++){
   24:         cin >> v[i];
   25:     }
   26:     i=0;
   27:
   28:     cout << "Inserte el solucion: ";
   29:     cin >> solucion;
   30:
   31:     //Cómputo
   32:     for (i=0;i<util;i++){
   33:         for (j=i;j<util;j++){
   34:             computo=computo+v[j];
   35:             if(computo == solucion){
   36:                 pos=i;
   37:             }
   38:         }
   39:         computo=0;
   40:     }
   41:    if(pos>0)
   42:       cout << pos+1 << endl; /*El mas 1 es para que la posicion a imprimir sea logica*/
   43:    else
   44:       cout << "No hay solucion" << endl;
   45: }
```

```cpp
////////////////////////////////////////////////////////////////////////
//
// Fundamentos de Programación
// ETS Informática y Telecomunicaciones
// Universidad de Granada
// Departamento de Ciencias de la Computación e Inteligencia Artificial
// Autor: Don Oreo
//
////////////////////////////////////////////////////////////////////////

// Frecuencias
 /*
   Algoritmo:

      Recorrer -i- el vector texto
         actual = texto[i]

         Si actual no está en el vector procesados:
            - Añadir actual a procesados
            - Contar el número de ocurrencias de actual
              en el vector texto -a partir de la posición i+1-
   */
#include <iostream>
using namespace std;

int main(){
   const char TERMINADOR = '@';
   const int NUM_CARACT_ASCII = 256;
   const int MAX_NUM_CARACT = 1e4;  // diez mil

   char a_buscar[NUM_CARACT_ASCII];
   int frecuencias[NUM_CARACT_ASCII] {};
   char texto[MAX_NUM_CARACT];
   int contador;
   char car;
   int util_a_buscar, util_texto;

   cout << "Frecuencias\n"
        << "Introduzca los caracteres del vector a buscar, con terminador "
        << TERMINADOR << "\n"
        << "A continuación introduzca  los caracteres del texto,"
        << " usando el mismo terminador.\n\n";

   // Introduccion de Datos

   car = cin.get();
   util_a_buscar = 0;

   while (car != TERMINADOR && util_a_buscar < NUM_CARACT_ASCII){
      a_buscar[util_a_buscar] = car;
      car = cin.get();
      util_a_buscar++;
   }

   car = cin.get();
   util_texto = 0;

   while (car != TERMINADOR && util_texto < MAX_NUM_CARACT){
      texto[util_texto] = car;
      car = cin.get();
      util_texto++;
   }


   //Computo de Datos

   /*//Metodo 1:
   for(int i=0;i<NUM_CARACT_ASCII;i++){
         car=i;
         for(int j=0;j<util_texto;j++)
            if(car==texto[j])
               frecuencias[i]++;
   }

   //Salida de Datos

   for(int j=0;j<util_a_buscar;j++)
      for(int i=0;i<NUM_CARACT_ASCII;i++)
         if(a_buscar[j]==i)
            cout << i << " : " << frecuencias[i] << endl;

   */
   //Metodo 2:

   for(int j=0;j<util_a_buscar;j++)
         for(int i=0;i<util_texto;i++)
```

```
87:             if(a_buscar[j]==texto[i])
88:                frecuencias[j]++;
89:
90:    //Salida de Datos
91:
92:    for(int j=0;j<util_a_buscar;j++)
93:       cout << j << " : " << frecuencias[j] << endl;
94:
95:    /*
96:    Ja@Juan Carlos Cubero@°
97:
98:    J: 1
99:    a: 2
100:   */
101: }
102:
103:
```

```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
//
////////////////////////////////////////////////////////////////////////

// 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];
        }

}
```

```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
//
////////////////////////////////////////////////////////////////////////

// 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:

```
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:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11:
12: using namespace std;
13:
14: int main(){
15:    const int MAX_PEDIDOS = 100;
16:    const int MAX_TECNICOS = 100;
17:    int tarifa[MAX_TECNICOS][MAX_PEDIDOS];
18:    bool A[MAX_TECNICOS][MAX_PEDIDOS];
19:    bool cogidos[MAX_PEDIDOS];
20:    int tecnicos,pedidos;
21:    int menor;
22:    int pos_menor = 100;      //Lo inicializamos a cualquier número que no se vaya a utilizar como posici
on en la matriz tarifa(como si pongo 123433124)
23:
24:    //Entrada de Datos
25:
26:    cout << "Introduce el numero de técnicos: ";
27:    cin >> tecnicos;
28:
29:    pedidos = tecnicos;
30:
31:    cout << "\nIntroduce la matriz tarifa: " << endl;
32:
33:    for(int i = 0; i < tecnicos; i++)
34:       for(int j = 0; j < pedidos; j++)
35:          cin >> tarifa[i][j];
36:
37:    //Cómputo
38:
39:    int precio_total = 0;
40:
41:    for(int i = 0; i < tecnicos; i++){
42:       menor = 100;
43:       pos_menor = 100;
44:       for(int j = 0; j < pedidos; j++){
45:          if(tarifa[i][j] < menor && cogidos[j] == false){
46:                menor = tarifa[i][j];
47:                pos_menor = j;
48:          }
49:       }
50:    cogidos[pos_menor] = true;
51:    A[i][pos_menor] = true;
52:    precio_total += menor;
53:    }
54:
55:
56:    //Salida de datos
57:
58:    /*Bool*/
59:    cout << "\n\n" << "Asignacion de pedidos:" << endl;
60:
61:    for(int i=0; i<tecnicos; i++){
62:       cout << "\n";
63:
64:    for(int j=0; j<pedidos; j++){
65:       cout << A[i][j] << '\t';
66:       }
67:    }
68:
69:    cout << endl << "------------------------------";
70:    /*Asignacion de pedidos*/
71:
72:    for(int i = 0; i < tecnicos; i++){
73:       for(int j = 0; j < pedidos; j++){
74:          if(A[i][j] == true)
75:          cout << "\nTécnico " << i << " --> Pedido " << j;
76:       }
77:         }
78:    cout <<endl << "------------------------------";
79:         cout << "\nEl coste total es de: " << precio_total << endl;
80:
81:
82:    return 0;
83: }
84:
85:
```

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


// Top K (versión eficiente)

#include <iostream>
using namespace std;

int main (){
        const int TERMINADOR = -1;
   int dato;
   const int TAMANIO = 1e+6;
   int vector[TAMANIO], topk[TAMANIO];
   int utilizados_vector, maximo, posicion_maximo, guardar, izda, i, k;

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

   //Entrada de Datos

   utilizados_vector = 0;
   cin >> dato;

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

   cin >> k;

   //computo

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

   for (izda = 0 ; izda < k ; izda++){
      maximo = topk[izda];
      for (i = izda + 1 ; i < utilizados_vector ; i++){
         if (topk[i] > maximo){
            maximo = topk[i];
            posicion_maximo = i;
         }
      }

      guardar = topk[izda];
      topk[izda] = maximo;
      topk[posicion_maximo] = guardar;
   }

   //Salida de Datos

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

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

#include <iostream>
using namespace std;

int main(){
   const int MIN_TEMP = -90,MAX_TEMP = 60;
   double anterior,actual;
   int secuencia = 1;            //Se va a repetir al menos 1 vez
   double mayor_secuencia = 0;
   double pos_secuencia = 1;
   int pos_mayor_secuencia = 1;
   int contador = 1;             //Empezamos desde 1 para que sea "lógico" y no empezar desde el 0 como los vectores
   bool final_entrada_datos=false;

   cin >> anterior;
   final_entrada_datos=anterior < MIN_TEMP
                       ||
                       anterior > MAX_TEMP;

   while(!final_entrada_datos){
      cin >> actual;
      if(actual > anterior){
         secuencia++;
      }
      else if(secuencia > mayor_secuencia){
         mayor_secuencia = secuencia;
         pos_mayor_secuencia = pos_secuencia;
         pos_secuencia = contador +1;      //Contador +1 ya que contamos apartir del siguiente
         secuencia = 1;
      }
      if(actual < MIN_TEMP || actual > MAX_TEMP){
         if(secuencia > mayor_secuencia){
            mayor_secuencia = secuencia -1;   // -1 ya que no contamos al "terminador"
            pos_mayor_secuencia = pos_secuencia;
         }
         final_entrada_datos=true;
      }
         anterior = actual;
         contador++;
   }


   cout << "Inicio: " << pos_mayor_secuencia << " Longitud: " << mayor_secuencia << endl;

   return 0;

}
```

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

#include <iostream>
#include <cmath>

using namespace std;

int Max2(int uno,int otro){
   if(uno > otro)
      return uno;
   else
      return otro;
}

int Max3(int uno,int otro,int otro_mas){
   int maximo = Max2(uno,otro);
   if( maximo > otro_mas)
      return maximo;
   else
      return otro_mas;
}

int main(){
   int entero1,entero2,entero3;
   int maximo;

   cin >> entero1 >> entero2 >> entero3;

   maximo = Max2(entero1,entero2);
   maximo = Max3(entero1,entero2,entero3);

   cout << maximo << endl;
}

```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9: /*
10: int ValorAbsoluto(int entero){
11:    if (entero < 0)
12:        entero = -entero;
13:    else
14:        return entero;
15: }
16: En esta primera función el error está en poner el
17: return entero dentro de un else ya que si el entero
18: es negativo, no va a devolver nada
19:
20: bool EsPositivo(int valor){
21:    if (valor > 0)
22:        return true;
23: }
24: Le estamos dando un valor int, nos debe devolver un
25: valor int, no un true/false como si fuera un boolean
26:
27: long ParteEntera(double real){
28:    int parte_entera;
29:
30:    parte_entera = trunc(real);
31:    return parte_entera;
32: }
33:
34: El int parte_entera es totalmente innecesario ya que es
35: justamente lo que hace el trunc. O quitas el trunc y haces
36: el casting automatico double a int o modificas el double real
37: */
```

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

// Elasticidad Precio-Demanda


#include <iostream>
#include <cmath>
using namespace std;

double Variacion_porcentual(double valor_ini,double valor_fin){
   double resultado;
   resultado = abs(100*((valor_ini - valor_fin)/valor_ini));
   return resultado;
}

double Elasticidad(double precio_ini,double precio_fin, double demanda_ini,double demanda_fin){
   double vp_demanda, vp_precio, elasticidad;
   vp_precio = Variacion_porcentual(precio_ini,precio_fin);
   vp_demanda = Variacion_porcentual(demanda_ini,demanda_fin);
   elasticidad = vp_demanda / vp_precio;
   return elasticidad;
}
////////////////////////////////////////////////////////////////


int main(){
   double precio_ini, precio_fin;
   double demanda_ini, demanda_fin;
   double elast_pd;

   cout << "Cálculo de la Elasticidad Precio-Demanda.\n\n"
        << "Introduzca cuaternas de valores:"
        << "El precio inicial, el precio final, la demanda inicial y la demanda final.\n"
        << "Introduzca cualquier negativo en el precio inicial para terminar\n\n";

   cin >> precio_ini;

   while (precio_ini >= 0){
      cin >> precio_fin;
      cin >> demanda_ini;
      cin >> demanda_fin;

      elast_pd = Elasticidad(precio_ini,precio_fin,demanda_ini,demanda_fin);

      cout << "Elasticidad Precio-Demanda: " << elast_pd << "\n";
      cin >> precio_ini;
   }
}
```

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

#include <iostream>
#include <cmath>
using namespace std;

double Redondeado(double numero,int cifra){
numero = round(numero * pow(10,cifra));
numero = numero / pow(10,cifra);
return numero;
}


int main(){
   double numero;
   int cifra;

   cin >> numero >> cifra;
   numero = Redondeado(numero,cifra);

   cout << "El numero redondeado es: "  << numero << endl;

   return 0;

}

```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10:
11: // Sustituir carácter por vector
12: #include <iostream>
13: using namespace std;
14:
15:
16: int main(){
17:    const char TERMINADOR = '#';
18:    const int MAX_NUM_CARACT = 3e6;  // Compile con la opción -Wl,--stack,21000000
19:    char v[MAX_NUM_CARACT],
20:         nuevo[MAX_NUM_CARACT];
21:    char a_borrar;
22:    char car;
23:    int i, util_v, util_nuevo;
24:
25:    // Lectura
26:
27:    car = cin.get();
28:    i = 0;
29:
30:    while (car != TERMINADOR){
31:       v[i] = car;
32:       car = cin.get();
33:       i++;
34:    }
35:
36:    util_v = i;
37:
38:    car = cin.get();
39:    i = 0;
40:
41:    while (car != TERMINADOR){
42:       nuevo[i] = car;
43:       car = cin.get();
44:       i++;
45:    }
46:
47:    util_nuevo = i-1;
48:
49:    a_borrar = cin.get();
50:
51:
52:    ////////////////////////////////////////////////////////
53:    int contador=0;
54:
55:    /*Contamos el numero de ocurrencias a borrar*/
56:    for(int i=0; i<util_v;i++){
57:       if(v[i]==a_borrar){
58:          contador++;
59:       }
60:    }
61:
62:    int util_resultado=util_v;
63:    util_resultado+=contador*util_nuevo;      //Obtenemos el util final
64:
65:    /*Inicializamos las variables de lectura/escritura*/
66:    int lec=util_v;
67:    int esc=util_resultado;
68:
69:    /** Comprueba si los caracteres de v coinciden con a_borrar
70:    *si coinciden, imprime el vector nuevo de derecha a izquierda
71:    *si no coincide, imprime el caracter de v a la izquierda del caracter de más a la derecha
72:    */
73:    //Nota:Los vectores leen de derecha a izquierda
74:
75:    while(lec>=0){
76:       if(v[lec]==a_borrar){
77:          for(int j=util_nuevo; j>=0; j--){
78:             v[esc]=nuevo[j];
79:             esc--;
80:          }
81:       }
82:       else{
83:          v[esc]=v[lec];
84:          esc--;
85:       }
86:       lec--;
```

```
87:     }
88:     ////////////////////////////////////////////////////////
89:
90:     cout << "<";
91:
92:     for (int i = 0; i < util_resultado; i++)
93:        cout << v[i] ;
94:
95:     cout << ">";
96:
97: // Ejemplo de entrada:
98: // unoadosaa#TTU#a
99:
100: // Salida:
101: // unoTTUdosTTUTTU
102: }
```

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

//DoubleToString

#include <iostream>
#include <cmath>
using namespace std;

string EliminaUltimo(string cadena){
   int ultimo;
   ultimo = cadena.size() - 1;
   while(cadena[ultimo] == '0' || cadena[ultimo] == '.'){
      cadena.erase(ultimo);
      ultimo = cadena.size() - 1;
   }
   return cadena;
}

double Redondeado(double real, int decimales){
   real = round(real * pow(10,decimales));
   real = real / pow(10,decimales);

   return real;
}
string DoubleToString(double real, int decimales){
   string cadena;
   real = Redondeado(real,decimales);
   cadena = to_string(real);
   cadena = EliminaUltimo(cadena);
   return cadena;
}

int main(){
   const char TERMINADOR = '#';
   double real;
   int decimales;
   string cadena;

   cout << "Convertir a string\n\n"
        << "Introduzca el real a convertir con las decimaless a redondear" << TERMINADOR << endl;

   cin >> real >> decimales;

   cadena = DoubleToString(real,decimales);

   cout << cadena << endl;
}

```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11: using namespace std;
12:
13: string LeeString(char terminador){
14:    string cadena;
15:    char car;
16:
17:    car = cin.get();
18:
19:    while (car != terminador){
20:       cadena.push_back(car);
21:       car = cin.get();
22:    }
23:
24:    return cadena;
25: }
26:
27: void ImprimeMarco (char car, int repeticiones){
28:    for(int i = 0; i < repeticiones; i++)
29:       cout << "*";
30:
31: }
32:
33: void Enmarca(string mensaje, char caracter){
34:    int util;
35:    string marco;
36:
37:    util = mensaje.size();
38:    ImprimeMarco(caracter,util);
39:    cout << "\n" << mensaje << "\n";
40:    ImprimeMarco(caracter,util);
41:
42: }
43: int main(){
44:    const char TERMINADOR = '@';
45:    char car;
46:    string msj;
47:
48:    // Lectura
49:
50:    cout << "Presentacion\n\n"
51:         << "Introduzca caracteres con terminador " << TERMINADOR << "\n";
52:
53:    msj = LeeString(TERMINADOR);
54:    Enmarca(msj, car);
55:
56: }
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: #include <iostream>
11: using namespace std;
12:
13: int LeeIntMayorIgual(int min) {
14:    int a_leer;
15:    do
16:       cin >> a_leer;
17:    while (min > a_leer);
18:
19:    return a_leer;
20: }
21:
22: int LeeIntRango(int min,int max){
23:    int dato = min -1;   //Para que siempre esté fuera del intervalo y entre en el bucle
24:
25:    while(min > dato || max < dato)
26:       cin >> dato;
27:
28:    return dato;
29:
30: }
31:
32: int main(){
33:
34:    long min, max, dato;
35:
36:    cout << "\nIntroduzca el valor mínimo y el máximo"
37:         << "\nA continuación introduzca enteros en el rango anterior\n";
38:
39:    cin >> min;
40:
41:    max = LeeIntMayorIgual(min);
42:
43:    dato = LeeIntRango(min,max);
44:
45:    cout << dato << " está en el intervalo [" << min << "," << max << "]" << endl;
46: }
```

```cpp
////////////////////////////////////////////////////////////////////////
//
// Fundamentos de Programación
// ETS Informática y Telecomunicaciones
// Universidad de Granada
// Departamento de Ciencias de la Computación e Inteligencia Artificial
// Autor: Don Oreo
//
////////////////////////////////////////////////////////////////////////

// Elimina Ultimos

#include <iostream>
#include <string>

using namespace std;

string LeeString(char terminador){
   string cadena;
   char caracter;

   caracter = cin.get();

   while (caracter != terminador){
      cadena.push_back(caracter);
      caracter = cin.get();
   }

   return cadena;
}

string EliminaUltimo(string cadena, char a_borrar){
   int ultimo;

   ultimo = cadena.size() - 1;
   while(cadena[ultimo] == a_borrar){
      cadena.erase(ultimo);
      ultimo = cadena.size() - 1;
   }
   return cadena;
}

/* Metodo 2
string EliminaUltimo(string cadena,char a_borrar){
   while( cadena.back() == a_borrar)
      cadena.pop_back();
   return cadena;
}
*/
int main(){
   const char TERMINADOR = '#';
   char a_borrar;
   string cadena;

   // Lectura

   cout << "Lee string\n\n"
        << "Introduzca caracteres con terminador " << TERMINADOR
        << ". Y despues escribe el caracter ultimo a borrar." << endl;

   cadena = LeeString(TERMINADOR);

   a_borrar = cin.get();

   cadena = EliminaUltimo(cadena,a_borrar);

   cout << cadena;
}
```

```
 1:
 2: #include <iostream>
 3: #include <string>
 4: using namespace std;
 5:
 6: /**
 7: * 3. [Errores en funciones void] Encuentre los errores, si los hubiese, en las siguientes
 8: * funciones void:
 9: **/
10:
11: /*
12: void EliminaUltimo(string cadena){
13: cadena.pop_back();
14: }
15: */
16: //Una funcion void no puede modificar una variable ya que no la puede devolver
17:
18: /*
19: void Imprime(double valor){
20: double valor;
21: cout << valor;
22: }
23: */
24: //Se redeclara la variable valor
25:
26: /*
27: void Cuadrado (int entero){
28: return entero*entero;
29: }
30: */
31: //Un void no puede devolver una variable mediante un return
32:
```

```cpp
////////////////////////////////////////////////////////////////////////
//
// Fundamentos de Programación
// ETS Informática y Telecomunicaciones
// Universidad de Granada
// Departamento de Ciencias de la Computación e Inteligencia Artificial
// Autor: Don Oreo
//
////////////////////////////////////////////////////////////////////////

// Mapa de distancias entre ciudades

#include <iostream>
#include <cmath>
using namespace std;

int main(){
   const int MAX_NUM_CIUDADES = 50;
   double mapa[MAX_NUM_CIUDADES][MAX_NUM_CIUDADES];
   int num_ciudades;
   const int CENTINELA = 0;

   cout << "Mapa de distancias"
        << "\n\nIntroduzca los datos en el siguiente orden:"
        << "\n- Número de ciudades"
        << "\n- Distancias entre ellas en forma de matriz diagonal superior"
        << "\n\n";

   cin >> num_ciudades;

   for (int i = 0; i < num_ciudades; i++)
      for (int j = 0; j < num_ciudades; j++)
         mapa[i][j] = 0;

   for (int i = 0; i < num_ciudades - 1; i++)
      for (int j = i+1; j < num_ciudades; j++){
         double dist;

         cin >> dist;
         mapa[i][j] = mapa[j][i] = dist;
      }

   // ----------------------------------------------------------------------

   int ciudad_mas_conectada;
   int max_conex = -1, num_conex;

   // COMPUTO DE DATOS

   for (int i = 0; i < num_ciudades; i++){
      num_conex = 0;
      for (int j = 0; j < num_ciudades; j++){
         if(mapa[i][j] != CENTINELA){
            num_conex++;
         }
      }
      if(num_conex > max_conex){
         max_conex = num_conex;
         ciudad_mas_conectada = i;
      }
   }


   cout << "\nCiudad más conectada: " << ciudad_mas_conectada
        << " con un total de " << max_conex
        << " conexiones";
}
/*
   5
   50  100   0    150
       70    0    0
             60   80
                  90
*/
/*
Salida:

Ciudad más conectada: 2 con un total de 4 conexiones
*/
```

```
    1: ////////////////////////////////////////////////////////////////////////
    2: //
    3: // Fundamentos de Programación
    4: // ETS Informática y Telecomunicaciones
    5: // Universidad de Granada
    6: // // Don Oreo
    7: //
    8: ////////////////////////////////////////////////////////////////////////
    9:
   10: //Interfaz Secuencia Caracteres
   11: /*
   12: Se quiere definir una clase SecuenciaCaracteres similar a la clase string para
   13: manipular secuencias de caracteres (de datos de tipo char).
   14: Esto es una secuencia de caracteres
   15: Como dato miembro privado, se recomienda usar un vector de caracteres. Ya sabemos
   16: que debemos reservar memoria suficiente. Supondremos que el máximo será 100. En
   17: vez de usar el literal 100, mejor usamos una constante. En ese caso, C++ obliga a que
   18: sea una constante estática (si aún no ha visto las constantes estáticas en clase de
   19: Teoría, no se preocupe ya que no se le pide que el programa compile). Nos quedaría
   20: lo siguiente:
   21:
   22: private:
   23:    static const int TAM = 100;
   24:    char caracteres[TAM];
   25: Se proponen dos alternativas para marcar el bloque del vector que se va a usar:
   26: \225 Usar un terminador fijo, por ejemplo, #, al final del bloque usado. En el ejemplo
   27: anterior, el vector caracteres contendría:
   28: 'E' 's' 't' 'o' ' ' 'e' 's' .... 'r 'e' 's' '#' ? ? .... ?
   29: \225 Usar un dato miembro privado utilizados que nos diga cuántos elementos
   30: estamos usando en cada momento. En el ejemplo anterior, utilizados valdría
   31: 35.
   32: ¿Qué opción le parece más adecuada? ¿Por qué?
   33: El mejor es el segundo ya que el util es más manejable que el TERMINADOR
   34: ¿Qué métodos definiría para manipular la secuencia? Al menos debe definir las cabeceras de los métodos p
ara realizar lo siguiente:
   35: a) Método Aniade para añadir un carácter al final de la secuencia.
   36: Tenga en cuenta que a una variable cadena de tipo string se le puede asignar
   37: directamente cadena = "Hola". Sin embargo, eso no lo sabemos hacer por
   38: ahora con objetos de nuestras propias clases. Por lo tanto, la única forma de
   39: añadir caracteres a un objeto de la clase SecuenciaCaracteres sería hacerlo
   40: de uno en uno. Así pues, llamando a un método Aniade, añadiríamos la 'H',
   41: luego la 'o' y así sucesivamente.
   42: b) Método Utilizados para obtener la longitud actual de la secuencia (el número
   43: de caracteres que contiene)
   44: c) Método Invierte para invertir la secuencia. En el ejemplo anterior, la secuencia
   45: se quedaría en:
   46: seretcarac ed aicneuces anu se otsE
   47: d) Método PrimeraOcurrencia para buscar la primera ocurrencia de un carácter.
   48: e) Método EliminaOcurrencias para eliminar las ocurrencias de un carácter.
   49: Por ejemplo, después de eliminar el carácter 'a', la secuencia quedaría así:
   50: Esto es un secuenci de crcteres
   51: */
   52: #include <iostream>
   53: using namespace std;
   54:
   55: class SecuenciaCaracteres{
   56: private:
   57:    static const int TAMANIO = 100;
   58:    char caracteres[TAMANIO];
   59:    int utilizados = 0;
   60: public:
   61:    void Aniade(char car){
   62:       caracteres[utilizados] = car;
   63:       utilizados++;
   64:    }
   65:
   66:    int Utilizados(){
   67:       return utilizados;
   68:    }
   69:
   70:    char Elemento(int posicion){
   71:       return caracteres[posicion];
   72:    }
   73:
   74:    void Invierte(){
   75:       char secuencia[TAMANIO];
   76:       int pos;
   77:       for(int i = 0; i <= utilizados; i++)
   78:          secuencia[i] = caracteres[i];
   79:
   80:       pos = 0;
   81:       for(int j = utilizados-1; j >= 0; j--){
   82:          caracteres[pos] = secuencia[j];
   83:          pos++;
   84:       }
   85:       cout << "\n";
```

```
 86:       for(int i = 0; i < utilizados; i++)
 87:          cout << caracteres[i];
 88:    }
 89:
 90:    int PrimeraOcurrencia(char car){
 91:       bool encontrado = 0;
 92:       int pos_ocurrencia = -1;
 93:       for(int i = 0; i < utilizados && ! encontrado; i++){
 94:          if(caracteres[i] == car){
 95:             pos_ocurrencia = i+1;
 96:             encontrado = true;
 97:          }
 98:       }
 99:    return pos_ocurrencia;
100:    }
101:    void EliminaOcurrencias(char a_borrar){
102:       char secuencia[TAMANIO];
103:       int pos;
104:
105:       pos = 0;
106:       for(int i = 0; i < utilizados; i++){
107:          if(caracteres[i] != a_borrar ){
108:             secuencia[pos] = caracteres[i];
109:             pos++;
110:          }
111:       }
112:       utilizados = pos;
113:       cout << "\n";
114:       for(int i = 0; i < utilizados; i++){
115:          caracteres[i] = secuencia[i];
116:          cout << caracteres[i];
117:       }
118:    }
119:
120:    void ImprimeSecuencia(){
121:       for(int i=0; i<utilizados; i++)
122:          cout << caracteres[i];
123:    }
124:
125: };
126:
127: int main(){
128:
129:    SecuenciaCaracteres Prueba;
130:
131:    Prueba.Aniade('H');
132:    Prueba.Aniade('o');
133:    Prueba.Aniade('l');
134:    Prueba.Aniade('a');
135:
136:    Prueba.ImprimeSecuencia();
137:
138:    Prueba.EliminaOcurrencias('H');
139:
140:    return 0;
141: }
```

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

//Cuadrado con constructor

#include <iostream>
#include <cmath>
using namespace std;

class Cuadrado{
private:
   double x,y;
   double lado;
public:
   Cuadrado(double coord_x, double coord_y, double longitud){
      if(longitud > 0){
         x = coord_x;
         y = coord_y;
         lado = longitud;
      }
      else{
         x = NAN;
         y = NAN;
         lado = -1;
      }
   }

   void SetCoordenadasLado(double coord_x, double coord_y, double longitud){
      x = coord_x;
      y = coord_y;
      lado = longitud;
   }

   double Coord_x(){
      return x;

   }
   double Coord_y(){
      return y;
   }

   double Lado(){
      return lado;
   }

   double Area(){
      return lado*lado;
   }

   double Perimetro(){
      return 4*lado;
   }

};

int main(){

   const string MSJ_COORDENADAS = "\nCoordenadas: ";
   const string MSJ_LONGITUD = "\nLongitud: ";
   const string MSJ_AREA = "\nÁrea: ";
   const string MSJ_PERIMETRO = "\nPerímetro: ";
   const string MSJ_PARCELA = "\nParcela ";
   double esquina_x1, esquina_y1, lado1,
       esquina_x2, esquina_y2, lado2;

   cout << "Introduce las coordenadas de la esquina inferior izquierda del cuadrado 1 y su lado: " << endl;
   cin >> esquina_x1
       >> esquina_y1
       >> lado1;

   Cuadrado una_parcela(esquina_x1, esquina_y1, lado1);

   cout << "Introduce las coordenadas de la esquina inferior izquierda del cuadrado 2 y su lado: " << endl;
   cin >> esquina_x2
       >> esquina_y2
       >> lado2;

   Cuadrado otra_parcela(esquina_x2, esquina_y2, lado2);
```

```
 85:
 86:    cout << MSJ_PARCELA << "1";
 87:    cout << MSJ_COORDENADAS << una_parcela.Coord_x() << " , " << una_parcela.Coord_y();
 88:    cout << MSJ_LONGITUD << una_parcela.Lado();
 89:    cout << MSJ_AREA << una_parcela.Area();
 90:    cout << MSJ_PERIMETRO << una_parcela.Perimetro() << endl;
 91:
 92:    cout << MSJ_PARCELA << "2";
 93:    cout << MSJ_COORDENADAS << otra_parcela.Coord_x() << " , " << otra_parcela.Coord_y();
 94:    cout << MSJ_LONGITUD << otra_parcela.Lado();
 95:    cout << MSJ_AREA << otra_parcela.Area();
 96:    cout << MSJ_PERIMETRO << otra_parcela.Perimetro() << endl;
 97:
 98:
 99:    return 0;
100: }
101: /*
102: 3.4  5.7  2.9
103: -5.6 -4.1 1.8
104: */
105: /*
106: Coordenadas: 3.4 , 5.7
107: Longitud: 2.9
108: Área: 8.41
109: Perímetro: 11.6
110:
111: Coordenadas: -5.6 -4.1
112: Longitud: 1.8
113: Área: 3.24
114: Perímetro: 7.2
115: */
116:
```

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

//Simulador Depósito

#include<iostream>
#include<cmath>
using namespace std;

class SimuladorDeposito{
private:
   double capital;
   double interes;
public:
   SimuladorDeposito()
   :capital(0),interes(0)
   {
   }
   void SetCapital(double capital_inicial){
      if(capital_inicial >= 0)
         capital = capital_inicial;
   }
   void SetInteres(double I){
      if(I >= 0)
         interes = I;
   }
   double Capital(){
      return capital;
   }
   int Interes(){
      return interes;
   }
   void CapitalFinal(int tope_anio){
      double elev;
      double int_compuesto;
      for(int anios = 0; anios < tope_anio;anios++){
         elev = pow((1 + interes/100),anios+1);
         int_compuesto = capital*elev;
         cout << "Capital obtenido transcurrido el anio numero " << anios
              << " = " << int_compuesto << endl;
      }

   }

   void DoblarCapital(){
      double tope_cap = 2*capital;
      int anios;

      while(capital<=tope_cap){
         capital = capital + capital*(interes/100);
         anios++;
      }

   cout << "\nPara doblar la cantidad inicial han de pasar " << anios << " anios" << endl;
   cout << "Al finalizar, se obtendra un total de " << capital << " euros" << endl;

   }

};

int main(){
   SimuladorDeposito sueldo1;

   double capital_inicial;
   int interes;
   int anios_a_invertir;

   cout << "Introduce el capital y el interes: ";
   cin >> capital_inicial >> interes;

   sueldo1.SetCapital(capital_inicial);
   sueldo1.SetInteres(interes);

   cout << "Introduce los años a invertir: ";
   cin >> anios_a_invertir;

   sueldo1.CapitalFinal(anios_a_invertir);
   sueldo1.DoblarCapital();
}
```

```cpp
////////////////////////////////////////////////////////////////////////
// Fundamentos de Programación
// ETS Informática y Telecomunicaciones
// Universidad de Granada
// Departamento de Ciencias de la Computación e Inteligencia Artificial
// Autor: Juan Carlos Cubero
////////////////////////////////////////////////////////////////////////

// Generador aleatorio de números enteros


#include <random>  // para la generación de números pseudoaleatorios
#include <chrono>  // para la semilla
#include <iostream>
using namespace std;


class GeneradorAleatorioEnteros{
private:
   mt19937 generador_mersenne;   // Mersenne twister
   uniform_int_distribution<int>  distribucion_uniforme;

   long long Nanosec(){
      return chrono::high_resolution_clock::now().time_since_epoch().count();
   }

public:
   GeneradorAleatorioEnteros()
      :GeneradorAleatorioEnteros(0, 1){
   }

   GeneradorAleatorioEnteros(int min, int max){
      const int A_DESCARTAR = 70000; // Panneton et al. ACM TOMS Volume 32 Issue 1, March 2006
      auto semilla = Nanosec();

      generador_mersenne.seed(semilla);
      generador_mersenne.discard(A_DESCARTAR);
      distribucion_uniforme = uniform_int_distribution<int> (min, max);
   }

   int Siguiente(){
      return distribucion_uniforme(generador_mersenne);
   }
};


int main(){
   int num_a_generar;
   GeneradorAleatorioEnteros aleat_num_a_generar(1,5);
   GeneradorAleatorioEnteros aleat_0_1;

   for (int i = 0; i < 4; i++){
   num_a_generar = aleat_num_a_generar.Siguiente();
   cout << "\n El numero aleatorio entre 1 y 5 es: " << num_a_generar << endl;
      for(int j = 0; j < num_a_generar; j++){
         cout << " " << aleat_0_1.Siguiente();
      }
   }

}
```

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

//Interfaz Simulador Depósito

#include<iostream>
#include<cmath>
using namespace std;

class SimuladorDeposito{
private:
   double capital;
   double interes;
public:
   SimuladorDeposito()
   :capital(0),interes(0)
   {
   }
   void SetCapital(double capital_inicial){
      if(esPositivo)
         ....
   }
   void SetInteres(double I){
      if(esPositivo)
         ....
   }
   double Capital(){
      return capital;
   }
   int Interes(){
      return interes;
   }
   void CapitalFinal(int tope_anio){
      ....
   }

   void DoblarCapital(){
      ....

   }

};

int main(){
   SimuladorDeposito sueldo1;

   double capital_inicial;
   int interes;
   int anios_a_invertir;

   cout << "Introduce el capital y el interes: ";
   cin >> capital_inicial >> interes;

   sueldo1.SetCapital(capital_inicial);
   sueldo1.SetInteres(interes);

   cout << "Introduce los años a invertir: ";
   cin >> anios_a_invertir;

   sueldo1.CapitalFinal(anios_a_invertir);
   sueldo1.DoblarCapital();
}
```

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

//Interfaz Instante

#include<iostream>
using namespace std;

class Instante{
private:
   int s = 0;
   int m = 0;
   int h = 0;
public:
   //opcion 1 introduciendo los segundos
   Instante(int seg){
      if(seg >=0){
         s=seg;
      }
      else
         s=-1;
   }
   //Opcion 2 introduciendo horas minutos segundos
   Instante(int hor, int min, int seg){
      if(seg >= 0 && min >= 0 && hor >= 0){
      s = seg;
      m = min;
      h = hor;
      }
      else{
         s = -1;
         m = -1;
         h = -1;
      }

   }

   void SetHorMinSeg(){
      while(s >= 60){
         m++;
         s-=60;
         while(m >= 60){
            h++;
            m-=60;
            while(h >= 24)
               h -= 24;
         }
      }
   }

   int S(){
      return s;
   }

   int M(){
      return m;
   }

   int H(){
      return h;
   }

   int SegTotales(){
      int s_total = 0;
      s_total = s + m*60 + h*3600;

      return s_total;
   }

   int Minutos(){
      int m_total = 0;
      m_total = m + h*60;

      return m_total;
   }

   void conversion_segundos(){
      while(s >= 60){
         m++;
         s-=60;
```

```
87:           while(m >= 60){
88:              h++;
89:              m-=60;
90:              while(h >= 24)
91:                 h = 0;
92:           }
93:
94:        }
95:     }
96: };
97:
98: int main(){
99:
100:
101:    int s,m,h;
102:    cout << "Introduce los segundos: ";
103:    cin >> s;
104:    Instante tiempo0(s);
105:    tiempo0.conversion_segundos();
106:    cout << "Tiempo 0: " << tiempo0.H() << "h " << tiempo0.M() << "min " << tiempo0.S() << "seg" << endl;
107:
108:    cout << "Introduzca la primera hora en formato(s m h): " << endl;
109:    cin >> h >> m >> s;
110:    Instante tiempo1(h,m,s);
111:    tiempo1.SetHorMinSeg();
112:
113:    cout << "Introduzca la segunda hora en formato(s m h): " << endl;
114:    cin >> h >> m >> s;
115:    Instante tiempo2(h,m,s);
116:    tiempo2.SetHorMinSeg();
117:
118:    cout << "\n\n";
119:    cout << "Tiempo 1: " << tiempo1.H() << "h " << tiempo1.M() << "min " << tiempo1.S() << "seg" << endl;
120:    cout << "Tiempo 2: " << tiempo2.H() << "h " << tiempo2.M() << "min " << tiempo2.S() << "seg" << endl;
121:
122:    cout   << "Segundos(t1): " << tiempo1.SegTotales() << " segundos."
123:           << "\nMinutos(t1): " << tiempo1.Minutos() << " minutos" << endl;
124:
125:    return 0;
126: }
127:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // // Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: //FormateadoDoubles
11:
12: #include<iostream>
13: using namespace std;
14:
15: double Redondea(double real, int num_decimales){
16: ...
17: }
18:
19: string EliminaUltimos(string cadena, char ultimo_car){
20: ...
21: }
22:
23: class FormateadorDoubles(){
24: private:
25:    string izda = "";
26:    string dcha = "";
27:    int decimales = 2;
28:    enum class SeparadorDecimal {PUNTO, COMA};
29:    SeparadorDecimal separador = SeparadorDecimal::COMA;
30: public:
31:    FormateadorDoubles()
32:    {
33:    }
34:    FormateadorDoubles(string delim_izda,delim_dcha)
35:    {
36:    }
37:    void Izda(){
38:       return izda;
39:    }
40:
41:    void Dcha(){
42:       return dcha;
43:    }
44:    int SetDecimales(int num_decimales){
45:       decimales = num_decimales;
46:    }
47:    void SetSeparadorPunto(){
48:       separador = SeparadorDecimal::PUNTO;
49:    }
50:    string GetCadena(double real){
51:       ... Redondea, EliminaUltimos, etc
52:    }
53:
54: };
55:
56: int main(){
57: double num;
58:
59: cout << "Introduce un numero: ";
60: cin >> num;
61: format_real(); //o format_real("<",">")
62:
63: format_real.GetCadena();
64:
65: }
66:
```

```
    1: ////////////////////////////////////////////////////////////////////////
    2: //
    3: // Fundamentos de Programación
    4: // ETS Informática y Telecomunicaciones
    5: // Universidad de Granada
    6: // // Don Oreo
    7: //
    8: ////////////////////////////////////////////////////////////////////////
    9:
   10: //Dinero Datos miembro Publicos
   11:
   12: #include <iostream>
   13: using namespace std;
   14:
   15: class Dinero{
   16: public:
   17:    int euros;
   18:    int centimos;
   19:
   20:    void SetEurCent(int eurs,int cents){
   21:       while(cents>=100){
   22:          eurs++;
   23:          cents-=100;
   24:       }
   25:       euros = eurs;
   26:       centimos = cents;
   27:    }
   28:
   29: } ;
   30:
   31: int main(){
   32:    Dinero un_dinero;
   33:    Dinero otro_dinero;
   34:    Dinero dinero_total;
   35:
   36:    int euros1,centimos1,
   37:        euros2,centimos2;
   38:    int eursuma,centsuma;
   39:    cout << "Introduce el primer dinero: " << endl;
   40:    cin >> euros1 >> centimos1;
   41:    cout << "Introduce el segundo dinero: " << endl;
   42:    cin >> euros2 >> centimos2;
   43:
   44:    un_dinero.SetEurCent(euros1,centimos1);
   45:    otro_dinero.SetEurCent(euros2,centimos2);
   46:
   47:    eursuma = un_dinero.euros + otro_dinero.euros;
   48:    centsuma = un_dinero.centimos + otro_dinero.centimos;
   49:
   50:    dinero_total.SetEurCent(eursuma,centsuma);
   51:
   52:    cout << "En el primer dinero hay " << un_dinero.euros << "," << un_dinero.centimos << " euros" << end
l;
   53:    cout << "En el segundo dinero hay " << otro_dinero.euros << "," << otro_dinero.centimos << " euros" <
< endl;
   54:    cout << "La suma del dinero es: " << dinero_total.euros << "," << dinero_total.centimos << " euros" <
< endl;
   55:
   56: }
```

```
    1: ////////////////////////////////////////////////////////////////////////
    2: //
    3: // Fundamentos de Programación
    4: // ETS Informática y Telecomunicaciones
    5: // Universidad de Granada
    6: // // Don Oreo
    7: //
    8: ////////////////////////////////////////////////////////////////////////
    9:
   10: //Dinero Datos miembro Privados
   11:
   12: #include <iostream>
   13: using namespace std;
   14:
   15: class Dinero{
   16: private:
   17:    int euros;
   18:    int centimos;
   19:
   20: public:
   21:    void SetEurCent(int eurs,int cents){
   22:       while(cents>=100){
   23:          eurs++;
   24:          cents-=100;
   25:       }
   26:       euros = eurs;
   27:       centimos = cents;
   28:    }
   29:
   30:    int Euros(){
   31:       return euros;
   32:    }
   33:    int Centimos(){
   34:       return centimos;
   35:    }
   36:
   37: } ;
   38:
   39: int main(){
   40:    Dinero un_dinero;
   41:    Dinero otro_dinero;
   42:    Dinero dinero_total;
   43:
   44:    int euros1,centimos1,
   45:        euros2,centimos2;
   46:    int eursuma,centsuma;
   47:    cout << "Introduce el primer dinero: " << endl;
   48:    cin >> euros1 >> centimos1;
   49:    cout << "Introduce el segundo dinero: " << endl;
   50:    cin >> euros2 >> centimos2;
   51:
   52:    un_dinero.SetEurCent(euros1,centimos1);
   53:    otro_dinero.SetEurCent(euros2,centimos2);
   54:
   55:    euros1 = un_dinero.Euros();
   56:    centimos1 = un_dinero.Centimos();
   57:    euros2 = otro_dinero.Euros();
   58:    centimos2 = otro_dinero.Centimos();
   59:
   60:    eursuma = euros1 + euros2;
   61:    centsuma = centimos1 + centimos2;
   62:
   63:    dinero_total.SetEurCent(eursuma,centsuma);
   64:
   65:    cout << "En el primer dinero hay " << un_dinero.Euros() << "," << un_dinero.Centimos() << " euros" <<
 endl;
   66:    cout << "En el segundo dinero hay " << otro_dinero.Euros() << "," << otro_dinero.Centimos() << " euro
s" << endl;
   67:    cout << "La suma del dinero es: " << dinero_total.Euros() << "," << dinero_total.Centimos() << " euro
s" << endl;
   68:
   69: }
```

```
    1: ////////////////////////////////////////////////////////////////////////
    2: //
    3: // Fundamentos de Programación
    4: // ETS Informática y Telecomunicaciones
    5: // Universidad de Granada
    6: // // Don Oreo
    7: //
    8: ////////////////////////////////////////////////////////////////////////
    9:
   10: //Cuadrado
   11:
   12: #include <iostream>
   13: using namespace std;
   14:
   15: class Cuadrado{
   16: private:
   17:    double x,y;
   18:    double lado;
   19: public:
   20:    void SetCoordenadasLado(double coord_x, double coord_y, double longitud){
   21:       x = coord_x;
   22:       y = coord_y;
   23:       lado = longitud;
   24:    }
   25:
   26:    double Coord_x(){
   27:       return x;
   28:
   29:    }
   30:    double Coord_y(){
   31:       return y;
   32:    }
   33:
   34:    double Lado(){
   35:       return lado;
   36:    }
   37:
   38:    double Area(){
   39:       return lado*lado;
   40:    }
   41:
   42:    double Perimetro(){
   43:       return 4*lado;
   44:    }
   45:
   46: };
   47:
   48: int main(){
   49:    Cuadrado una_parcela;
   50:    Cuadrado otra_parcela;
   51:
   52:    double esquina_x1, esquina_y1, lado1,
   53:        esquina_x2, esquina_y2, lado2;
   54:
   55:    cout << "Introduce las coordenadas de la esquina inferior izquierda del cuadrado 1 y su lado: " << en
dl;
   56:    cin >> esquina_x1 >> esquina_y1 >> lado1;
   57:
   58:    una_parcela.SetCoordenadasLado(esquina_x1, esquina_y1, lado1);
   59:
   60:    cout << "Introduce las coordenadas de la esquina inferior izquierda del cuadrado 2 y su lado: " << en
dl;
   61:    cin >> esquina_x2 >> esquina_y2 >> lado2;
   62:
   63:    otra_parcela.SetCoordenadasLado(esquina_x2, esquina_y2, lado2);
   64:
   65:    cout << "Coordenadas: " << una_parcela.Coord_x() << " , " << una_parcela.Coord_y() << endl;
   66:    cout << "Longitud: " << una_parcela.Lado() << endl;
   67:    cout << "Area: " << una_parcela.Area() << endl;
   68:    cout << "Perimetro " << una_parcela.Perimetro() << endl;
   69:    cout << "Coordenadas: " << otra_parcela.Coord_x() << " , " << una_parcela.Coord_y() << endl;
   70:    cout << "Longitud: " << otra_parcela.Lado() << endl;
   71:    cout << "Area: " << otra_parcela.Area() << endl;
   72:    cout << "Perimetro: " << otra_parcela.Perimetro() << endl;
   73:
   74:
   75:
   76: }
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // Autor: Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: // Secuencia de caracteres
11:
12: // IMPORTANTE:
13: // La clase SecuenciaCaracteres es "atípica" en el sentido de que es una clase
14: // con muchos métodos. El principio de responsabilidad única nos dice que
15: // las clases deben tener una única responsabilidad y, por tanto, no suelen tener
16: // un número elevado de métodos.
17: // Sin embargo, a veces nos encontramos con este tipo de clases "genéricas"
18: // En este caso, la responsabilidad es manejar una secuencia de caracteres
19: // lo que conlleva la definición de numerosos métodos
20: // La librería estándar STL contiene una clase (plantilla para ser más exactos)
21: // similar a esta clase: es la plantilla denominada "vector"
22:
23: #include <iostream>
24: #include <string>
25: using namespace std;
26:
27: class SecuenciaCaracteres{
28: private:
29:    static const int TAMANIO = 2e6;  // 2e6 es un real (dos millones)
30:                                     // -> casting automático a int
31:
32:                                     // Para poder dimensionar con un tamaño
33:                                     // tan grande, hay que cambiar unos parámetros
34:                                     // del compilador:
35:                                     // Herramientas -> Opciones del Compilador ->
36:                                     // Compilador -> Añadir las siguientes opciones
37:                                     // -Wl,--stack,26000000
38:    char v[TAMANIO];
39:    int util;
40:
41:    void IntercambiaComponentesDirectamente(int pos_izda, int pos_dcha){
42:       char intercambia;
43:
44:       intercambia = v[pos_izda];
45:       v[pos_izda] = v[pos_dcha];
46:       v[pos_dcha] = intercambia;
47:    }
48:
49:    bool EsCorrectaPosicion(int indice){
50:       return 0 <= indice && indice < util;
51:    }
52: public:
53:    SecuenciaCaracteres()
54:       :util(0) {
55:    }
56:
57:    int Utilizados(){
58:       return util;
59:    }
60:
61:    int Capacidad(){
62:       return TAMANIO;
63:    }
64:
65:    void EliminaTodos(){
66:       util = 0;
67:    }
68:
69:    void Aniade(char nuevo){
70:       if (util < TAMANIO){
71:          v[util] = nuevo;
72:          util++;
73:       }
74:    }
75:
76:    void Modifica(int posicion, char nuevo){
77:       if (EsCorrectaPosicion(posicion))
78:          v[posicion] = nuevo;
79:    }
80:
81:    char Elemento(int indice){
82:       return v[indice];
83:    }
84:
85:    string ToString(){
86:       // Si el número de caracteres en memoria es muy grande,
```

```
87:       // es mucho más eficiente reservar memoria previamente
88:       // y usar push_back
89:
90:       string cadena;
91:
92:       cadena.reserve(util);
93:
94:       for (int i=0; i < util; i++)
95:          cadena.push_back(v[i]);
96:          //cadena = cadena + v[i]  <- Evitarlo. Muy ineficiente para tamaños grandes;
97:
98:       return cadena;
99:    }
100:
101:   int PrimeraOcurrenciaEntre (int pos_izda, int pos_dcha, char buscado){
102:      int i = pos_izda;
103:      bool encontrado = false;
104:
105:      while (i <= pos_dcha  &&  !encontrado)
106:         if (v[i] == buscado)
107:            encontrado = true;
108:         else
109:            i++;
110:
111:      if (encontrado)
112:         return i;
113:      else
114:         return -1;
115:   }
116:
117:   int PrimeraOcurrencia (char buscado){
118:      return PrimeraOcurrenciaEntre (0, util - 1, buscado);
119:   }
120:
121:
122:   ////////////////////////////////////////////////////////////
123:   // Búsquedas
124:
125:   // Precond: 0 <= izda <= dcha < util
126:   int PosMinimoEntre(int izda, int dcha){
127:      int pos_minimo = -1;
128:      char minimo;
129:
130:      minimo = v[izda];
131:      pos_minimo = izda;
132:
133:      for (int i = izda+1 ; i <= dcha ; i++)
134:         if (v[i] < minimo){
135:            minimo = v[i];
136:            pos_minimo = i;
137:         }
138:
139:      return pos_minimo;
140:   }
141:
142:   int PosMinimo(){
143:      return PosMinimoEntre(0, util - 1);
144:   }
145:
146:   int BusquedaBinaria (char buscado){
147:      int izda, dcha, centro;
148:      bool encontrado = false;
149:
150:      izda = 0;
151:      dcha = util - 1;
152:      centro = (izda + dcha) / 2;
153:
154:      while (izda <= dcha  &&  !encontrado){
155:         if (v[centro] == buscado)
156:            encontrado = true;
157:         else if (buscado < v[centro])
158:            dcha = centro - 1;
159:         else
160:            izda = centro + 1;
161:
162:         centro = (izda + dcha) / 2;
163:      }
164:
165:      if (encontrado)
166:         return centro;
167:      else
168:         return -1;
169:   }
170:
171:
172:   ////////////////////////////////////////////////////////////
```

```
173:    // Recorridos que modifican las componentes
174:
175:    // Inserta un valor en la posición especificada
176:    void Inserta(int pos_insercion, char nuevo){
177:       if (util < TAMANIO  &&  pos_insercion >= 0
178:          &&  pos_insercion <= util){
179:
180:          for (int i = util ; i > pos_insercion ; i--)
181:             v[i] = v[i-1];
182:
183:          v[pos_insercion] = nuevo;
184:          util++;
185:       }
186:    }
187:
188:    /*
189:    Tipos de borrados:
190:       - Lógico
191:          Usar un valor de componente especial y marcar la componente con dicho valor
192:          Un vector de edades -> valor -1
193:          Un vector de caracteres alfabéticos -> '@'
194:          Ventajas: Muy rápido
195:
196:          Inconvenientes: Cualquier procesado posterior del vector
197:          debe tratar las componentes marcadas de una forma especial
198:
199:       - Físico
200:          Implica desplazar 1 posición a la izquierda, todas las componentes que hay a la derecha de
201:          la que queremos borrar.
202:
203:          Tiene justo las ventajas e incovenientes contrarias que el método anterior.
204:
205:        En esta versión, implementamos el borrado físico.
206:    */
207:
208:    // Elimina una componente, dada por su posición
209:    void Elimina (int posicion){
210:       /*
211:       Algoritmo:
212:
213:          Recorremos de izquierda a derecha toda las componentes
214:          que hay a la derecha de la posición a eliminar
215:             Le asignamos a cada componente la que hay a su derecha
216:       */
217:       if (posicion >= 0 && posicion < util){
218:          int tope = util-1;
219:
220:          for (int i = posicion ; i < tope ; i++)
221:             v[i] = v[i+1];
222:
223:          util--;
224:       }
225:
226:       // Nota:
227:
228:       // En vez de usar la asignación
229:       //    v[i] = v[i+1];
230:       // también podríamos haber puesto lo siguiente:
231:       //    Modifica(i, Elemento(i+1));
232:       // Hemos preferido acceder directamente a las componentes con la notación en corchete
233:       // para aumentar la eficiencia del método Elimina, ya que si el vector es muy grande
234:       // tendrá que realizar muchos desplazamientos y, por tanto, muchos accesos al método
235:       // Elemento. En general, desde dentro de la clase, los métodos de la clase Secuencia
236:       // accederán directamente a las componentes con la notación corchete
237:
238:       // Además, cuando entramos en la función Elimina, comprobamos con el condicional
239:       // que los accesos a los índices son correctos.
240:       // Si usamos el método Modifica, volveríamos a comprobar lo mismo.
241:
242:       // Nota:
243:
244:       // ¿Y si en vez de asignar v[i] = v[i+1];
245:       // llamamos a IntercambiaComponentesDirectamente(i, i+1) ?
246:       // La componente se eliminaría pero realizando el doble de asignaciones
247:       // Obviamente, no es necesario intercambiar las componentes.
248:       // Únicamente debemos ir asignando v[i] = v[i+1] de izquierda a derecha.
249:    }
250:
251:    void EliminaOcurrencias(char a_borrar){
252:       for(int i = util; i >= 0; i--)
253:          if(v[i] == a_borrar)
254:             Elimina(i);
255:
256:       cout << "El vector sin a_borrar: ";
257:       for(int i = 0; i < util; i++)
258:          cout << v[i];
```

```
259:    }
260:
261:    ////////////////////////////////////////////////////////////
262:    // Algoritmos de ordenación
263:
264:    void Ordena_por_Seleccion(){
265:       int pos_min;
266:
267:       for (int izda = 0 ; izda < util ; izda++){
268:          pos_min = PosMinimoEntre(izda, util - 1);
269:          IntercambiaComponentesDirectamente(izda, pos_min);
270:       }
271:    }
272:
273:    void Ordena_por_Insercion(){
274:       int izda, i;
275:       char a_desplazar;
276:
277:       for (izda=1; izda < util; izda++){
278:          a_desplazar = v[izda];
279:
280:          for (i=izda; i > 0 && a_desplazar < v[i-1]; i--)
281:             v[i] = v[i-1];
282:
283:          v[i] = a_desplazar;
284:       }
285:    }
286:
287:    void InsertaOrdenadamente(char nuevo){
288:         int i;
289:
290:         if (util > TAMANIO){
291:            for (i=util; i>0 && nuevo < v[i-1]; i--)
292:                 v[i] = v[i-1];
293:
294:            v[i] = nuevo;
295:            util++;
296:         }
297:    }
298:
299:
300:    void Ordena_por_Burbuja(){
301:       int izda, i;
302:
303:       for (izda = 0; izda < util; izda++)
304:         for (i = util-1 ; i > izda ; i--)
305:           if (v[i] < v[i-1])
306:              IntercambiaComponentesDirectamente(i, i-1);
307:    }
308:
309:    void Ordena_por_BurbujaMejorado(){
310:       int izda, i;
311:       bool cambio;
312:
313:       cambio= true;
314:
315:       for (izda=0; izda < util && cambio; izda++){
316:         cambio=false;
317:
318:         for (i=util-1 ; i>izda ; i--)
319:           if (v[i] < v[i-1]){
320:              IntercambiaComponentesDirectamente(i, i-1);
321:              cambio=true;
322:           }
323:       }
324:    }
325:
326:    void AniadeVarios(SecuenciaCaracteres nuevos){
327:       int totales_a_aniadir = nuevos.Utilizados();
328:
329:       for (int i = 0; i < totales_a_aniadir; i++)
330:          Aniade(nuevos.Elemento(i)); // Es importante entender
331:    }
332:
333:
334:    SecuenciaCaracteres ToUpper(){
335:       SecuenciaCaracteres en_mayuscula;
336:
337:       for(int i = 0; i < util; i++)
338:          en_mayuscula.Aniade(toupper(v[i]));
339:
340:       return en_mayuscula;
341:    }
342: };
343:
344: int main(){
```

```
345:     SecuenciaCaracteres cadena;
346:     char car,a_borrar;
347:
348:     car = cin.get();
349:     while(car != '#'){     //Pide caracteres hasta introducir '#'
350:        cadena.Aniade(car);
351:        car = cin.get();
352:     }
353:
354:     a_borrar = cin.get();
355:
356:     cadena.EliminaOcurrencias(a_borrar);
357:
358:     return 0;
359: }
360:
```

```
   1: ////////////////////////////////////////////////////////////////////////
   2: //
   3: // Fundamentos de Programación
   4: // ETS Informática y Telecomunicaciones
   5: // Universidad de Granada
   6: // Autor: Don Oreo
   7: //
   8: ////////////////////////////////////////////////////////////////////////
   9:
  10: // Secuencia de caracteres
  11:
  12: // IMPORTANTE:
  13: // La clase SecuenciaCaracteres es "atípica" en el sentido de que es una clase
  14: // con muchos métodos. El principio de responsabilidad única nos dice que
  15: // las clases deben tener una única responsabilidad y, por tanto, no suelen tener
  16: // un número elevado de métodos.
  17: // Sin embargo, a veces nos encontramos con este tipo de clases "genéricas"
  18: // En este caso, la responsabilidad es manejar una secuencia de caracteres
  19: // lo que conlleva la definición de numerosos métodos
  20: // La librería estándar STL contiene una clase (plantilla para ser más exactos)
  21: // similar a esta clase: es la plantilla denominada "vector"
  22: #include <iostream>
  23: #include <string>
  24: using namespace std;
  25:
  26: class SecuenciaCaracteres{
  27: private:
  28:    static const int TAMANIO = 2e6;  // 2e6 es un real (dos millones)
  29:                                     // -> casting automático a int
  30:
  31:                                     // Para poder dimensionar con un tamaño
  32:                                     // tan grande, hay que cambiar unos parámetros
  33:                                     // del compilador:
  34:                                     // Herramientas -> Opciones del Compilador ->
  35:                                     // Compilador -> Añadir las siguientes opciones
  36:                                     // -Wl,--stack,26000000
  37:    char v[TAMANIO];
  38:    int util;
  39:
  40:    void IntercambiaComponentesDirectamente(int pos_izda, int pos_dcha){
  41:       char intercambia;
  42:
  43:       intercambia = v[pos_izda];
  44:       v[pos_izda] = v[pos_dcha];
  45:       v[pos_dcha] = intercambia;
  46:    }
  47:
  48:    bool EsCorrectaPosicion(int indice){
  49:       return 0 <= indice && indice < util;
  50:    }
  51: public:
  52:    /*//Si lo ponemos publico, debemos comprobrar si los datos proporcionados son correctos(en private se
puede omitir)
  53:     void IntercambiaComponentesDirectamente(int pos_izda, int pos_dcha){
  54:       char intercambia;
  55:       if(EsCorrectaPosicion(pos_izda)
  56:          &&
  57:          EsCorrectaPosicion(pos_dcha)){
  58:
  59:          intercambia = v[pos_izda];
  60:          v[pos_izda] = v[pos_dcha];
  61:          v[pos_dcha] = intercambia;
  62:       }
  63:    }
  64:    */
  65:    SecuenciaCaracteres()
  66:       :util(0) {
  67:    }
  68:
  69:    int Utilizados(){
  70:       return util;
  71:    }
  72:
  73:    int Capacidad(){
  74:       return TAMANIO;
  75:    }
  76:
  77:    void EliminaTodos(){
  78:       util = 0;
  79:    }
  80:
  81:    void Aniade(char nuevo){
  82:       if (util < TAMANIO){
  83:          v[util] = nuevo;
  84:          util++;
  85:       }
```

```
    }

    void Modifica(int posicion, char nuevo){
       if (EsCorrectaPosicion(posicion))
          v[posicion] = nuevo;
    }

    char Elemento(int indice){
       return v[indice];
    }

    string ToString(){
       // Si el número de caracteres en memoria es muy grande,
       // es mucho más eficiente reservar memoria previamente
       // y usar push_back

       string cadena;

       cadena.reserve(util);

       for (int i=0; i < util; i++)
          cadena.push_back(v[i]);
          //cadena = cadena + v[i]  <- Evitarlo. Muy ineficiente para tamaños grandes;

       return cadena;
    }

    int PrimeraOcurrenciaEntre (int pos_izda, int pos_dcha, char buscado){
       int i = pos_izda;
       bool encontrado = false;

       while (i <= pos_dcha  &&  !encontrado)
          if (v[i] == buscado)
             encontrado = true;
          else
             i++;

       if (encontrado)
          return i;
       else
          return -1;
    }

    int PrimeraOcurrencia (char buscado){
       return PrimeraOcurrenciaEntre (0, util - 1, buscado);
    }


    ////////////////////////////////////////////////////////////
    // Búsquedas

    // Precond: 0 <= izda <= dcha < util
    int PosMinimoEntre(int izda, int dcha){
       int pos_minimo = -1;
       char minimo;

       minimo = v[izda];
       pos_minimo = izda;

       for (int i = izda+1 ; i <= dcha ; i++)
          if (v[i] < minimo){
             minimo = v[i];
             pos_minimo = i;
          }

       return pos_minimo;
    }

    int PosMinimo(){
       return PosMinimoEntre(0, util - 1);
    }

    int BusquedaBinaria (char buscado){
       int izda, dcha, centro;
       bool encontrado = false;

       izda = 0;
       dcha = util - 1;
       centro = (izda + dcha) / 2;

       while (izda <= dcha  &&  !encontrado){
          if (v[centro] == buscado)
             encontrado = true;
          else if (buscado < v[centro])
             dcha = centro - 1;
          else
```

```
172:             izda = centro + 1;
173:
174:          centro = (izda + dcha) / 2;
175:       }
176:
177:       if (encontrado)
178:          return centro;
179:       else
180:          return -1;
181:    }
182:
183:
184:    ////////////////////////////////////////////////////////
185:    // Recorridos que modifican las componentes
186:
187:    // Inserta un valor en la posición especificada
188:    void Inserta(int pos_insercion, char nuevo){
189:       if (util < TAMANIO  &&  pos_insercion >= 0
190:          &&  pos_insercion <= util){
191:
192:          for (int i = util ; i > pos_insercion ; i--)
193:             v[i] = v[i-1];
194:
195:          v[pos_insercion] = nuevo;
196:          util++;
197:       }
198:    }
199:
200:    /*
201:    Tipos de borrados:
202:       - Lógico
203:          Usar un valor de componente especial y marcar la componente con dicho valor
204:          Un vector de edades -> valor -1
205:          Un vector de caracteres alfabéticos -> '@'
206:          Ventajas: Muy rápido
207:
208:          Inconvenientes: Cualquier procesado posterior del vector
209:          debe tratar las componentes marcadas de una forma especial
210:
211:       - Físico
212:          Implica desplazar 1 posición a la izquierda, todas las componentes que hay a la derecha de
213:          la que queremos borrar.
214:
215:          Tiene justo las ventajas e incovenientes contrarias que el método anterior.
216:
217:        En esta versión, implementamos el borrado físico.
218:    */
219:
220:    // Elimina una componente, dada por su posición
221:    void Elimina (int posicion){
222:       /*
223:       Algoritmo:
224:
225:          Recorremos de izquierda a derecha toda las componentes
226:          que hay a la derecha de la posición a eliminar
227:             Le asignamos a cada componente la que hay a su derecha
228:       */
229:       if (posicion >= 0 && posicion < util){
230:          int tope = util-1;
231:
232:          for (int i = posicion ; i < tope ; i++)
233:             v[i] = v[i+1];
234:
235:          util--;
236:       }
237:
238:       // Nota:
239:
240:       // En vez de usar la asignación
241:       //    v[i] = v[i+1];
242:       // también podríamos haber puesto lo siguiente:
243:       //    Modifica(i, Elemento(i+1));
244:       // Hemos preferido acceder directamente a las componentes con la notación en corchete
245:       // para aumentar la eficiencia del método Elimina, ya que si el vector es muy grande
246:       // tendrá que realizar muchos desplazamientos y, por tanto, muchos accesos al método
247:       // Elemento. En general, desde dentro de la clase, los métodos de la clase Secuencia
248:       // accederán directamente a las componentes con la notación corchete
249:
250:       // Además, cuando entramos en la función Elimina, comprobamos con el condicional
251:       // que los accesos a los índices son correctos.
252:       // Si usamos el método Modifica, volveríamos a comprobar lo mismo.
253:
254:       // Nota:
255:
256:       // ¿Y si en vez de asignar v[i] = v[i+1];
257:       // llamamos a IntercambiaComponentesDirectamente(i, i+1) ?
```

```
258:       // La componente se eliminaría pero realizando el doble de asignaciones
259:       // Obviamente, no es necesario intercambiar las componentes.
260:       // Únicamente debemos ir asignando v[i] = v[i+1] de izquierda a derecha.
261:    }
262:
263:
264:    ////////////////////////////////////////////////////////
265:    // Algoritmos de ordenación
266:
267:    void Ordena_por_Seleccion(){
268:       int pos_min;
269:
270:       for (int izda = 0 ; izda < util ; izda++){
271:          pos_min = PosMinimoEntre(izda, util - 1);
272:          IntercambiaComponentesDirectamente(izda, pos_min);
273:       }
274:    }
275:
276:    void Ordena_por_Insercion(){
277:       int izda, i;
278:       char a_desplazar;
279:
280:       for (izda=1; izda < util; izda++){
281:          a_desplazar = v[izda];
282:
283:          for (i=izda; i > 0 && a_desplazar < v[i-1]; i--)
284:             v[i] = v[i-1];
285:
286:          v[i] = a_desplazar;
287:       }
288:    }
289:
290:    void InsertaOrdenadamente(char nuevo){
291:         int i;
292:
293:         if (util > TAMANIO){
294:            for (i=util; i>0 && nuevo < v[i-1]; i--)
295:                 v[i] = v[i-1];
296:
297:            v[i] = nuevo;
298:            util++;
299:         }
300:    }
301:
302:
303:    void Ordena_por_Burbuja(){
304:       int izda, i;
305:
306:       for (izda = 0; izda < util; izda++)
307:         for (i = util-1 ; i > izda ; i--)
308:           if (v[i] < v[i-1])
309:              IntercambiaComponentesDirectamente(i, i-1);
310:    }
311:
312:    void Ordena_por_BurbujaMejorado(){
313:       int izda, i;
314:       bool cambio;
315:
316:       cambio= true;
317:
318:       for (izda=0; izda < util && cambio; izda++){
319:         cambio=false;
320:
321:         for (i=util-1 ; i>izda ; i--)
322:           if (v[i] < v[i-1]){
323:              IntercambiaComponentesDirectamente(i, i-1);
324:              cambio=true;
325:           }
326:       }
327:    }
328:
329:    void AniadeVarios(SecuenciaCaracteres nuevos){
330:       int totales_a_aniadir = nuevos.Utilizados();
331:
332:       for (int i = 0; i < totales_a_aniadir; i++)
333:          Aniade(nuevos.Elemento(i)); // Es importante entender
334:    }
335:
336:
337:    SecuenciaCaracteres ToUpper(){
338:       SecuenciaCaracteres en_mayuscula;
339:
340:       for(int i = 0; i < util; i++)
341:          en_mayuscula.Aniade(toupper(v[i]));
342:
343:       return en_mayuscula;
```

```
344:    }
345:    bool EsPalindromo(){
346:       int izda = 0;
347:       int dcha = util - 1;
348:       bool es_palindromo = true;
349:       while(izda < dcha && es_palindromo){
350:          if(v[izda] != v[dcha])
351:             es_palindromo = false;
352:          izda++;
353:          dcha--;
354:       }
355:    return es_palindromo;
356:    }
357:
358:    void Invierte(){
359:       int izda = 0;
360:       int dcha = util - 1;
361:       bool es_palindromo;
362:
363:       es_palindromo = EsPalindromo();
364:
365:       while(izda < dcha && !es_palindromo){
366:          IntercambiaComponentesDirectamente(izda,dcha);
367:          izda++;
368:          dcha--;
369:       }
370:       cout << "El vector invertido es: ";
371:       for (int i = 0; i < util; i++)
372:          cout  << v[i];
373:    }
374:
375: };
376:
377:
378: int main(){
379:    SecuenciaCaracteres cadena;
380:    char car;
381:    bool es_palindromo;
382:
383:    car = cin.get();
384:    while(car != '\n'){      //Pide caracteres hasta introducir 'enter'
385:       cadena.Aniade(car);
386:       car = cin.get();
387:    }
388:
389:    es_palindromo = cadena.EsPalindromo();
390:
391:    if(es_palindromo)
392:       cout << "Es palindromo" << endl;
393:    else
394:       cout << "No es palindromo" << endl;
395:
396:    cadena.Invierte();
397: }
398:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // Autor: Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: // Secuencia de caracteres
11:
12: // IMPORTANTE:
13: // La clase SecuenciaCaracteres es "atípica" en el sentido de que es una clase
14: // con muchos métodos. El principio de responsabilidad única nos dice que
15: // las clases deben tener una única responsabilidad y, por tanto, no suelen tener
16: // un número elevado de métodos.
17: // Sin embargo, a veces nos encontramos con este tipo de clases "genéricas"
18: // En este caso, la responsabilidad es manejar una secuencia de caracteres
19: // lo que conlleva la definición de numerosos métodos
20: // La librería estándar STL contiene una clase (plantilla para ser más exactos)
21: // similar a esta clase: es la plantilla denominada "vector"
22:
23: #include <iostream>
24: #include <string>
25: using namespace std;
26:
27: class SecuenciaCaracteres{
28: private:
29:    static const int TAMANIO = 3e6;  // 2e6 es un real (dos millones)
30:                                     // -> casting automático a int
31:
32:                                     // Para poder dimensionar con un tamaño
33:                                     // tan grande, hay que cambiar unos parámetros
34:                                     // del compilador:
35:                                     // Herramientas -> Opciones del Compilador ->
36:                                     // Compilador -> Añadir las siguientes opciones
37:                                     // -Wl,--stack,26000000
38:    char v[TAMANIO];
39:    int util;
40:
41:    void IntercambiaComponentesDirectamente(int pos_izda, int pos_dcha){
42:       char intercambia;
43:
44:       intercambia = v[pos_izda];
45:       v[pos_izda] = v[pos_dcha];
46:       v[pos_dcha] = intercambia;
47:    }
48:
49:    bool EsCorrectaPosicion(int indice){
50:       return 0 <= indice && indice < util;
51:    }
52: public:
53:    SecuenciaCaracteres()
54:       :util(0) {
55:    }
56:
57:    int Utilizados(){
58:       return util;
59:    }
60:
61:    int Capacidad(){
62:       return TAMANIO;
63:    }
64:
65:    void EliminaTodos(){
66:       util = 0;
67:    }
68:
69:    void Aniade(char nuevo){
70:       if (util < TAMANIO){
71:          v[util] = nuevo;
72:          util++;
73:       }
74:    }
75:
76:    void Modifica(int posicion, char nuevo){
77:       if (EsCorrectaPosicion(posicion))
78:          v[posicion] = nuevo;
79:    }
80:
81:    char Elemento(int indice){
82:       return v[indice];
83:    }
84:
85:    string ToString(){
86:       // Si el número de caracteres en memoria es muy grande,
```

```
87:       // es mucho más eficiente reservar memoria previamente
88:       // y usar push_back
89:
90:       string cadena;
91:
92:       cadena.reserve(util);
93:
94:       for (int i=0; i < util; i++)
95:          cadena.push_back(v[i]);
96:          //cadena = cadena + v[i]  <- Evitarlo. Muy ineficiente para tamaños grandes;
97:
98:       return cadena;
99:    }
100:
101:   int PrimeraOcurrenciaEntre (int pos_izda, int pos_dcha, char buscado){
102:      int i = pos_izda;
103:      bool encontrado = false;
104:
105:      while (i <= pos_dcha  &&  !encontrado)
106:         if (v[i] == buscado)
107:            encontrado = true;
108:         else
109:            i++;
110:
111:      if (encontrado)
112:         return i;
113:      else
114:         return -1;
115:   }
116:
117:   int PrimeraOcurrencia (char buscado){
118:      return PrimeraOcurrenciaEntre (0, util - 1, buscado);
119:   }
120:
121:
122:   ////////////////////////////////////////////////////////////
123:   // Búsquedas
124:
125:   // Precond: 0 <= izda <= dcha < util
126:   int PosMinimoEntre(int izda, int dcha){
127:      int pos_minimo = -1;
128:      char minimo;
129:
130:      minimo = v[izda];
131:      pos_minimo = izda;
132:
133:      for (int i = izda+1 ; i <= dcha ; i++)
134:         if (v[i] < minimo){
135:            minimo = v[i];
136:            pos_minimo = i;
137:         }
138:
139:      return pos_minimo;
140:   }
141:
142:   int PosMinimo(){
143:      return PosMinimoEntre(0, util - 1);
144:   }
145:
146:   int BusquedaBinaria (char buscado){
147:      int izda, dcha, centro;
148:      bool encontrado = false;
149:
150:      izda = 0;
151:      dcha = util - 1;
152:      centro = (izda + dcha) / 2;
153:
154:      while (izda <= dcha  &&  !encontrado){
155:         if (v[centro] == buscado)
156:            encontrado = true;
157:         else if (buscado < v[centro])
158:            dcha = centro - 1;
159:         else
160:            izda = centro + 1;
161:
162:         centro = (izda + dcha) / 2;
163:      }
164:
165:      if (encontrado)
166:         return centro;
167:      else
168:         return -1;
169:   }
170:
171:
172:   ////////////////////////////////////////////////////////////
```

```
173:    // Recorridos que modifican las componentes
174:
175:    // Inserta un valor en la posición especificada
176:    void Inserta(int pos_insercion, char nuevo){
177:       if (util < TAMANIO  &&  pos_insercion >= 0
178:          &&  pos_insercion <= util){
179:
180:          for (int i = util ; i > pos_insercion ; i--)
181:             v[i] = v[i-1];
182:
183:          v[pos_insercion] = nuevo;
184:          util++;
185:       }
186:    }
187:
188:    /*
189:    Tipos de borrados:
190:       - Lógico
191:          Usar un valor de componente especial y marcar la componente con dicho valor
192:          Un vector de edades -> valor -1
193:          Un vector de caracteres alfabéticos -> '@'
194:          Ventajas: Muy rápido
195:
196:          Inconvenientes: Cualquier procesado posterior del vector
197:          debe tratar las componentes marcadas de una forma especial
198:
199:       - Físico
200:          Implica desplazar 1 posición a la izquierda, todas las componentes que hay a la derecha de
201:          la que queremos borrar.
202:
203:          Tiene justo las ventajas e incovenientes contrarias que el método anterior.
204:
205:        En esta versión, implementamos el borrado físico.
206:    */
207:
208:    // Elimina una componente, dada por su posición
209:    void Elimina (int posicion){
210:       /*
211:       Algoritmo:
212:
213:          Recorremos de izquierda a derecha toda las componentes
214:          que hay a la derecha de la posición a eliminar
215:             Le asignamos a cada componente la que hay a su derecha
216:       */
217:       if (posicion >= 0 && posicion < util){
218:          int tope = util-1;
219:
220:          for (int i = posicion ; i < tope ; i++)
221:             v[i] = v[i+1];
222:
223:          util--;
224:       }
225:    }
226:
227:    void EliminaOcurrencias(char a_borrar){
228:       int escr, lect;
229:       lect = escr = 0;
230:
231:       while (v[lect] != a_borrar){
232:          lect++;
233:          escr++;
234:       }
235:
236:       while (lect < util){
237:          if (v[lect] != a_borrar){
238:             v[escr] = v[lect];
239:             escr++;
240:          }
241:
242:          lect++;
243:       }
244:       util = escr;
245:
246:       cout << "El vector sin a_borrar: ";
247:       for(int i = 0; i < util; i++)
248:          cout << v[i];
249:    }
250:
251:    ////////////////////////////////////////////////////////
252:    // Algoritmos de ordenación
253:
254:    void Ordena_por_Seleccion(){
255:       int pos_min;
256:
257:       for (int izda = 0 ; izda < util ; izda++){
258:          pos_min = PosMinimoEntre(izda, util - 1);
```

```
259:          IntercambiaComponentesDirectamente(izda, pos_min);
260:       }
261:    }
262:
263:    void Ordena_por_Insercion(){
264:       int izda, i;
265:       char a_desplazar;
266:
267:       for (izda=1; izda < util; izda++){
268:          a_desplazar = v[izda];
269:
270:          for (i=izda; i > 0 && a_desplazar < v[i-1]; i--)
271:             v[i] = v[i-1];
272:
273:          v[i] = a_desplazar;
274:       }
275:    }
276:
277:    void InsertaOrdenadamente(char nuevo){
278:         int i;
279:
280:         if (util > TAMANIO){
281:            for (i=util; i>0 && nuevo < v[i-1]; i--)
282:                 v[i] = v[i-1];
283:
284:            v[i] = nuevo;
285:            util++;
286:         }
287:    }
288:
289:
290:    void Ordena_por_Burbuja(){
291:       int izda, i;
292:
293:       for (izda = 0; izda < util; izda++)
294:         for (i = util-1 ; i > izda ; i--)
295:           if (v[i] < v[i-1])
296:              IntercambiaComponentesDirectamente(i, i-1);
297:    }
298:
299:    void Ordena_por_BurbujaMejorado(){
300:       int izda, i;
301:       bool cambio;
302:
303:       cambio= true;
304:
305:       for (izda=0; izda < util && cambio; izda++){
306:         cambio=false;
307:
308:         for (i=util-1 ; i>izda ; i--)
309:           if (v[i] < v[i-1]){
310:              IntercambiaComponentesDirectamente(i, i-1);
311:              cambio=true;
312:           }
313:       }
314:    }
315:
316:    void AniadeVarios(SecuenciaCaracteres nuevos){
317:       int totales_a_aniadir = nuevos.Utilizados();
318:
319:       for (int i = 0; i < totales_a_aniadir; i++)
320:          Aniade(nuevos.Elemento(i)); // Es importante entender
321:    }
322:
323:
324:    SecuenciaCaracteres ToUpper(){
325:       SecuenciaCaracteres en_mayuscula;
326:
327:       for(int i = 0; i < util; i++)
328:          en_mayuscula.Aniade(toupper(v[i]));
329:
330:       return en_mayuscula;
331:    }
332: };
333:
334: int main(){
335:    SecuenciaCaracteres cadena;
336:    char car,a_borrar;
337:
338:    car = cin.get();
339:    while(car != '#'){     //Pide caracteres hasta introducir '#'
340:       cadena.Aniade(car);
341:       car = cin.get();
342:    }
343:
344:    a_borrar = cin.get();
```

```
345:
346:    cadena.EliminaOcurrencias(a_borrar);
347:
348:    return 0;
349: }
350:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // Autor: Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: // Secuencia de caracteres
11:
12: // IMPORTANTE:
13: // La clase SecuenciaCaracteres es "atípica" en el sentido de que es una clase
14: // con muchos métodos. El principio de responsabilidad única nos dice que
15: // las clases deben tener una única responsabilidad y, por tanto, no suelen tener
16: // un número elevado de métodos.
17: // Sin embargo, a veces nos encontramos con este tipo de clases "genéricas"
18: // En este caso, la responsabilidad es manejar una secuencia de caracteres
19: // lo que conlleva la definición de numerosos métodos
20: // La librería estándar STL contiene una clase (plantilla para ser más exactos)
21: // similar a esta clase: es la plantilla denominada "vector"
22:
23: #include <iostream>
24: #include <string>
25: using namespace std;
26:
27: class SecuenciaCaracteres{
28: private:
29:    static const int TAMANIO = 2e6;  // 2e6 es un real (dos millones)
30:                                     // -> casting automático a int
31:
32:                                     // Para poder dimensionar con un tamaño
33:                                     // tan grande, hay que cambiar unos parámetros
34:                                     // del compilador:
35:                                     // Herramientas -> Opciones del Compilador ->
36:                                     // Compilador -> Añadir las siguientes opciones
37:                                     // -Wl,--stack,26000000
38:    char v[TAMANIO];
39:    int util;
40:
41:    void IntercambiaComponentesDirectamente(int pos_izda, int pos_dcha){
42:       char intercambia;
43:
44:       intercambia = v[pos_izda];
45:       v[pos_izda] = v[pos_dcha];
46:       v[pos_dcha] = intercambia;
47:    }
48:
49:    bool EsCorrectaPosicion(int indice){
50:       return 0 <= indice && indice < util;
51:    }
52: public:
53:    SecuenciaCaracteres()
54:       :util(0) {
55:    }
56:
57:    int Utilizados(){
58:       return util;
59:    }
60:
61:    int Capacidad(){
62:       return TAMANIO;
63:    }
64:
65:    void EliminaTodos(){
66:       util = 0;
67:    }
68:
69:    void Aniade(char nuevo){
70:       if (util < TAMANIO){
71:          v[util] = nuevo;
72:          util++;
73:       }
74:    }
75:
76:    void Modifica(int posicion, char nuevo){
77:       if (EsCorrectaPosicion(posicion))
78:          v[posicion] = nuevo;
79:    }
80:
81:    char Elemento(int indice){
82:       return v[indice];
83:    }
84:
85:    string ToString(){
86:       // Si el número de caracteres en memoria es muy grande,
```

```
87:       // es mucho más eficiente reservar memoria previamente
88:       // y usar push_back
89:
90:       string cadena;
91:
92:       cadena.reserve(util);
93:
94:       for (int i=0; i < util; i++)
95:          cadena.push_back(v[i]);
96:          //cadena = cadena + v[i]  <- Evitarlo. Muy ineficiente para tamaños grandes;
97:
98:       return cadena;
99:    }
100:
101:   int PrimeraOcurrenciaEntre (int pos_izda, int pos_dcha, char buscado){
102:      int i = pos_izda;
103:      bool encontrado = false;
104:
105:      while (i <= pos_dcha  &&  !encontrado)
106:         if (v[i] == buscado)
107:            encontrado = true;
108:         else
109:            i++;
110:
111:      if (encontrado)
112:         return i;
113:      else
114:         return -1;
115:   }
116:
117:   int PrimeraOcurrencia (char buscado){
118:      return PrimeraOcurrenciaEntre (0, util - 1, buscado);
119:   }
120:
121:
122:   ////////////////////////////////////////////////////////////
123:   // Búsquedas
124:
125:   // Precond: 0 <= izda <= dcha < util
126:   int PosMinimoEntre(int izda, int dcha){
127:      int pos_minimo = -1;
128:      char minimo;
129:
130:      minimo = v[izda];
131:      pos_minimo = izda;
132:
133:      for (int i = izda+1 ; i <= dcha ; i++)
134:         if (v[i] < minimo){
135:            minimo = v[i];
136:            pos_minimo = i;
137:         }
138:
139:      return pos_minimo;
140:   }
141:
142:   int PosMinimo(){
143:      return PosMinimoEntre(0, util - 1);
144:   }
145:
146:   int BusquedaBinaria (char buscado){
147:      int izda, dcha, centro;
148:      bool encontrado = false;
149:
150:      izda = 0;
151:      dcha = util - 1;
152:      centro = (izda + dcha) / 2;
153:
154:      while (izda <= dcha  &&  !encontrado){
155:         if (v[centro] == buscado)
156:            encontrado = true;
157:         else if (buscado < v[centro])
158:            dcha = centro - 1;
159:         else
160:            izda = centro + 1;
161:
162:         centro = (izda + dcha) / 2;
163:      }
164:
165:      if (encontrado)
166:         return centro;
167:      else
168:         return -1;
169:   }
170:
171:
172:   ////////////////////////////////////////////////////////////
```

```
173:    // Recorridos que modifican las componentes
174:
175:    // Inserta un valor en la posición especificada
176:    void Inserta(int pos_insercion, char nuevo){
177:       if (util < TAMANIO  &&  pos_insercion >= 0
178:          &&  pos_insercion <= util){
179:
180:          for (int i = util ; i > pos_insercion ; i--)
181:             v[i] = v[i-1];
182:
183:          v[pos_insercion] = nuevo;
184:          util++;
185:       }
186:    }
187:
188:    /*
189:    Tipos de borrados:
190:       - Lógico
191:          Usar un valor de componente especial y marcar la componente con dicho valor
192:          Un vector de edades -> valor -1
193:          Un vector de caracteres alfabéticos -> '@'
194:          Ventajas: Muy rápido
195:
196:          Inconvenientes: Cualquier procesado posterior del vector
197:          debe tratar las componentes marcadas de una forma especial
198:
199:       - Físico
200:          Implica desplazar 1 posición a la izquierda, todas las componentes que hay a la derecha de
201:          la que queremos borrar.
202:
203:          Tiene justo las ventajas e incovenientes contrarias que el método anterior.
204:
205:        En esta versión, implementamos el borrado físico.
206:    */
207:
208:    // Elimina una componente, dada por su posición
209:    void Elimina (int posicion){
210:       /*
211:       Algoritmo:
212:
213:          Recorremos de izquierda a derecha toda las componentes
214:          que hay a la derecha de la posición a eliminar
215:             Le asignamos a cada componente la que hay a su derecha
216:       */
217:       if (posicion >= 0 && posicion < util){
218:          int tope = util-1;
219:
220:          for (int i = posicion ; i < tope ; i++)
221:             v[i] = v[i+1];
222:
223:          util--;
224:       }
225:    }
226:
227:    // Algoritmos de ordenación
228:
229:    void Ordena_por_Seleccion(){
230:       int pos_min;
231:
232:       for (int izda = 0 ; izda < util ; izda++){
233:          pos_min = PosMinimoEntre(izda, util - 1);
234:          IntercambiaComponentesDirectamente(izda, pos_min);
235:       }
236:    }
237:
238:    void Ordena_por_Insercion(){
239:       int izda, i;
240:       char a_desplazar;
241:
242:       for (izda=1; izda < util; izda++){
243:          a_desplazar = v[izda];
244:
245:          for (i=izda; i > 0 && a_desplazar < v[i-1]; i--)
246:             v[i] = v[i-1];
247:
248:          v[i] = a_desplazar;
249:       }
250:    }
251:
252:    void InsertaOrdenadamente(char nuevo){
253:         int i;
254:
255:         if (util > TAMANIO){
256:            for (i=util; i>0 && nuevo < v[i-1]; i--)
257:                 v[i] = v[i-1];
258:
```

```
259:            v[i] = nuevo;
260:            util++;
261:         }
262:    }
263:
264:
265:    void Ordena_por_Burbuja(){
266:       int izda, i;
267:
268:       for (izda = 0; izda < util; izda++)
269:         for (i = util-1 ; i > izda ; i--)
270:           if (v[i] < v[i-1])
271:              IntercambiaComponentesDirectamente(i, i-1);
272:    }
273:
274:    void Ordena_por_BurbujaMejorado(){
275:       int izda, i;
276:       bool cambio;
277:
278:       cambio= true;
279:
280:       for (izda=0; izda < util && cambio; izda++){
281:         cambio=false;
282:
283:         for (i=util-1 ; i>izda ; i--)
284:           if (v[i] < v[i-1]){
285:              IntercambiaComponentesDirectamente(i, i-1);
286:              cambio=true;
287:           }
288:       }
289:    }
290:
291:    void AniadeVarios(SecuenciaCaracteres nuevos){
292:       int totales_a_aniadir = nuevos.Utilizados();
293:
294:       for (int i = 0; i < totales_a_aniadir; i++)
295:          Aniade(nuevos.Elemento(i)); // Es importante entender
296:    }
297:
298:
299:    SecuenciaCaracteres ToUpper(){
300:       SecuenciaCaracteres en_mayuscula;
301:
302:       for(int i = 0; i < util; i++)
303:          en_mayuscula.Aniade(toupper(v[i]));
304:
305:       return en_mayuscula;
306:    }
307:    bool EsVocal(int indice){
308:    /*
309:    Nos vale en privado y en publico ya que el objeto puede/debe contener
310:    el metodo tanto para acceder desde el main como para calculos dentro
311:    de otros metodos privados.
312:    */
313:    }
314:
315:    bool EsVocal(char caracter){
316:    /*
317:    No tiene sentido meterlo en la clase como método publico porque llegaremos
318:    a conclusiones absurdas como bool EsVocal('e'). ¿Es false porque no existe
319:    en cadena = "Adios" pero es true porque 'e' es una vocal?. No tiene sentido
320:    Sin embargo, si nos vale como privado para hacer calculos o como funcion
321:    global para otras cadenas o vectores.
322:    */
323:    }
324: };
325:
326:
327: int main(){
328:    SecuenciaCaracteres cadena;
329:    int pos_car;
330:    char car;
331:    bool es_vocal = false;
332:
333:    car = cin.get();
334:    while(car != '\n'){     //Pide caracteres hasta introducir 'enter'
335:       cadena.Aniade(car);
336:       car = cin.get();
337:    }
338:    cout << "Introduce la posicion del caracter que quieres comprobar si es vocal o no: ";
339:    cin >> pos_car;
340:
341:    cadena.EsVocal(pos_car);
342:
343: }
```

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

// Túnel

#include <iostream>
#include <cmath>
#include "IV_FormateadorDoubles.cpp"
#include "IV_Instante.cpp"

using namespace std;


/*
Túnel en un momento dado:

matr:     ["4733MTI"  ,  "5232LTL"  , "3330PRB" ,   ?  , ... ,  ?  ]
seg_ent:  [    13     ,     79      ,    160    ,   ?  , ... ,  ?  ]
seg_sal:  [    96     ,    NULO     ,    NULO   , NULO , ... , NULO]

tot = 3
*/

class Tunel{
private:
   double distancia_km;

   static const int MAX_NUM_VEHICULOS = 100;
   string matriculas[MAX_NUM_VEHICULOS];
   int entradas[MAX_NUM_VEHICULOS];
   int salidas[MAX_NUM_VEHICULOS];
   int total_entradas = 0;

   static const int INSTANTE_NULO = -1;

public:
   Tunel(double long_tunel)
   :distancia_km(long_tunel)
   {
      for(int i = 0; i < MAX_NUM_VEHICULOS; i++)
         salidas[i] = -1;
   }

   string Matriculas(int posicion){
      return matriculas[posicion];
   }

   int Entradas(int posicion){
      return entradas[posicion];
   }

   int Salidas(int posicion){
      return salidas[posicion];
   }

   double Longitud(){
      return distancia_km;
   }

   int TotalEntradas(){
      return total_entradas;
   }

   void Entra(string matricula, int hora, int minuto, int segundo){
      int pos = -1;
      Instante inst_entrada(hora,minuto,segundo);
      int seg_totales = inst_entrada.SegundosTotales();

      for(int i = 0; i < total_entradas; i++)
         if(matricula == matriculas[i])
            pos = i;
      if(pos == -1){
         matriculas[total_entradas] = {matricula};
         entradas[total_entradas] = seg_totales;
         total_entradas++;
      }
      else{
         entradas[pos] = seg_totales;
         salidas[pos] = INSTANTE_NULO;
      }
```

```
87:     }
88:
89:     void Sale(string matricula, int hora, int minuto, int segundo){
90:        int pos = -1;
91:        Instante inst_salida(hora,minuto,segundo);
92:        int seg_totales = inst_salida.SegundosTotales();
93:
94:        for(int i = 0; i < total_entradas; i++)
95:           if(matricula == matriculas[i])
96:              pos = i;
97:
98:        if(pos != -1)
99:           salidas[pos] = seg_totales;
100:
101:    }
102:
103:    bool HaSalido(int pos){
104:       bool ha_salido = true;
105:
106:       if(Salidas(pos) == -1)
107:          ha_salido = false;
108:
109:       return ha_salido;
110:    }
111:
112:    double Velocidad(int pos){
113:       double velocidad;
114:       double diferencia_hora = (Salidas(pos) - Entradas(pos))/3600.0;
115:
116:       velocidad = Redondea(distancia_km/diferencia_hora,1);
117:
118:       return velocidad;
119:    }
120:
121: };
122:
123:
124: int main(){
125:    const char FIN_ENTRADA = '#';
126:    const char ENTRADA = 'E';
127:    const char SALIDA  = 'S';
128:    char acceso;
129:    bool error_lectura;
130:
131:    string matricula;
132:    string cadena;
133:    double long_tunel;
134:    int hora, min, seg;
135:
136:    cin >> long_tunel;
137:    Tunel tunel(long_tunel);
138:
139:    //Escribe E para entrada, S para salida y # para terminar la lectura
140:    cin >> acceso;
141:    error_lectura = false;
142:
143:    while (acceso != FIN_ENTRADA && !error_lectura){
144:       cin >> matricula;
145:       cin >> hora >> min >> seg;
146:
147:       if (acceso == ENTRADA)
148:          tunel.Entra(matricula,hora,min,seg);
149:       else if (acceso == SALIDA)
150:          tunel.Sale(matricula,hora,min,seg);
151:       else
152:          error_lectura = true;
153:
154:       cin >> acceso;
155:    }
156:
157:    //-------------------------------------------------------------
158:
159:    FormateadorDoubles format_veloc("," km/h", 1);
160:    if (error_lectura)
161:       cout << "\nSe produjo un error en la lectura. " << endl;
162:    else{
163:       int total_entradas = tunel.TotalEntradas();
164:       for (int i = 0; i < total_entradas; i++){
165:          cadena += "\n\nMatricula:\t" + tunel.Matriculas(i) +
166:          "\nVelocidad:\t";
167:
168:          if(!tunel.HaSalido(i))
169:             cadena += "No ha salido";
170:          else
171:             cadena += format_veloc.GetCadena(tunel.Velocidad(i));
172:       }
```

```
173:    }
174:
175:    cout << cadena << endl;
176: }
177:
178:    // longitud_túnel  <entrada_o_salida  Matrícula# Instante> ... #
179:
180: // Entrada:
181: /*
182: 3.4
183: E 4733MTI 0 0 13
184: E 5232LTL 0 1 19
185: S 4733MTI 0 1 36
186: E 3330PRB 0 2 40
187: S 5232LTL 0 3 25
188: #
189: */
190:
191: // Salida:
192: /*
193: Matrícula:      4733MTI
194: Velocidad:      147.5 km/h
195:
196: Matrícula:      5232LTL
197: Velocidad:      97.1 km/h
198:
199: Matrícula:      3330PRB
200: Velocidad:      No ha salido
201: */
202:
203:
204: ////////////////////////////////////////////////////////////////
205:
206: // Entrada:
207: /*
208:    3.4
209: E 4733MTI 0 0 13
210: E 1976KEX 0 0 34
211: E 7717UQS 0 0 47
212: E 4744SEU 0 0 56
213: E 5232LTL 0 1 19
214: S 4733MTI 0 1 36
215: E 6188MOH 0 1 36
216: E 6603JHQ 0 2 4
217: E 6898DVW 0 2 17
218: E 3330PRB 0 2 40
219: S 1976KEX 0 2 53
220: E 1758HRV 0 2 56
221: E 8210YVI 0 3 9
222: S 5232LTL 0 3 25
223: S 6603JHQ 0 3 25
224: S 7717UQS 0 3 29
225: S 6188MOH 0 3 29
226: E 9265JJA 0 3 35
227: S 4744SEU 0 3 40
228: E 4864DUN 0 3 49
229: S 3330PRB 0 3 51
230: E 1071VVF 0 3 54
231: S 1758HRV 0 4 30
232: E 5917FBY 0 4 43
233:    */
234:
235:    // Salida:
236:    /*
237:    Matrícula:   4733MTI
238: Velocidad:      147.5 km/h
239:
240: Matrícula:      1976KEX
241: Velocidad:      88.1 km/h
242:
243: Matrícula:      7717UQS
244: Velocidad:      75.6 km/h
245:
246: Matrícula:      4744SEU
247: Velocidad:      74.6 km/h
248:
249: Matrícula:      5232LTL
250: Velocidad:      97.1 km/h
251:
252: Matrícula:      6188MOH
253: Velocidad:      108.3 km/h
254:
255: Matrícula:      6603JHQ
256: Velocidad:      151.1 km/h
257:
258: Matrícula:      6898DVW
```

```
259: Velocidad:      No ha salido
260:
261: Matrícula:      3330PRB
262: Velocidad:      172.4 km/h
263:
264: Matrícula:      1758HRV
265: Velocidad:      130.2 km/h
266:
267: Matrícula:      8210YVI
268: Velocidad:      No ha salido
269:
270: Matrícula:      9265JJA
271: Velocidad:      No ha salido
272:
273: Matrícula:      4864DUN
274: Velocidad:      No ha salido
275:
276: Matrícula:      1071VVF
277: Velocidad:      No ha salido
278:
279: Matrícula:      5917FBY
280: Velocidad:      No ha salido
281:    */
282:
```

```
1: ////////////////////////////////////////////////////////////////////////
2: //
3: // Fundamentos de Programación
4: // ETS Informática y Telecomunicaciones
5: // Universidad de Granada
6: // Autor: Don Oreo
7: //
8: ////////////////////////////////////////////////////////////////////////
9:
10: // Mapa de distancias entre ciudades
11:
12: #include <iostream>
13: #include <cmath>
14: using namespace std;
15:
16:
17:
18: // ------------------------------------------------------------------------
19: class MapaDistancias{
20: private:
21:     static const int NUM_MAX_CIUDADES = 50;
22:     static const int DISTANCIA_NULA = 0;
23:     double mat_dist[NUM_MAX_CIUDADES][NUM_MAX_CIUDADES];
24:     int  num_ciudades;
25:
26:     bool EsCorrectaDistancia(double distancia){
27:        return distancia > DISTANCIA_NULA;
28:     }
29:
30:     bool EsCorrectoIndice(int indice){
31:       return 0 <= indice && indice < num_ciudades;
32:     }
33:
34:     bool EsCorrectaPosicion(int origen, int destino){
35:        return  EsCorrectoIndice(origen) && EsCorrectoIndice(destino);
36:     }
37:
38: public:
39:    MapaDistancias (int numero_ciudades)
40:       : num_ciudades(numero_ciudades)
41:    {
42:          for(int i = 0; i < NUM_MAX_CIUDADES; i++){
43:             for( int j = 0; j < NUM_MAX_CIUDADES; j++)
44:                mat_dist[i][j] = DISTANCIA_NULA;
45:          }
46:    }
47:
48:    int Capacidad(){
49:       return NUM_MAX_CIUDADES;
50:    }
51:
52:    int NumCiudades(){
53:       return num_ciudades;
54:    }
55:    double DistanciaCiudad(int pos_i, int pos_j){
56:       return mat_dist[pos_i][pos_j];
57:    }
58:
59:    void ModificaDistancia(int una, int otra, double distancia){
60:       if (EsCorrectaDistancia(distancia) && EsCorrectaPosicion(una, otra)){
61:          mat_dist[una][otra] = mat_dist[otra][una] = distancia;
62:       }
63:    }
64:
65:    int CiudadMejorConectada(){
66:       int indice_mas_conectada = -1;
67:       int max_conex = -1, num_conex;
68:
69:       for (int origen = 0; origen < num_ciudades; origen++){
70:          num_conex = 0;
71:
72:          for (int destino = 0; destino < num_ciudades; destino++)
73:             if (mat_dist[origen][destino] != 0)
74:                num_conex++;
75:
76:          if (num_conex > max_conex){
77:             max_conex = num_conex;
78:             indice_mas_conectada = origen;
79:          }
80:       }
81:
82:       return indice_mas_conectada;
83:    }
84:
85:    int MejorEscalaEntre (int origen, int destino){
86:       int escala_de_min_distancia = -1;
```

```
87:       double distancia_con_escala;
88:       double min_distancia = INFINITY;
89:
90:       for (int escala = 0; escala < num_ciudades; escala++){
91:          distancia_con_escala = 0;
92:
93:          if (mat_dist[origen][escala] != 0 && mat_dist[escala][destino] != 0)
94:             distancia_con_escala = mat_dist[origen][escala] +
95:                                    mat_dist[escala][destino];
96:
97:          if (distancia_con_escala != 0){
98:             if (distancia_con_escala < min_distancia){
99:                escala_de_min_distancia = escala;
100:               min_distancia = distancia_con_escala;
101:            }
102:         }
103:      }
104:
105:      return escala_de_min_distancia;
106:   }
107:
108:   string ToString() {
109:      string cadena;
110:
111:      for (int i = 0; i < num_ciudades; i++){
112:         for (int j = 0; j < num_ciudades; j++){
113:            cadena.append(to_string(mat_dist[i][j]));
114:            cadena.append("\t");
115:         }
116:         cadena.append("\n");
117:      }
118:      return cadena;
119:   }
120: };
121:
122:
123:
124: // ------------------------------------------------------------------------
125: // ------------------------------------------------------------------------
126: int main (){
127:    int ciudad_mas_conectada;
128:    int origen, destino, escala;
129:    int num_ciudades;
130:    const int TERMINADOR_CIUDADES = -1;
131:
132:    cout << "Mapa de distancias"
133:         << "\n\nIntroduzca los datos en el siguiente orden:"
134:         << "\na) Número de ciudades"
135:         << "\nb) Distancias entre ellas en forma de matriz diagonal superior"
136:         << "\n   Lista de índices de ciudades para las que se quiere ver "
137:         << "\n   si están todas conectadas entre sí. Terminador: "
138:         << TERMINADOR_CIUDADES
139:         << "\n"
140:         << "\n   Ciudad de origen y ciudad de destino."
141:         << "\n\n";
142:
143:    cin >> num_ciudades;
144:
145:    MapaDistancias mapa(num_ciudades);
146:
147:    for (int i = 0; i < num_ciudades - 1; i++){
148:       for (int j = i+1; j < num_ciudades; j++){
149:          double distancia;
150:
151:          cin >> distancia;
152:          mapa.ModificaDistancia(i, j, distancia);
153:       }
154:    }
155:
156:    /*for(int i = 0; i < num_ciudades; i++){
157:       cout << "\n";
158:       for( int j = 0; j < num_ciudades; j++)
159:          cout << mapa.DistanciaCiudad(i,j) << "\t";
160:    }*/
161:
162:    cin >> origen >> destino;
163:
164:    //-------------------------------------------------------------
165:    // Ciudad mejor conectada (con mayor número de conexiones directas)
166:    ciudad_mas_conectada = mapa.CiudadMejorConectada();
167:
168:    cout << "La ciudad con más conexiones directas es la ciudad: "
169:         << ciudad_mas_conectada;
170:
171:
172:    //-------------------------------------------------------------
```

```
173:    // Mejor escala entre origen y destino
174:    escala = mapa.MejorEscalaEntre(origen,destino);
175:
176:    if (escala == -1)
177:       cout << "No existe escala" << endl;
178:    else
179:       cout << "\nLa mejor escala entre " << origen << " y "
180:            << destino << " es " << escala << endl;
181:    return 0;
182: }
183:
184: /*
185: Entrada:
186:
187:    5
188:    50  100   0    150
189:        70    0    0
190:              60   80
191:                   90
192:    0  4
193:
194: Salida:
195:
196:    La ciudad con más conexiones directas es la ciudad: 2
197:    La mejor escala entre 0 y 4 es 2
198: */
```