UNIVERSIDAD DE MÁLAGA
DPTO. DE LENGUAJES Y CC. DE LA COMPUTACIÓN
E.T.S. DE INGENIERÍA INFORMÁTICA

Práctica Nº 3.
Estructuras de Control (Iteración)

Ejercicios de clase

1.- Programa que vaya pidiendo por teclado las notas (nºs reales) de los alumnos de una clase, hasta que se introduzca un valor negativo (el cual ya no sería una nota, y solo servirá para indicar el final de la secuencia de notas), y que podría ser incluso el primero (indicando que no hay notas). Si se introdujesen algunos valores (supuestas notas) mayores que 10.0, no se deberán considerar para los cálculos, sino que habrá que obviarlas. Comience planteando sólo la estructura iterativa que permite leer todas las notas (use un bucle tipo **while**, ya que no sabemos cuántas notas hay que leer, y podría ser que no hubiese ninguna que analizar, si el 1º valor fuese negativo). Después añada lo necesario para calcular el nº de **aprobados** y el nº de **suspensos**. Recuerde que si alguna nota es incorrecta (>10.0) **NO** se deberá considerar para nada (pero tampoco deberá terminar la lectura de notas, simplemente, obviar ese valor > 10). Modifique el programa para que muestre también la **nota media** de la clase. Complételo para que calcule la **mejor nota** y la **peor nota** de la clase. Por último, ofrezca al usuario la posibilidad de repetir el proceso cuantas veces quiera (cálculos para otras clases distintas), para lo cual deberá responder con una **'s'** o una **'S'** a la pregunta **"¿Desea repetir el proceso (S/N)?: "** que el programa debe hacerle.

```
Introduce las notas (separadas por blancos, finalice con un valor negativo
y pulse INTRO): -2.0
No has introducido ninguna nota
¿Desea repetir el proceso (S/N)? S

Introduce las notas (separadas por blancos, finalice con un valor negativo
y pulse INTRO):
6     22.3      2      5.5      6.5     30.4    -1.0
Aprobados=3,  Suspensos= 1 ,  Nota media=5,  Mejor=6.5,  Peor= 2
¿Desea repetir el proceso (S/N)?   S

Introduce las notas (separadas por blancos, finalice con un valor negativo
y pulse INTRO):
12.45     3.5      20    -1.0
Suspensos=1,  Nota media= 3.5,  Mejor=3.5,  Peor=3.5
¿Desea repetir el proceso (S/N)?   N
```

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

int main()
{

    float    nota, suma, mejor, peor;
    unsigned suspensos, aprobados, total;
    char repetir;

    do
    {
        suspensos = aprobados = 0;
        suma = 0.;
        mejor = 0.;
        peor = 10.;

        cout << "INTRO UNA NOTA (NEGATIVA PARA TERMINAR): ";
        cin >> nota; // leemos la 'supuesta' primera nota,
        // OJO, ESTE 1º VALOR PUEDE SER MAYOR QUE 10
        // -> NO DEBERÍAMOS CONSIDERARLO COMO LA 1ª MEJOR NOTA,
        // YA QUE ENTONCES NINGUNA NOTA VÁLIDA (Y POR TANTO, <=10
        // )SUPERARÁ ESE VALOR

        while (nota >= 0.0) // mientras no sea negativa:
        {
            if ( nota <= 10.0 )// si la NOTA es correcta (ENTRE 0 Y 10),
                               // la procesamos:
            {
                suma = suma + nota;  // actualizamos la suma de notas,
                               //sumando tb la  actual (si es correcta)

                       if ( nota >= 5 )
                {
                    aprobados++;
                }
                else
                {
                    suspensos++;
                }

                if ( nota > mejor )
                {
                    mejor = nota;
                }

                if ( nota < peor )
                {
                    peor = nota;
                }
            }
            cout << "INTRO OTRA NOTA (NEGATIVA PARA TERMINAR): ";
            cin >> nota;    // leemos siguiente nota
        }

        total = aprobados + suspensos;
        if ( total > 0 ) // si hay notas válidas, escribimos resultados
        {
```

```
            cout << "\n APROBADOS=" << aprobados << "\n SUSPENSOS=" <<
            suspensos;
            cout << "\n NOTA MEDIA=" << suma / total;
            cout << "\n MEJOR NOTA=" << mejor << "\n PEOR NOTA=" << peor
<< endl;
        }
        else
        {
            cout << "NO HAS INTRODUCIDO NINGUNA NOTA CORRECTA " << endl;
        }
        cout << "REPETIR (S/N): ";
        cin >> repetir;
    }
    while ( repetir=='s' || repetir=='S' );
    return 0;

}
```

2.- Programa que pida un nº natural por teclado y calcule y muestre la cantidad de dígitos que tiene. **PISTA**: Si un número entero lo divides entre 10, el cociente será ese mismo nº, pero sin la última cifra:

```
98125 /  10 = 9812
```

Por tanto, vaya quitando cifras (dividiendo el número entre 10 sucesivamente, y contando las divisiones hechas) mientras el número obtenido sea mayor que 9 (o sea, mientras tenga más de una cifra).

```
Ejemplo1:  Introduce un nº natural:   98125   --> Cantidad dígitos = 5
Ejemplo2:  Introduce un nº natural:       0   --> Cantidad dígitos = 1
```

```
#include <iostream>
using namespace std;
int main()
{
    unsigned  numero, digitos;

    cout << "INTRODUCE UN NUMERO NATURAL: ";
    cin >> numero;

    digitos = 0; // AÚN NO HEMOS CONTADO NINGUN DÍGITO
    while ( numero >= 10 )
    {
         numero = numero / 10; //ACORTAMOS EL Nº, QUITÁNDOLE EL ÚLTIMO
                               //DIGITO
         digitos++;   // CONTAMOS EL DIGITO QUE ACABAMOS QUE QUITAR
    }

    digitos++; //CONTAMOS EL DIGITO QUE QUEDA AL FINAL (CUANDO num YA NO
               //SEA >= 10)
    cout << "CANTIDAD DE DIGITOS=" << digitos << endl;
    return 0;
}
```

3.- Calcule el número que ocupa una posición dada (**pos**) en la secuencia de Fibonacci: 0, 1, 1, 2, 3, 5, 8, 13, ... Observe que, a partir del 3º, cada número de la secuencia se obtiene sumando los dos anteriores a él. Por tanto, el programa deberá recordar siempre los 2 números previos al que se quiera calcular en cada momento, salvo para el 1º y el 2º de la secuencia, cuyos valores se conocen de partida (el 0 y el 1, y que se pueden mostrar sin hacer cálculos).

```
     Ejemplo1:
Introduce la posición del número de la secuencia que desea obtener: 5
El elemento que ocupa la posición 5  en la secuencia de Fibonacci es el 3
     Ejemplo2:
Introduce la posición del número de la secuencia que desea obtener: 3
El elemento que ocupa la posición 5 en la secuencia de Fibonacci es el 1
```

```
#include <iostream>
using namespace std;
int main()
{
    int c, b, a, pos;
    cout << "INTRODUCE LA POSICION DEL NUMERO DE FIBONACCI A OBTENER: ";
    cin >> pos;

    if (pos == 1)
        c = 0;
    else if (pos==2)
        c = 1;
    else
    {
        a = 0;
        b = 1;
        for (int p=3;  p<=pos;  p++)
        {
            c = a + b; /* calculamos el nº que va en la posiciónp actual:
la suma de los dos anteriores a él*/

//calculamos los siguientes valores que deben tomar a y b por si se
//calcula otro c:
            a = b; // el que es el b ahora mismo, será el a para la
//siguiente vuelta
// (ya podemos 'borrar', en la siguiente instrucción, este valor de b
//porque ya se ha copiado en a)
            b = c; // el que es el c ahora mismo, será el b para la
//siguiente vuelta

        }
    }
    cout << "EL TERMINO DE LA POSICION " <<  pos  << " ES EL: " << c <<
endl;
    return 0;
}
```

4.- Programa que calcule el cociente y el resto de la división entera de dos números naturales (no negativos) leídos por teclado, sin usar los operadores de C (/ y %). Recuerde que el cociente de la división entera es el resultado de una serie de restas sucesivas del divisor al dividendo, hasta que quede un nº menor que el divisor, el cual será el resto. En caso de que el divisor sea cero (lo cual no debería ocurrir), indíquelo mediante un mensaje y no realice ningún cálculo.

```
Ejemplo1:   10 / 3  ---> 10  - 3 = 7  (1° resta)
      7  - 3 = 4  (2° resta)
      4  - 3 = 1  (3° resta)
      1  --> fin (porque dividendo es menor que divisor)
--> hemos hecho 3 restas: cociente es 3, y el resto sería 1 (el ultimo
divisor)

Ejemplo2:    2 / 3--> fin (porque  dividendo es menor que divisor)
--> hemos hecho 0 restas: cociente es 0, y el resto sería 2 (el ultimo
divisor)
```

```
#include <iostream>
using namespace std;
int main()
{
    unsigned  dividendo, divisor, cociente, resto;

    cout << "INTRODUCE EL DIVIDENDO Y EL DIVISOR: ";
    cin >> dividendo >> divisor;

    if ( divisor == 0 )
        cout << "NO SE PUEDE HACER ESA DIVISIÓN, PORQUE EL DIVISOR NO
PUEDE SER CERO";
    else
    {

        cociente = 0;
        while ( dividendo >= divisor )
        {
            dividendo = dividendo - divisor;
            cociente++;   // CADA RESTA REALIZADA INCREMENTA EN 1 EL
COCIENTE
        }
        resto = dividendo;

        cout << "COCIENTE=" << cociente << endl << "RESTO=" << resto <<
endl;
    }

    return 0;
}
```

5.- Programa que pida por teclado un numero natural n (obligue a que sea menor que 10) y realice un dibujo siguiendo el patrón de los ejemplos siguientes:

Ejemplo 1:	Ejemplo 2:
INTRODUCE LA ALTURA (<10): 5	INTRODUCE LA ALTURA (<10): 3
54321**12345	321**123
4321****1234	21****12
321******123	1******1
21********12	
1**********1	

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

int main()
{
    int n;

    do{
        cout << "INTRODUCE LA ALTURA (<10): ";
        cin >> n;
    }while( n >= 10 );

    int max_asteriscos = 2;
    int max_numero = n;

    for (int f=1; f<=n; f++)
    {
    // PINTAMOS LOS NUMEROS INICIALES (CRECIENTES) DE LA FILA
        for (int i=max_numero; i>=1; i--){
            cout << i;
        }

    // PINTAMOS LOS ASTERISCOS QUE VAN DESPUÉS
        for (int i=1; i<=max_asteriscos; i++){
            cout << "*";
        }

    // PINTAMOS LA PARTE FINAL DE LA FILA (LOS NUMEROS CRECIENTES)
        for (int i=1; i<=max_numero; i++){
            cout << i;
        }

        cout << endl;  // CAMBIAMOS DE FILA

        // NOS PREPARAMOS PARA LA SIGUIENTE FILA, PARA LA QUE:
        max_asteriscos += 2;
        max_numero--;
    }
    return 0;
}
```

Ejercicios de refuerzo

6.- Implementar un programa que pida un numero seguido de cuatro letras y muestre cada una de las letras codificada según el número leído. Dados una letra **c** mayúscula y un número natural **n**, devuelva la letra **n** posiciones en el alfabeto a continuación de **c**. El alfabeto deberá considerarse circular. Esto es, detrás de la 'Z' vuelve a estar la 'A'. Por ejemplo, para la letra c = 'B' y n = 3 devolvería la letra 'E'; para la letra c = 'X' y el número n = 4 el devolvería 'B'.

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

int main()
{
    char c,l;
    int n, posicion;
    cout<< "Dame una posición y 4 letras "<<endl;
    cin>> n;

    for(int i=1; i<=4; i++)
    {
        cin>> c;
        posicion= int(c) + n;
        if( posicion > int('Z') )
        {
            posicion = posicion - (int('Z') - int('A')) - 1;
        }
        cout << char(posicion);
    }
    return 0;
}
```

7.- Escribir un programa que, dados tres números enteros **a, b** y **c**, muestre cuantos números enteros entre a y b (ambos incluidos) son divisibles por c. Por ejemplo, para a = 10, b = 20 y c = 3 el programa debería mostrar el valor 3 ya que entre 10 y 20 hay tres números divisibles por 3 (12, 15 y 18).

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

int main(){
    unsigned a,b,c,cont;

    cout<<"Dame a,b y c: ";
    cin>>a>>b>>c;

    cont=0;
    for(int i=a; i<=b; i++){
        if(i%c==0){
            cont++;
        }
    }
```

```
    cout<< "Hay: "<< cont <<" números.";

    return 0;
}
```

8.- Escriba un programa que lea de teclado una cantidad de tiempo en segundos y escriba por pantalla su equivalente en días, horas, minutos y segundos. Por ejemplo: 3663 segundos son 1 hora, 1 minuto y 3 segundos. (Ayuda: El operador 'resto de la división entera' permite solucionar este problema). Un ejemplo de la ejecución de este programa podría ser (en negrita lo tecleado por el usuario);

```
Dime el tiempo en segundos: 88204
88204 segundos son:
1 dia y
30 minutos y
4 segundos
```

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

int main(){
      int t, horas, dias, seg, min;
      cout<<"Dime el tiempo en segundos"<<endl;
      cin>>t;

      dias=t/(60*60*24);
      t=t%(60*60*24);
      horas=t/(60*60);
      t=t%(60*60);
      min=t/60;
      seg=t%60;

      if (dias!=0){
            if(dias==1){
                  cout<<dias<<" dia,"<<endl;
            }else{
                  cout<<dias<<" dias,"<<endl;
            }
      }
      if (horas!=0){
            if(horas==1){
                  cout<<horas<<" hora,"<<endl;
            }else{
                  cout<<horas<<" horas,"<<endl;
            }
      }
      if (min!=0){
            if(min==1){
                  cout<<min<<" minuto,"<<endl;
            }else{
                  cout<<min<<" minutos,"<<endl;
            }
      }
      if (seg!=0){
            if(seg==1){
                  cout<<seg<<" segundo."<<endl;
            }else{
```

```
                    cout<<seg<<" segundos."<<endl;
              }
       }
}
```

9.- Supongamos que, dado un número **n**, realizamos el siguiente proceso:

Si **n** es par entonces se divide por 2.

Si **n** es impar entonces se multiplica por 3 y se le suma 1

Si esta operación se repite sucesivamente para cada número obtenido entonces al final obtendremos el número 1. Por ejemplo:

Para el número 5 obtenemos los números 16, 8, 4, 2 y 1. Por tanto, en 5 pasos hemos obtenido el número 1.

Para el número 12 obtenemos los números 6, 3, 10, 5, 16, 8, 4, 2 y 1. Por tanto, en 9 pasos hemos obtenido el número 1.

Para el número 1 directamente ya tenemos el número 1 en 0 pasos.

Escribir un programa que, dado un número natural, muestre cuantas veces hay que aplicar el proceso descrito para obtener el número 1.

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

int main(){
      int n, cont;

      cout<<"Escribe el numero"<<endl;
      cin>>n;

      cont=0;
      while(n!=1){
            if(n%2==0){
                  n=n/2;
            }else{
                  n=(n*3)+1;
            }
            cont++;
      }

      cout<<"En "<<cont<<" pasos hemos encontrado el 1"<<endl;
}
```

10.- Realizar un programa que lea una secuencia de caracteres terminada en un punto (.). El programa deberá mostrar la longitud de cada secuencia de caracteres entre dos comas y la longitud de las secuencias antes la primera coma y detrás de la última. Por ejemplo, para la secuencia *aabc,adddff,ddf,ee,,ddd,e*. El programa deberá mostrar *4 6 3 2 0 3 1*

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

int main()
{
    char l;
    int cont=0;
```

```
    cout << "Dame secuencia de caracteres (. para terminar): ";
    cin >> l;

    while (l!='.')
    {
        if (l==',')
        {
            cout << cont << " ";
            cont=0;
        } else {
            cont++;
        }

        cin >> l;
    }

    cout << cont << endl;

    return 0;
}
```

11.- Realizar un programa que lea un único número por teclado en una variable de tipo *unsigned* y muestre por pantalla cuantos dígitos pares tiene. Por ejemplo: 34461 tiene 3 dígitos pares; 55914 tiene 1 dígito par.

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

int main(){
    unsigned N;
    unsigned d=0;
    unsigned cont=0;

    cout<< "Dame número: ";
    cin>> N;

    do{
        d=N%10;

        if(d%2==0){
            cont++;
        }

        N=N/10;

    }while(N!=0);

    cout<< "El número de pares es: "<< cont <<endl;

    return 0;
}
```

12.- Realiza un programa que lea una lista de números enteros terminada en 0 y diga si en la lista hay más números negativos que números positivos, más positivos que negativos o

la misma cantidad de positivos que de negativos. El 0 no forma parte de la lista, sólo se utiliza para indicar el final de la misma.

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

int main()
{
    int n, cont1=0, cont2=0;

    cout<< "Dame números: ";
    cin>>n;

    while(n!=0)
    {
        if(n<0)
        {
            cont1++; //Negativos
        }
        else
        {
            cont2++; //Positivos
        }
        cin>>n;
    }

    if(cont1<cont2)
    {
        cout<< "Hay más positivos que negativos"<<endl;
    } else if(cont1>cont2)
    {
        cout<< "Hay más negativos que positivos"<<endl;
    } else
    {
        cout<< "Mismo número de negativos y positivos"<<endl;
    }

    return 0;
}
```