

Estructuras de repetición

Fundamentos de Programación
Fundamentos de Programación I

Estructuras iterativos

Sentencia for

```
for(inic; cond; increm)
{
    sentencia1;
    sentencia2;
}
```

Sentencia while

```
while( expresion logica)
{
    sentencia;
    sentencia2;
}
```

Sentencia do..while

```
do
{
    sentencia;
    sentencia2;
}
while (expresión lógica);
```

Recomendaciones de uso???

- **Ejercicio 1: Sumar n números enteros introducidos por teclado, siendo n un valor entero también introducido por teclado.**

Entradas: el número de enteros a sumar (n) y esos números (num)

Salidas: el resultado de la suma (suma)

Análisis: utilizaremos una estructura de repetición,pero cuál??

```
//Programa que suma n numeros enteros
#include <iostream>

using namespace std;

int main()
{
    //declaración de variables
    int n;
    int x;
    int suma;
    int i;

    cout << "Introducir el valor de n:";
    cin >> n;

    suma = 0;
    cout << "Introducir " << n << " numeros enteros\n";
    for (i=0 ;i < n; i++)
    {
        cin >> x;
        suma = suma + x; //acumulo x en la suma
    }
    cout << "Resultado de la suma: " << suma;

    return 0;
}
```

- **Ejercicio 2: Sumar todos los números introducidos por teclado antes de introducir un 0.**

Entradas: Números que se quieren sumar...

Salidas: El resultado de la suma

Análisis: se van introduciendo números y cuando se introduce un 0
Se debe parar...que estructura de repetición nos conviene utilizar??

```
//Programa que suma numeros hasta introducir un 0=fin
#include <iostream.h>

int main()
{
    //declaración de variables
    int x;
    int suma;
    bool fin;

    //Inicializar variables
    fin = false;
    suma = 0;

    cout << "Introducir los numeros enteros a sumar (0 = fin)\n";
    do
    {
        cin >> x;
        if (x != 0)
            suma = suma + x;//acumulo x en la suma
        else
            fin = true;
    }while (fin != true);

    cout << "Resultado de la suma: " << suma;

    return 0;
}
```

- **Ejercicio 3: Encontrar el primer valor de n para el cual 2^n es mayor que 1000**

Entradas: Ninguna

Salidas: el valor de n que cumpla la condición anterior $2^n > 1000$

Análisis: ¿Qué estructura de repetición puedo utilizar?

```
//Programa calcula el numero n tal que  $2^n > 1000$ 
#include <iostream>
using namespace std;

int main()
{
    //declaración de variables
    int n; //lo que hay que buscar
    int pot_2; //para calcular la potencia de dos

    //Valores iniciales para empezar a buscar
    n = 0;
    pot_2 = 1;

    while (pot_2 <= 1000)
    {
        //Calculo la potencia
        pot_2 = pot_2 * 2; //siguiente potencia
        n = n + 1;
    }

    cout << "Resultado " << n << endl;

    return 0;
}
```

- **Ejercicio 4: Calcular cuantas vocales se han introducido por teclado antes de aparecer el carácter '#'**

Entradas: Una secuencia de caracteres terminados con el carácter '#'

Salidas: El número de vocales que hay en la secuencia (total)

Análisis: Se van leyendo caracteres y si es una vocal se incrementa un contador (cont), como sabemos si es una vocal???

```
//Programa que cuenta el numero de vocales en una secuencia de caracteres
#include <iostream>

int main()
{
    //declaración de variables
    char ch;
    int cont; //numero de vocales

    //lectura de datos
    cout << "Introducir caracteres (# = fin)";
    cont = 0; //todavía no hay vocales
    do
    {
        cin >> ch; //leo carácter
        switch(ch)
        {
            case 'a': case 'A':
            case 'e': case 'E':
            case 'i': case 'I':
            case 'o': case 'O':
            case 'u': case 'U': cont = cont + 1;
                break;
        }
    }while( ch != '#');

    cout << "Vocales introducidas: " << cont << endl;

    return 0;
}
```

- **Ejercicio 5:** Escribir un programa que muestre los valores de la recta $y = mx + b$ en el rango $[r0..r1]$. Los valores de los coeficientes m y b , así como el rango deben ser introducidos por teclado. Por pantalla deben aparecer una secuencia de líneas según el siguiente formato:

```
Recta: y= 2x+1 Rango:[1..3]
Valor de x = 1, Valor de y = 3
Valor de x = 2, Valor de y = 5
Valor de x = 3, Valor de y = 7
```

Entradas: coeficientes m y b , y el rango $[r0..r1]$
Salidas: Los puntos (x,y) por los que pasa la recta en el rango $[r0..r1]$
Análisis: Que operaciones se repiten y cuantas veces???

```
#include <iostream>

int main()
{
    int m, b; //coeficientes de la ecuacion de la recta
    int r0,r1; //puntos extremos del intervalo
    int x, y; //para el bucle for

    cout << "Introducir coeficientes de la recta: y=mx+b\n";
    cin >> m >> b;

    cout << "Recta: "<< "y=" << m << "x + " << b ;
    cout << "\tRango: [" << r0 <<" " << r1 << "]" ;
    //Calculo los puntos de la recta
    for (x = r0 ; x <= r1; x++)
    {
        //Calculo valores de y para cada valor de x
        y = m * x + b;

        //Muestro resultado
        cout << " Valor de x: " << x << ", Valor de y: " << y <<endl;
    }

    return 0;
}
```

- **Ejercicio 6:** Escribir un programa que dibuje los puntos por los que pasa una circunferencia especificada por su radio. Asumimos que la circunferencia esta centrada en el origen de coordenadas.

```
#include <iostream>
#include <stdlib.h>
#include <math.h>

using namespace std;

int main()
{
    //Declaración de variables
    float x, y;
    float R;

    //introduce el radio de la circunferencia a dibuja
    cout << "Introduce el radio de la circunferencia\n";

    cin >> R;

    //Calculo los distintos puntos por los que pasa
    for (x = -R ; x <= R; x++)
    {
        y = sqrt( (R*R) - (x*x) );
        //Muestro los puntos
        cout << "x =" << x << "->" << y << ", " << -y << endl;
    }

    system("PAUSE");
    return 0;
}
```

Problema: División entera de dos números

Entrada: Dos números, dividendo (dendo) y divisor (disor)

Salida: Dos números, cociente (C) y resto (R)

```
#include <iostream>
#include <stdlib.h>
using namespace std;

int main()
{
    int dendo, disor;
    int cociente, resto;

    cout << " INTRODUCIR DIVIDENDO DIVISOR\n";
    cin >> dendo >> disor;

    cociente = 0;
    //resto = dendo;

    while ( dendo >= disor)
    {
        cociente = cociente +1 ;
        dendo = dendo - disor;
    }
    resto = dendo;
    cout << cociente << " " << resto << endl;
    system ("PAUSE");
}
```

FP-PFI Curso 2005-2006

¿Qué pasa si disor==0?
Modificar el algoritmo
Para que tenga en cuenta
Ese caso.

15

- **Ejercicio 6:** Escribir un programa para jugar al master mind. El jugador 1 introduce una combinación de cuatro números del 1 al 4, sin posibilidad de repetirlos. Posteriormente se borra la pantalla, y el jugador 2 debe introducir combinaciones hasta acertar la combinación original introducida por el jugador 1. Tras la introducción de cada combinación por el jugador 2, se debe mostrar el número de aciertos obtenidos (es decir, cuantas posiciones han sido acertadas). Al terminar el programa, se debe mostrar como resultado el número de intentos empleados en resolverlo

Entradas: Combinaciones de 4 números del jugador 1 y n combinaciones del jugador 2 (hasta que acierte)

Salidas: Número de aciertos en cada jugada y número de intentos al final.

Análisis: El programa pedirá una combinación al jugador 2, comprobará cuantos numeros ha acertado de la combinacion del jugador 1. Si el numero de aciertos no es igual a 4, volverá a solicitar una nueva combinacion (se repite el proceso anterior) hasta que acierte.

FP-PFI Curso 2005-2006

16

```

#include <iostream>

int main()
{
    //Declaracion de variables
    int c1, c2, c3, c4;
    int n1, n2, n3, n4;
    int intentos, aciertos;
    bool terminar = false;

    cout << "Jugador 1: Introduce 4 numeros de la combinacion (c1 c2 c3 c4)\n";
    cin >> c1 >> c2 >> c3 >> c4;
    //limpio la pantalla
    system("CLS");
    intentos = 0;
    do{
        cout << "Jugador 2: introduce jugada ( c1 c2 c3 c4)\n";
        cin >> n1 >> n2 >> n3 >> n4;
        intentos = intentos + 1;
        aciertos = 0;
        if (c1 == n1) aciertos = aciertos + 1;
        if (c2 == n2) aciertos = aciertos + 1;
        if (c3 == n3) aciertos = aciertos + 1;
        if (c4 == n4) aciertos = aciertos + 1;
        //Compruebo si ya se termina el juego
        if (aciertos == 4)
            terminar = true;
        cout << "Numero de aciertos: " << aciertos << endl;
    } while(terminar !=true);
    cout << "Numero de intentos: " << intentos;
    return 0;
}

```

FP-PFI Curso 2005-2006

17

- Ejercicio 7: Utilizando la serie de Taylor para
- e^x , hacer un programa que calcule dicha función con una precisión dada por el número de términos que se suman

FP-PFI Curso 2005-2006

18

```

#include <iostream>
#include <stdlib.h>
#include <math.h>

using namespace std;

int main()
{
    int x, n;
    float potencia, factorial, termino, res;
    int i;

    cout << "Este programa la potencia de e elevado a x\n";
    cout << "Introduce el exponente\n";
    cin >> x;
    cout << "Introduce el numero de terminos de la serie a calcular\n";
    cin >> n;

    //Calculos a realizar
    res = 1;
    potencia = 1;
    factorial = 1;
    for (i = 1; i < n; i++)
    {
        potencia = potencia * x; //potencia
        factorial = factorial * i; //factorial
        termino = potencia / factorial;
        res = res + termino;
    }
    cout << "La aproximacion es: "<< res << " con numero de termino " << n<< endl;
    system("PAUSE");
}

```

FP-PFI Curso 2005-2006

19

- **Ejercicio 8:** Escribir un programa que genere y muestre x ($1 < x < 6$) números aleatorios no repetidos enteros entre 0 y 9 y calcule cual es el máximo, cual es el mínimo y cual es la media de los números. El valor de x se introducirá por teclado al iniciar la ejecución del programa.

Entradas: La cantidad de números aleatorios que se van a generar (n)

Salidas: máximo, mínimo y media de esos (n) números

Análisis: ¿Cómo se calcula un número aleatorio?? Ayuda:
Existe una funcion `rand()` que devuelve un numero aleatorio en el intervalo `[0, RAND_MAX]`

FP-PFI Curso 2005-2006

20

```

#include <iostream>
#include <stdlib.h>

using namespace std;

int main()
{
    //Declaracion de variables
    int N, i, n_ale, xr;
    int maximo, minimo, suma;
    float media;

    cout << "Introduce la cantidad de numeros aleatorios a generar\n";
    cin >> N;
    //inicializo el maximo y el minimo
    maximo = -1;
    minimo = 10;

    for (i=0; i < N; i++)
    {
        xr = rand(); //Calculo el numero aleatorio

        n_ale = xr % 10; //Lo cambio de rango
        cout << n_ale << " "; //muestro el resultado
        if ( maximo < n_ale) //Compruebo si es el maximo
            maximo = n_ale;
        if (minimo > n_ale) //Compruebo si es el minimo
            minimo = n_ale;
        suma = suma + n_ale;
    }
    media = suma / N; //Calculo la media
    cout << endl << minimo << endl;
    cout << maximo << endl;
    cout << media << endl;

    return 0;
}

```

FP-PFI Curso 2005-2006

21

```

#include <iostream>
#include <stdlib.h>

using namespace std;

int main()
{
    //Declaracion de variables
    int a, b, cociente, resto;

    cout << "Este programa calcula la division entera entre dos numeros\n";

    cout << "Introduce el dividendo y el divisor\n";
    cin >> a >> b;
    cociente = 0;
    if ( b == 0)
        cout << " divisor igual a cero\n";
    else
    {
        while (a >= b)
        {
            a = a - b;
            cociente = cociente + 1;
        }
        resto = a;

        cout << a << "/" << b << '=' << cociente << " Resto = " << resto;
    }
    system("PAUSE");

    return 0;
}

```

FP-PFI Curso 2005-2006

22