## OBJETIVOS

- Familiarizarse con los conceptos básicos: elementos del lenguaje, uso de las funciones básicas y construcción de programas sencillos en lenguaje C.
- Profundizar en la construcción en lenguaje C de algunos métodos estáticos ya implementados en Java.

## PARTE 1: EJERCICIOS BÁSICOS PARA HACER EN EL AULA

### EJERCICIO 1 – CÁLCULO DEL FACTORIAL DE UN NÚMERO NATURAL CON FUNCIONES ALMACENADAS EN ARCHIVOS INDEPENDIENTES Y POR TANTO CON ARCHIVOS DE CABECERAS CREADOS POR EL USUARIO.

Los ejercicios realizados en el Boletín C.0 se resolvieron programando un único archivo C denominado holaMundo.c u holaYo.c respectivamente, al que se le incluía al principio una librería que viene incorporada en el compilador de C para el manejo de funciones de E/S: **stdio.h**.

Ahora se trata de construir un proyecto con tres archivos independientes en un mismo proyecto que denominaremos respectivamente *testNumeroCombinatorio.c*, *numeroCombinatorio.h* y *numeroCombinatorio.c*. Realice los siguientes pasos:

1. Cree un proyecto llamado: **NumeroCombinatorio**.
2. Cree un programa C (*Source File*) llamado: *testNumeroCombinatorio.c*.
3. Incluya la librería que permite usar las funciones de entrada/salida e incluya el archivo de cabecera *numeroCombinatorio.h***.** Recuerde que mientras las librerías del compilador de C necesitan en la sintaxis <_.h>, las creadas por el usuario van entre comillas: "_.h".
4. Programe la función ***int main (void)*** con las sentencias necesarias para que se visualice el mensaje "Teclee el numerador y el denominador de un número combinatorio:\n" y a continuación permita introducirlos por teclado (recuerde incluir *fflush (stdout)* antes de la sentencia y que las variables que se necesitan en un programa C hay que definirlas en las primeras líneas del programa) .
5. Una vez capturados los valores invoque a la función *numeroCombinatorio(m, n)* y visualice el mensaje: " xx sobre yy es zz" donde xx es el valor introducido para m, yy el introducido para n y zz el resultado obtenido por la función numeroCombinatorio.
6. En la ventana del explorador de proyectos, encima del nombre del proyecto con –**BC**- se elige new Header File y se crea un archivo de cabeceras denominado *numeroCombinatorio.h*, en el que se incluyen, entre otras cosas, los prototipos de las funciones que serán invocadas desde el archivo principal y que se han codificado en otros archivos (esto le sirve al compilador para tener declaradas las funciones cuando compile el citado archivo principal). Incluya por tanto:

```
#ifndef NUMEROCOMBINATORIO_H_
#define NUMEROCOMBINATORIO_H_

long numeroCombinatorio(int,int);
long factorial(int);

#endif /* NUMEROCOMBINATORIO_H_ */
```

7. Por último, en otro (*Source File*) llamado *numeroCombinatorio.c*, en el que se debe incluir como primera línea #include "numeroCombinatorio.h", programe las dos siguientes funciones:

    a. **long factorial (int n),** que devuelve el factorial del parámetro de entrada n. Recuerde que el factorial de un número *n* es $n * (n-1) * (n-2) * ... * 1$.
    b. **long numeroCombinatorio(int m, int n),** que apoyándose en la función anterior devuelve el número combinatorio de m sobre n:

$$\binom{m}{n} = \frac{m!}{n!(m-n)!}$$

Pruebe el resultado del ejercicio con 8 sobre 5 cuyo resultado es 56, lo mismo que 8 sobre 3.

## PARTE 2: EJERCICIOS SOBRE TRANSFORMACIÓN DE MÉTODOS ESTÁTICOS JAVA EN FUNCIONES C PARA HACER EN EL AULA

### EJERCICIO 2 – FUNCIONES SOBRE ENTEROS.

Cree un proyecto que se denominará **Enteros**.

1. Cree un archivo de cabecera de nombre "enteros.h" donde debe incluir:
    a) Los ficheros de cabecera que necesite para el manejo de las funciones de entrada/salida <*stdio.h*>, las matemáticas <*math.h*> y la función valor absoluto abs, que se encuentra en <*stdlib.h*>.
    b) La definición de un tipo enumerado (use la declaración *typedef*) denominado Logico que pueda tomar los valores FALSO y CIERTO, para usarlo en lugar de sus homónimos de Java.
    c) Y, como en el ejercicio anterior, los prototipos de las funciones que más abajo se describen (añádalos ya, con independencia de que las funciones las vaya realizando progresivamente mientras avanza en el ejercicio)
2. Todas la funciones que figuran más abajo se implementarán en el archivo denominado "*funcionesDeEnteros.c*". Recuerde que debe incluir como primera línea el fichero de cabecera que acaba de realizar.
3. Por último, para probar las citadas funciones se creará otro archivo denominado *"testEnteros.c*" donde estará codificada la función ***int main(void)*** que irá solicitanto oportunamente los datos por teclado invocando a las funciones que se quieran probar y visualizando los resultados de las mismas. Recuerde también incluir como primera línea el fichero de cabecera enteros.h.

Los métodos Java a transformar en funciones C son los siguientes:

a) Comprueba que el valor v está en el intervalo [a,b]
    **Logico estaEn(int a, int b, double v)**
b) Máximo común divisor (recuerde al algoritmo de Euclides que consiste en dividir el número mayor entre el menor. Si la división es exacta, el divisor es el m.c.d.; en otro caso, se continúa dividiendo el divisor entre el resto obtenido y así sucesivamente hasta obtener una división exacta, siendo el último divisor el m.c.d.)
    **int mcd(int m, int n)**
c) Mínimo común multiplo (se recuerda que es el producto de los números dividido por el m.c.d. de los mismos)
    **int mcm(int m , int n)**

## PARTE 3: EJERCICIOS SOBRE TRANSFORMACIÓN DE MÉTODOS ESTÁTICOS JAVA EN FUNCIONES C PARA HACER EN CASA

**EJERCICIO 3 – MÁS FUNCIONES SOBRE ENTEROS.**

Siguiendo la metodología del ejercicio anterior, aumente las funcionalidades de "*funcionesDeEnteros.c"* añadiendo las siguentes funciones:

1. **Logico esPrimo(int n**), que compruebe si un número es primo. Si el número es negativo o cero se devuelve FALSO (recuerde que basta probar si es divisible por los números comprendidos desde 2 hasta la raíz cuadrada del propio número)
2. **int sumaPrimosMenores(int n)**, que dado un número n>0, devuelva la suma de los números primos menores o iguales que n.
3. **int sumaNoPrimosEntre(int a, int b)**, que dados dos numeros a y b, positivos y tales que a<b, devuelva la suma de los números que no son primos y que están en el intervalo [a,b]
4. **Logico formanTrianguloRectangulo(int a, int b, int c)**, que dados tres números a, b y c, positivos, compruebe si forman un triángulo rectángulo; recuerde el teorema de Pitágoras.