

Práctica 4. Recursividad I

Objetivos de la práctica.

- Implementar subalgoritmos recursivos en C++.
- Aprender a hacer pruebas de nuestros programas.

Parte 1.

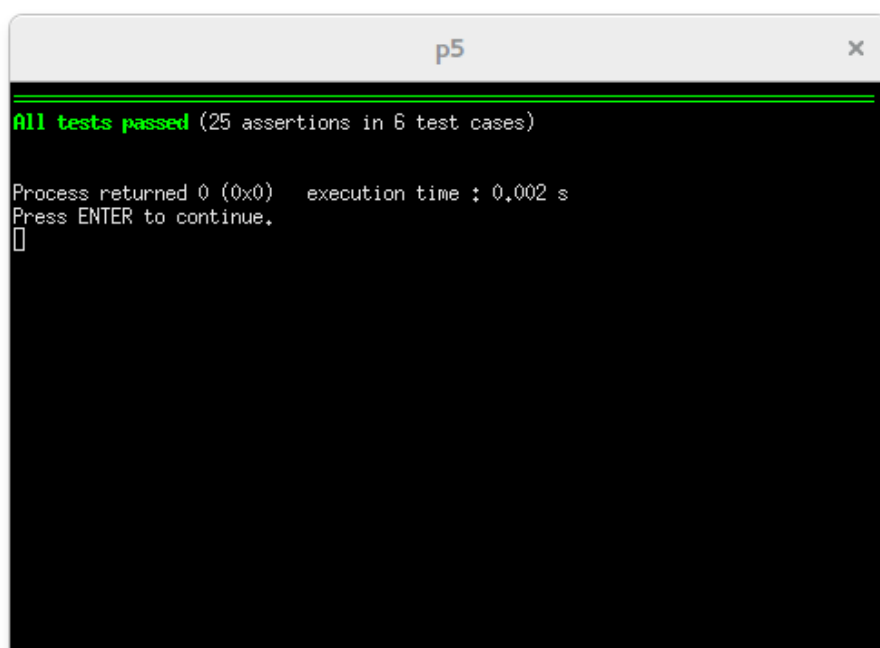
Como vimos en prácticas anteriores, al desarrollar nuestros programas, lo primero que debemos comprobar es que dichos programas compilan. Una vez que nuestros programas compilan hay que comprobar si dichos programas tienen el comportamiento esperado. Para la segunda tarea se suelen utilizar [*pruebas de software*](#). Existen diversos tipos de pruebas software, las que veremos en esta práctica se conocen como [*pruebas unitarias*](#). En estas pruebas lo que comprobamos es si un subalgoritmo funciona correctamente utilizando una serie de casos de prueba que se conocen como la *suite de pruebas*. En un caso de prueba se indica un algoritmo a probar, unas entradas para ese algoritmo y la salida esperada del algoritmo para los valores de entrada dados. Si nuestro programa pasa los casos de prueba habremos aumentado la fiabilidad del mismo, pero no habremos probado que es correcto (para ello es necesario utilizar métodos formales que veréis en el segundo curso).

En esta práctica veremos como utilizar una librería de C++ para hacer pruebas unitarias. En concreto utilizaremos la librería [*Catch*](#). Para ello deberás seguir los siguientes pasos.

1. Crea un nuevo proyecto de consola en CodeBlocks llamado practica4.
2. Elimina de dicho proyecto el fichero main.cpp. Para ello en el árbol mostrado en *Projects*, pulsar con el botón derecho sobre el fichero main.cpp y seleccionar la opción *Remove file from project*.
3. Ve al aula virtual y descarga los ficheros practica4.cpp y catch.hpp que encontrarás en la carpeta de la práctica 4.
4. Copia los ficheros que has descargado en la carpeta donde has creado el proyecto de Codeblocks.
5. Añade los ficheros practica4.cpp y catch.hpp al proyecto de Codeblocks. Para ello en el árbol mostrado en *Projects*, pulsar con el botón derecho sobre practica4 y seleccionar la opción *Add files...* desde ahí selecciona los ficheros practica4.cpp y catch.hpp.
6. Abre el fichero practica4.cpp. En dicho fichero verás que aparecen las cabeceras de varias funciones que tienes que completar. Además al final del fichero aparecen unos casos de test.
7. Antes de completar el código de los distintos subalgoritmos prueba a compilar y ejecutar el código. Como puedes comprobar aparece que varias de las pruebas han fallado.
8. A continuación debes completar los siguientes ejercicios rellenando el código correspondiente a cada subalgoritmo; además debes proporcionar la especificación (precondición y postcondición) de cada subalgoritmo.
 1. Implementar una función recursiva para el cálculo del factorial.
 2. Implementar una función recursiva para el cálculo de la potencia a^n para cualquier real a y para cualquier entero n (notar que n puede ser negativo).
 3. Construye subprogramas recursivos que a partir de un vector de enteros v de tamaño n (con $n > 0$) calculen:

- a. La suma de sus elementos.
- b. La media de sus elementos.
- c. El máximo de sus elementos.
- d. El mínimo de sus elementos.

A medida que vayas definiendo cada uno de estos subalgoritmos vuelve a compilar el programa, verás que si vas implementando de manera correcta los distintos algoritmos cada vez aparecen más pruebas superadas. Si una vez que has implementado todos los subalgoritmos, ejecutas el programa y obtienes una pantalla similar a la siguiente es que habrás completado con éxito esta parte de la práctica.

A screenshot of a terminal window titled 'p5'. The window has a black background with green and white text. The text reads: 'All tests passed (25 assertions in 6 test cases)' in green. Below this, in white, it says 'Process returned 0 (0x0) execution time : 0.002 s' and 'Press ENTER to continue.' followed by a white cursor icon.

```
p5
All tests passed (25 assertions in 6 test cases)
Process returned 0 (0x0) execution time : 0.002 s
Press ENTER to continue.
█
```

Parte 2. Problema de las torres de Hanoi.

En el siglo XVII, un juego llamado las Torres de Hanoi hizo su aparición en Europa y se decía que representaba una tarea que estaba realizándose en el templo de Brahma. En el momento de la creación del mundo, los sacerdotes recibieron una plataforma de bronce sobre la cual había tres agujas de diamante. En la primera aguja estaban apilados sesenta y cuatro disco de oro, cada uno ligeramente menor que el que está debajo de él. A los sacerdotes se les encomendó la tarea de pasarlos todos de la primera aguja a la tercera, con dos condiciones: sólo puede moverse un disco a la vez y ningún disco podrá ponerse encima de otro más pequeño. Se dijo a los sacerdotes que, cuando hubieran terminado de mover los sesenta y cuatro discos, llegaría el fin del mundo.

Construir un programa que liste todos los movimientos que deben hacer los sacerdotes.

Nota: Diseñar e implementar la siguiente acción recursiva:

acción mueve(**ent** n:**entero**, **ent** origen: **carácter**, **ent** destino:**carácter**, **ent** auxiliar:
carácter)

{PRE: $n \geq 1$ }

{POST: escribe los movimientos necesarios para mover correctamente n discos desde
origen a destino }

¿Cuál es la complejidad de este algoritmo ?