Actividad 2.3

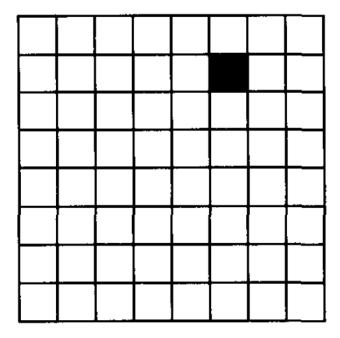
Embaldosado recursivo

Algoritmos y Estructuras de Datos

Tema 2: análisis de algoritmos recursivos

1º Grado en Ingeniería en Desarrollo de Contenidos Digitales
© Profesor Dr. Carlos Grima Izquierdo (<u>www.carlosgrima.com</u>)
U-tad (<u>www.u-tad.com</u>), curso 2014/15

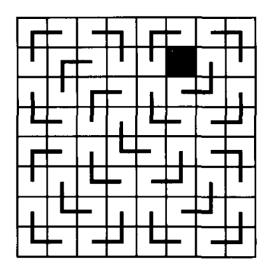
Tenemos un tablero cuadrado de lado «n» (n es potencia de 2), con un cuadrado especial (el marcado en negro). Por ejemplo:



El problema consiste en cubrirlo entero de baldosas, excepto el cuadrado negro (la "celda prohibida"). Cada baldosa ocupa tres celdas en «L», en cualquier orientación. Por ejemplo:

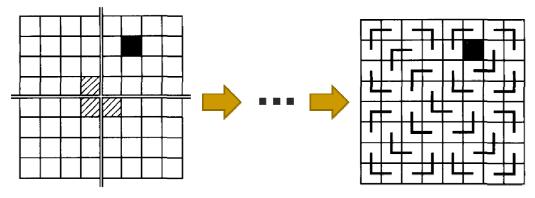


Debemos encontrar un algoritmo para, a partir del tamaño del lado ("n") y de la posición del hueco especial (sus coordenadas), rellenar todo de baldosas sin que sobre ningún hueco (excepto el hueco especial). Una solución sería:



La resolución puede realizarse de forma iterativa, pero es más fácil de forma recursiva, y por lo tanto nosotros lo haremos así:

- 1. Se divide el área total nxn en cuatro partes iguales. Estas cuatro partes son iguales por n es potencia de 2.
- 2. Se coloca una baldosa en medio, de tal modo que la baldosa no toque el cuadrante en donde está el hueco especial
- 3. Cada una de las cuatro cuadrantes es ahora un nuevo problema de embaldosado (pero ahora de tamaño n/2), en donde la parte de la baldosa central que ha caído en el cuadrante en cuestión es ahora el hueco especial de dicho nuevo problema.
- 4. Se procede recursivamente con cada una de estas cuatro partes hasta que lleguemos al problema trivial, que es un cuadrado con cuatro huecos en el cual uno de ellos es especial, y por lo tanto en los otros tres ponemos una baldosa



Se pide solucionar este problema en C++, aplicando recursividad. Para ello, ten en cuenta las siguientes consideraciones:

- Hay que aprovechar la filosofía de la orientación a objetos. Al menos tendremos una clase llamada AreaEmbaldosada que represente a todo el área a embaldosar. Y, dentro de ella, tendremos (entre otros) el método "embaldosar", que será el recursivo.
- Cada celda del área guardará un int, que contendrá el número (identificador) de baldosa que cubre ese hueco, o bien -1 si no hay ninguna baldosa aún, o bien 0 si se trata de la celda prohibida.
- Cada baldosa tendrá un número identificador, empezando por 1. Cada baldosa que se ponga tendrá un número consecutivo. Así pues, el identificador de la primera baldosa que se ponga será un 1, la siguiente un 2, la siguiente un 3, etc.
- El programa deberá imprimir por pantalla el área completa antes de poner la primera baldosa, e imprimirlo de nuevo después de poner cada una de las baldosas. Para que salga todo alineado, usar las posibilidades del printf().
- En este programa se valorarán los principios de máxima cohesión y mínimo acoplamiento, ya que se deberá pensar la mejor distribución de clases, atributos, métodos, parámetros, etc. Puedes saltarte la división entre interfaz y modelo sólo para que se imprima por pantalla el estado del área cada vez que se ponga una baldosa.
- El sistema de coordenadas para las celdas del área será (fila,columna), empezando en (0,0) en la celda superior izquierda (como en las matrices matemáticas).
- En C/C++, las matrices de dos dimensiones se pueden guardar contiguas en memoria. Primero se guarda la primera fila, a continuación la segunda, y así sucesivamente.

A continuación se especifica con un ejemplo la salida del programa:

