CC3301 Programación de Software de Sistemas

Tarea 1 – Semestre Otoño 2016 – Prof.: Luis Mateu

En esta tarea Ud. deberá implementar el juego de la vida usando vectores de bits. Estudie primero este juego en este URL:

http://es.wikipedia.org/wiki/Juego_de_la_vida

Ud. debe representar una matriz de 8x8 células usando un entero de 64 bits (tipo *unsigned long long*). Dada una matriz representada por el entero *m*, el siguiente mecanismo le permite determinar si una célula $c_{i,j}$ está viva o muerta:

$f_i = m >> (i*8)$	$i = 0, 1, 2, \ldots, 7$	*i*-ésima fila
$c_{i,j} = (f_i >> j) \& 1$	$j = 0, 1, 2, \ldots, 7$	*j*-ésima célula de f_i

Si $c_{i,j} = 1$ la célula está viva, de lo contrario está muerta.

Concretamente se le pide:

Parte a.-

Programe la función:

```
unsigned long long transicion(unsigned long long m);
```

Dado un estado de la matriz representada por *m*, calcula y entrega el siguiente estado (o turno). En esta función Ud. *no puede usar punteros, arreglos o strings*. Tampoco puede usar los operadores de multiplicación o división.

El detalle de las reglas para calcular el siguiente estado está explicado en el URL de más arriba. Pero en resumen:

- Una célula muerta con exactamente 3 células vecinas vivas "nace" (al turno siguiente estará viva).
- Una célula viva con 2 ó 3 células vecinas vivas sigue viva, en otro caso muere o permanece muerta (por "soledad" o "superpoblación").

Parte b.-

Programe en el archivo *lifegame.c* un comando que se invoque desde Unix de la siguiente manera:

```
$ ./lifegame 2 00000010 11000000 01000111 00000000  00000000 00000000 00000000 00000000
00000010
11000000
01000111
00000000
00000000
00000000
00000000
00000000

00000000
11000101
11000010
00000010
00000000
00000000
00000000
00000000

00000000
11000010
11000111
00000000
00000000
00000000
00000000
00000000
```

El primer parámetro corresponde al número *k* de transiciones que se debe calcular y los siguientes 8 corresponden a las filas de la matriz inicial de células. Su programa debe calcular y desplegar los siguientes *k* estados exactamente de la manera como se indica en el ejemplo. Para calcular cada transición Ud. debe usar la parte a. Descargue el archivo *t1.zip* de material docente y descomprímalo. Contiene un *Makefile* para

compilar su tarea y el programa *decod-args.c* en donde se muestra como transformar la matriz inicial señalada en los argumentos del comando a un entero en el formato *unsigned long long*.

Compile y ejecute el programa decod-args.c con:

```
$ make decod-args
$ ./decod-args 2 00000010 11000000 01000111 00000000  00000000 00000000 00000000 00000000
transiciones= 2, matriz inicial= 2c0470000000000 (en hexadecimal)
```

Programe su tarea en el archivo lifegame.c. Compile y ejecute con:

```
$ make lifegame
$ ./lifegame 2 00000010 11000000 01000111 00000000  00000000 00000000 00000000 00000000
...
```

Importante: Pruebe su tarea en anakena.dcc.uchile.cl. No se aceptarán tareas que funcionen de manera distinta al ejemplo dado o que no cumplan con las restricciones de este enunciado. ¡Cuidado! Si la salida de su solución difiere en un solo caracter a la salida del ejemplo dado, su tarea será rechazada. Recuerde que suele ocurrir que una tarea funciona correctamente en un computador, pero no funciona en anakena.

Además se descontará medio punto si su tarea entrega warnings al compilar en anakena.

Entrega

Ud. debe entregar el archivo *lifegame.c* por medio de U-cursos antes del viernes 8 de abril a las 23:59. Se descontará medio punto por día de atraso. No se consideran los días Sábado, Domingo o festivos. Resuelva esta tarea antes del control 1, le servirá de estudio.