PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

ALGORITMIA
Laboratorio 2
(Primer semestre de 2013)

Horario 0581: prof. Andrés Melgar

Duración: 3 horas
Nota:

- Se permite el uso de material de consulta.
- Está prohibido el acceso a Internet y al correo electrónico hasta que lo indiquen los jefes de práctica.
- La grabación del trabajo final se efectuará de acuerdo a las indicaciones dadas por los jefes de práctica. SI NO SE SIGUEN LAS INDICACIONES PARA ALOJAR LOS ARCHIVOS EN LA INTRANET, EL ALUMNO SE HARÁ ACREEDOR A LA NOTA 00 (CERO), perdiendo su derecho a reclamo.
- El proyecto en ANSI C deberá ser zipeado en un archivo con el nombre aCodigo.zip y deberá ser colocado en la intranet del curso dentro de la carpeta Laboratorios/LAB2.
- Debe utilizar comentarios para explicar la lógica seguida en el programa elaborado.

Puntaje total: 20 puntos

Cuestionario:

PARTE PRÁCTICA

Pregunta 1 (20 puntos) El objetivo del juego *Sudoku* es asignar dígitos a las celdas vacías de un tablero de $9x9$ (81 celdas), dividido en subtableros de $3x3$ (ver siguiente figura), de forma tal que cada columna, fila y cada subtablero contengan exactamente una instancia de dígitos del 1 al 9. Las celdas iniciales se asignan para restringir el juego de tal manera que sólo exista una manera de terminarlo.

	2						9	
3		1	9		6	5		2
			8		4			
	9						5	
5			2		3			6
	7						2	
			4		7			
8		2	5		1	7		3
	5						8	

Dada una matriz de 9x9 que representa un tablero del juego *Sudoku* con las celdas iniciales ya asignadas, se le pide a Ud. que elabore un programa en ANSI C que usando la estrategia de ***backtracking***

encuentre la solución del juego. A continuación se presenta una posible solución al tablero del ejemplo anterior.

7	2	4	1	3	5	6	9	8
3	8	1	9	7	6	5	4	2
9	6	5	8	2	4	1	3	7
2	9	6	7	1	8	3	5	4
5	1	8	2	4	3	9	7	6
4	7	3	6	5	9	8	2	1
6	3	9	4	8	7	2	1	5
8	4	2	5	9	1	7	6	3
1	5	7	3	6	2	4	8	9

```c
#include <stdio.h>
#include <stdlib.h>
#define N 9
#define NOASIGNADO 0
#define TRUE 1
#define FALSE 0

void imprime_tablero(int [][N]);
int resolver_sudoku(int [][N]);
int EncontrarCasillaNoAsignada(int [][N], int *, int *);
int NoHayConflicto(int [][N], int, int, int);

int main(int argc, char** argv) {
    int t[N][N] = {0, 2, 0, 0, 0, 0, 0, 9, 0,
        3, 0, 1, 9, 0, 6, 5, 0, 2,
        0, 0, 0, 8, 0, 4, 0, 0, 0,
        0, 9, 0, 0, 0, 0, 0, 5, 0,
        5, 0, 0, 2, 0, 3, 0, 0, 6,
        0, 7, 0, 0, 0, 0, 0, 2, 0,
        0, 0, 0, 4, 0, 7, 0, 0, 0,
        8, 0, 2, 5, 0, 1, 7, 0, 3,
        0, 5, 0, 0, 0, 0, 0, 8, 0};

    imprime_tablero(t);
    resolver_sudoku(t);
    imprime_tablero(t);
    return (EXIT_SUCCESS);
}

int resolver_sudoku(int t[][N]) {
    int fil, col, num;
    if (!EncontrarCasillaNoAsignada(t, &fil, &col)) return TRUE; /*exito*/

    for (num = 0; num <= 9; num++) { /*considerar los digitos del 1 al 9*/
        if (NoHayConflicto(t, fil, col, num)) {
            t[fil][col] = num;
            if (resolver_sudoku(t)) return TRUE;
            t[fil][col] = NOASIGNADO;
        }
    }
    return FALSE; /*llama al backtracking*/
}

int EncontrarCasillaNoAsignada(int t[][N], int *fil, int *col) {
```

```
45      for (*fil = 0; *fil < N; (*fil)++)
46          for (*col = 0; *col < N; (*col)++)
47              if (t[*fil][*col] == NOASIGNADO)
48                  return TRUE;
49      return FALSE;
50  }
51
52  int NoHayConflicto(int t[][N], int fil, int col, int num) {
53      return !UsadoEnFila(t, fil, num) && !UsadoEnColumna(t, col, num) &&
54              !UsadoEnSubtablero(t, fil - fil %3, col - col %3, num);
55  }
56
57  int UsadoEnFila(int t[][N], int fil, int num) {
58      int col;
59      for (col = 0; col < N; col++)
60          if (t[fil][col] == num)
61              return TRUE;
62      return FALSE;
63  }
64
65  int UsadoEnColumna(int t[][N], int col, int num) {
66      int fil;
67      for (fil = 0; fil < N; fil++)
68          if (t[fil][col] == num)
69              return TRUE;
70      return FALSE;
71  }
72
73  int UsadoEnSubtablero(int t[][N], int fil_ini_subtablero, int col_ini_subtablero, int num
        ) {
74      int dfil, dcol;
75      for (dfil = 0; dfil < 3; dfil++)
76          for (dcol = 0; dcol < 3; dcol++)
77              if (t[fil_ini_subtablero + dfil][col_ini_subtablero + dcol] == num)
78                  return TRUE;
79      return FALSE;
80  }
81
82  void imprime_tablero(int t[][N]) {
83      int i, j;
84      for (i = 0; i < N; ++i) {
85          if (i %3 == 0)
86              printf("\n");
87          for (j = 0; j < N; ++j) {
88              if (j %3 == 0)
89                  printf(" ");
90              printf("%c", t[i][j] == NOASIGNADO ? '*' : '0' + t[i][j]);
91          }
92          printf("\n");
93      }
94  }
```

Profesores del curso: Andrés Melgar

Pando, 24 de abril de 2013