C. P. S. - Universidad de Zaragoza
Dpto. Informática e Ingeniería de Sistemas

Intr. a la Programación / Fund. Informática

2ª convocatoria 7 - Septiembre - 2006

Examen teórico (80% de la calificación total) *Duración total:* 3 horas

Ejercicio 1 [4 puntos]

El juego del Sudoku se desarrolla a partir de un tablero de 9x9 casillas, algunas de las cuales contienen un número del 1 al 9. El objetivo consiste en completar el tablero con números del 1 al 9 de forma que no se repita ningún número en una fila, ni en una columna, ni en cada una de las 9 regiones de 3x3 en que se divide el tablero. Es decir, en cada una de las filas, columnas y regiones están todos los números del 1 al 9.

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

Un estudiante quiere probar el programa que ha desarrollado para resolver Sudokus con una colección de ejemplos que se ha bajado de Internet. Todos los Sudokus se encuentran en ficheros de texto pero, según su procedencia, presentan diferentes formatos para definir el Sudoku (véanse las figuras con algunos ejemplos de formato). Tras analizarlos todos, llega a la siguiente conclusión:

1) El contenido del Sudoku puede venir en una o varias líneas, según su procedencia, y contiene la información de todas las casillas del tablero ordenadas según un recorrido por filas de arriba a abajo.
2) Sea cual sea el formato, las casillas que están inicialmente ocupadas están representadas por un carácter numérico entre 1 y 9, y las que están vacías por un 0 ó por un punto. El resto de la información que aparece en el fichero de texto no es útil, por lo que no se trata.

*formato *.ss*

```
1..|9.7|..3
.8.|...|.7.
..9|...|6..
-----------
..7|2.9|4..
41.|...|.95
..8|5.4|3..
-----------
..3|...|7..
.5.|...|.4.
2..|8.6|..9
```

Para implementar el juego decide representar las casillas con un valor numérico del 0 al 9, donde el 0 representa que la casilla está vacía (todavía no se ha fijado su valor), junto con la información de qué números del 1 al 9 pueden o no estar en esa casilla (es decir, para cada número guarda la indicación de si puede, o no, estar). La idea del programa consistiría en repetir la aplicación de sencillas reglas lógicas para eliminar números posibles de las casillas, hasta que en alguna casilla sólo quede un número posible y se pueda fijar su valor.

*formato *.ssv*

```
100907003080000070009000600007209400410000095008504300003000700050000040200806009
```

Tarea: Desarrollar los siguientes puntos:

1.- Definir las constantes, tipos y estructuras de datos necesarias para representar un tablero de Sudoku con toda la información indicada en el enunciado.

2.- Desarrollar los procedimientos:

procedure `CargarTablero(?? nomFich: tpNomFich; ?? tbl: tpTablero);`
{Devuelve en `tbl` *el sudoku definido en el fichero de texto de nombre* `nomFich`*. Además, para cada número leído, lo elimina como posible del resto de las casillas de la fila, columna y región a que pertenece }*

procedure `fijarValor(?? f, c: tpIndice; ?? num: tpNum;?? tbl: tpTablero);`
{coloca `num` *en la casilla de fila* `f` *y columna* `c` *del tablero* `tbl`*, y lo elimina como posible en la fila, columna y región de 3x3 a la que pertenece la casilla}*

Notas: 1) Se supone que el fichero con la definición inicial del Sudoku no contiene otros números ni puntos '.' que los que definen las casillas, aunque sí puede haber entre ellos cualquier otro caracter.

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

solución al sudoku anterior

Ejercicio 2 [3 puntos]

Un triángulo de Tartaglia consiste en una serie de filas de números, colocados en forma de triángulo, que verifican las siguientes propiedades:

1) En la fila f-ésima hay f+1 números (la primera fila es la fila 0)
2) El primero y último número de cada fila es un 1
3) El resto de los números de cada fila se obtienen sumando los dos números que tiene situados encima (en la fila anterior).

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
1   5  10  10   5   1
```
resultado de ejecutar Tartaglia(6)

Tarea. Utilizando las propiedades anteriores, se pide:

1.- Tras definir las constantes y tipos de datos necesarios, desarrollar los procedimientos:

```
procedure siguienteFila(¿? f : integer; ¿? fila : tpFila);
```
{ devuelve en `fila` *los números de la fila (*`f`*+1)-ésima, a partir de los números de la fila* `f`*-ésima que hay en* `fila`*}*

```
procedure Tartaglia(?? n : integer);
```
{visualiza, aproximadamente centrado en pantalla, el triángulo de Tartaglia de `n` *filas}*

2.- Sabiendo que el valor del número c-ésimo de la fila f-ésima coincide con el número combinatorio $C_{f,c} = \binom{f}{c}$ desarrollar la función `combinatorio`, que devuelva el número combinatorio de los dos valores suministrados como argumentos, utilizando el método descrito. En el ejemplo, $C_{4,2}=6$ y $C_{5,3}=10$ (obsérvese que se empieza a contar en 0)

Nota: Se puede suponer que el ancho de la pantalla es 80 caracteres, y que el desplazamiento de la cifra de las unidades de un número al siguiente es de 4 espacios (el número más grande es de 3 dígitos).

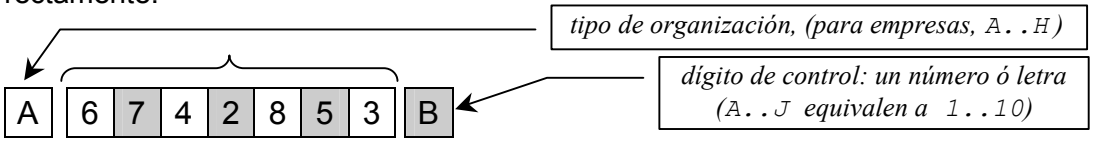
Ejercicio 3 [3 puntos]

El CIF (Código de Identificación Fiscal) es un elemento de identificación administrativa para organizaciones que consta siempre de 9 caracteres. El primero de ellos es una letra que sirve para identificar el tipo de organización (en el caso de una empresa es un letra de la A a la H), que va seguida de 7 dígitos decimales. El último carácter, o de control, se utiliza para detectar errores en el número formado por los dígitos anteriores (para las empresas españolas es un dígito, y para las extranjeras una letra de la A a la J).

La verificación se realiza del siguiente modo:

1) Se suman todos los dígitos que están en posición par (7, 2, 5 y B en el ejemplo). Si el carácter de control es una letra, se sustituye previamente por su valor numérico (A→1, B→2, ..., J→10) Obsérvese que el carácter de control siempre está en posición par.
2) Se suman todos los dígitos que están en posición impar multiplicados por 2 (6, 4, 8 y 3 en el ejemplo), pero si el resultado de multiplicar el dígito por 2 es mayor que 9, se suman sus dígitos.
3) Si el dígito de las unidades de la suma de las cantidades anteriores es distinto de 0, el CIF está construido incorrectamente.

Ejemplo:



Suma de posiciones impares:

```
6 * 2 = 12 -> 1 + 2 = 3
4 * 2 =  8 ->       = 8
8 * 2 = 16 -> 1 + 6 = 7
3 * 2 =  6 ->       = 6
```

S2 = 3 + 8 + 7 + 6 = 24

Suma de posiciones pares:

S1 = 7 + 2 + 5 + 2 = 16

Suma total: S1 + S2 = 16 + 24 = 40 ⇒ 40 *mod* 10 = 0
⇒ CIF empresa válido

Se pide escribir un programa en Pascal que lea del teclado una secuencia de caracteres y, haciendo uso de las reglas anteriores, muestre por pantalla si es, o no, un CIF válido de una empresa.

Observación: se deben procesar los caracteres a medida que son leídos (no se deberán utilizar estructuras de datos de tipo array o string).