Departamento de Informática e Ingeniería de Sistemas
Universidad Zaragoza

Examen de Teoría Soluciones

12 de enero de 2024

Fundamentos de Informática
Grado en Tecnologías Industriales
Duración: 2h 45m

Ejercicio 1 **[3.0 puntos]**

El **alfabeto Braille** es un sistema de lectura y escritura para personas ciegas. Cada carácter estándar se sustituye por una *celda* de 3 filas x 2 columnas de puntos, algunos de los cuales pueden estar en relieve, lo que permite reconocer el carácter correspondiente por el tacto. Existen dispositivos electro–mecánicos que permiten que un computador muestre caracteres en alfabeto Braille. En la Figura 1 puedes ver la representación Braille de los caracteres ASCII básicos, y uno de esos dispositivos. En este ejercicio vas a implementar un sencillo traductor de ASCII a Braille (limitado a las letras minúsculas del alfabeto), de forma que permita traducir una única palabra.

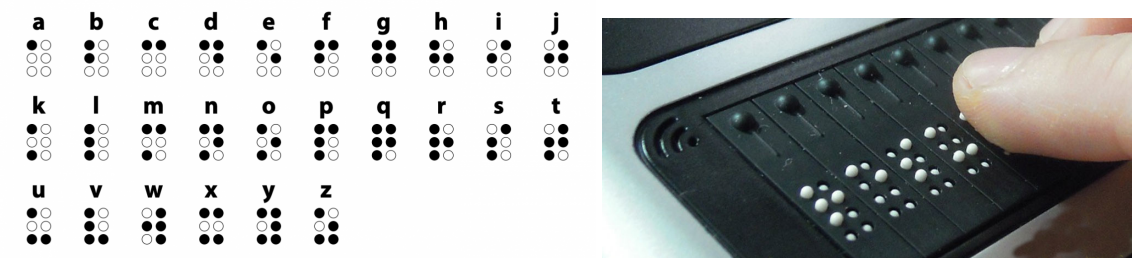

Figura 1: Alfabeto Braille (izquierda) y dispositivo de representación (derecha).

A partir de esta información, **define** en Pascal los siguientes tipos de datos:

- Un tipo **`tpBrailleChar`**, que almacene la información necesaria para representar una *celda* del alfabeto Braille (esto es, un carácter en alfabeto Braille), definiendo qué puntos están en relieve y cuales no.
- Un tipo **`tpBrailleAlphabet`**, que almacene la traducción de cada carácter ASCII a su correspondiente carácter Braille (este último representado mediante una *celda*, de tipo `tpBrailleChar`).
- Un tipo **`tpBrailleString`**, que permita almacenar una cadena de caracteres en alfabeto Braille (es decir, una secuencia de *celdas*, de tipo `tpBrailleChar`). El tamaño de la cadena es variable con un máximo de 255 caracteres en Braille.

A partir de esos tipos de dato, se pide **implementar** en lenguaje Pascal los siguientes subprogramas:

- Un procedimiento

```
procedure stringToBraille(??? aph: tpBrailleAlphabet;
                          ??? s: string; ??? bs: tpBrailleString);
```

que traduzca una cadena de caracteres ASCII dada (`s`) a una cadena Braille (`bs`), mediante el alfabeto dado (`aph`).
Nota: Recordad que se puede acceder a cada uno de los caracteres de un string `s` mediante indexación, esto es, `s[3]` hace referencia al tercer carácter del string, y `s[i]` al caracter en la posición *i* del string.

(continúa en la siguiente página...)

- Un procedimiento

  ```
  procedure writeBraille(??? bs: tpBrailleString)
  ```

  que muestre en pantalla la cadena de caracteres Braille dada (`bs`), con el formato que se muestra a continuación.
  Por ejemplo, para la palabra `'esternocleidomastoideo'`, la salida seria la siguiente:

  ```
  |O_|_O|_O|O_|O_|OO|O_|OO|O_|O_|_O|OO|O_|OO|O_|_O|_O|O_|_O|OO|O_|O_|
  |_O|O_|OO|_O|OO|_O|_O|__|O_|_O|O_|_O|_O|__|__|O_|OO|_O|O_|_O|_O|_O|
  |__|O_|O_|__|O_|O_|O_|__|O_|__|__|__|O_|O_|__|O_|O_|O_|__|__|__|O_|
  ```

  Los puntos en relieve se representan con el caracter `'O'`, el resto con el caracter `'_'`, y las celdas braille se separan mediante barras verticales `'|'`.

Propuesta de solución:

```
const
  MAXLEN = 255;

type
  tpBrailleChar = array[1..3,1..2] of boolean;

  tpBrailleAlphabet = array['a'..'z'] of tpBrailleChar;

  tpBrailleString = record
    bchars : array[1..MAXLEN] of tpBrailleChar;
    len    : 0..MAXLEN;
  end;

procedure stringToBraille(const aph: tpBrailleAlphabet;
            const s: string; var   bs: tpBrailleString);
var
  i: integer;
begin
  bs.len := length(s);
  for i:=1 to bs.len do
    bs.bchars[i] := aph[s[i]];
end;

procedure writeBraille(const bs: tpBrailleString);
var
  i,f,c : integer;
begin
  for f:=1 to 3 do
  begin
    write('|');
    for i:=1 to bs.len do
    begin
      for c:=1 to 2 do
        if bs.bchars[i][f,c]
        then write('O')
        else write('_');
      write('|');
    end;
    writeln();
  end;
end;
```

Ejercicio 2 **[3.5 puntos]**

En una academia de idiomas se imparten varios cursos de lenguas modernas en cuatro niveles y dos modalidades: grupos **regulares** (que reciben dos horas de clase a la semana durante todo el curso) y grupos **intensivos** (cuyos cursos duran un mes al año y reciben ocho horas de clases todos los días). No todos los alumnos pueden optar al curso intensivo, sino que el número de plazas disponibles se determina después de ver la pre–inscripción inicial, mediante un porcentaje para cada nivel.

La matriz A expresa el número total de alumnos pre–inscritos, según el idioma y nivel que quieren cursar. Cada columna corresponde a un idioma diferente. Las filas, a los niveles primero, segundo, tercero y cuarto respectivamente. El vector B refleja el tanto por uno de estudiantes que pueden optar a la modalidad intensiva para cada nivel. A partir de esa información, deseamos obtener las plazas disponibles para cada idioma, modalidad y nivel, guardando la información en una tercera matriz C, con una fila para cada nivel, y dos columnas por idioma, una para cada modalidad.

Para este ejercicio, **se pide** desarrollar un programa en Pascal que cumpla con los siguientes requerimientos y funcionalidades:

(a) Definir un tipo de dato adecuado para representar cada una de las matrices o vectores que se utilizarán durante el resto del programa (matrices A y C, vector B). Ver ejemplo al final del ejercicio.

Nota: Debido a la plantilla de profesores contratados en la academia, nunca se impartirán más de 10 idiomas de forma simultánea. Así, el máximo número de idiomas es 10, pero pueden ser menos.

(b) Diseñar e implementar un procedimiento que le pregunte al usuario cuántos idiomas se imparten en la academia y lea de teclado la matriz A y el vector B. En el caso de la matriz A, ésta se introducirá por filas, con los valores separados por espacios en blanco. En el caso del vector B, los valores del vector también se introducirán separados por espacios en blanco.

```
procedure leerDatos(??? matrizA: ???; ??? vectorB: ???);
```

(c) Mostrar por pantalla una única matriz que proporcione el número de plazas disponibles por modalidad, idioma y nivel siguiendo el formato de la matriz C. Ver ejemplo al final del ejercicio. Se mostrará sólo el contenido de la matriz C, sin encabezados en filas y columnas.

(d) Sabiendo que a los alumnos se les cobra 30€ por persona en grupos intensivos y 20€ en grupos regulares, ¿cuál es el grupo que le sale más rentable a la academia (con el que más dinero ingresan en términos absolutos)? Muestra por pantalla cuál es el grupo más rentable: nivel (del 1 al 4), modalidad (intensiva o regular) e idioma (número de columna).

Ejemplo: Durante el curso 2023/2024, la academia imparte tres idiomas: inglés, francés y alemán. La matriz A tiene un tamaño de 4 filas (una por cada nivel) por 3 columnas (una por cada idioma). El vector B tiene 4 elementos (uno por cada nivel). Siguiendo estas dimensiones, la matriz C (el desglose de alumnos por modalidad, idioma y nivel) tiene un tamaño de 4 filas (una por nivel) y *seis columnas* (dos por cada idioma, dependiendo de si la modalidad es intensiva o regular). Para saber cuál es la rentabilidad de cada grupo para la academia partiremos de la matriz C, teniendo en cuenta los distintos precios para cada modalidad impartida. Por ejemplo, la academia ingresa 780€ por el grupo intensivo de nivel 1 de inglés (26 alumnos x 30€ por cada alumno) y 2080€ (104 alumnos x 20€ por cada alumno) por el grupo regular del mismo nivel de inglés.

$$A = \begin{array}{c} \\ \text{Nivel 1} \\ \text{Nivel 2} \\ \text{Nivel 3} \\ \text{Nivel 4} \end{array} \begin{array}{c} \begin{array}{ccc} \text{Inglés} & \text{Alemán} & \text{Francés} \end{array} \\ \begin{pmatrix} 130 & 160 & 140 \\ 120 & 120 & 80 \\ 210 & 100 & 130 \\ 100 & 40 & 60 \end{pmatrix} \end{array} \qquad B = \begin{array}{c} \\ \text{Nivel 1} \\ \text{Nivel 2} \\ \text{Nivel 3} \\ \text{Nivel 4} \end{array} \begin{array}{c} \text{Modalidad Intensiva} \\ \begin{pmatrix} 0{,}20 \\ 0{,}25 \\ 0{,}40 \\ 0{,}75 \end{pmatrix} \end{array}$$

$$C = \begin{array}{c} \\ \text{Nivel 1} \\ \text{Nivel 2} \\ \text{Nivel 3} \\ \text{Nivel 4} \end{array} \begin{array}{c} \begin{array}{cccccc} \text{Inglés}_{\text{int}} & \text{Alemán}_{\text{int}} & \text{Francés}_{\text{int}} & \text{Inglés}_{\text{reg}} & \text{Alemán}_{\text{reg}} & \text{Francés}_{\text{reg}} \end{array} \\ \begin{pmatrix} 26 & 32 & 28 & 104 & 128 & 112 \\ 30 & 30 & 20 & 90 & 90 & 60 \\ 84 & 40 & 52 & 126 & 60 & 78 \\ 75 & 30 & 45 & 25 & 10 & 15 \end{pmatrix} \end{array}$$

Observaciones:

1. El vector B es común para todos los idiomas.
2. Siempre hay cuatro niveles para cada idioma.
3. Se puede obtener el tanto por uno de estudiantes en la modalidad regular restando 1 menos cada uno de los elementos del vector B.

Propuesta de solución:

```
program idiomas;

const

maxIdiomas = 10;
niveles = 4;

//apartado a, definir estructuras de datos. (1.5/10)
type
tpMatrizAC = record
  alumnos: array[1..niveles,1..maxIdiomas*2] of integer;
  nIdiomas: 1..maxIdiomas;
end;
tpvectorB = array[1..niveles] of real;

//apartado b, procedimiento leer datos (1.5/10)
//válido así  y leyendo con saltos de línea
//se valora positivamente el control de datos de entrada
procedure leerDatos(var matA:tpMatrizAC; var vectorB:tpVectorB);
var
 n, i, j: integer;
begin
  //pedir n al usuario (write)
  writeln('introduzca el número de idiomas');
  read(n); //leer el número de idiomas
  //punto positivo (+1) para control de entrada
  while (n<0) or (n>10) do begin
    writeln('debe haber entre 1 y 10 idiomas');
    read(n);
  end;
  matA.nIdiomas:= n;
  //pedir matriz A al usuario (write)
  writeln('introduzca la matriz A');
  for i:=1 to niveles do begin
    for j:=1 to n do begin
      read(matA.alumnos[i,j]);
    end;
  end;
  //pedir vector B al usuario (write)
  writeln('introduzca el vector B');
  for i:=1 to niveles do begin
    read(vectorB[i]);
  end;
end;

//apartado c, mostrar matriz C (3/10)
function mostrarC(const matA:tpMatrizAC; const vectorB:tpVectorB): tpMatrizAC;
var
  i,j:integer;
begin
  writeln('------Matriz C------');
  mostrarC.nIdiomas:=matA.nIdiomas;
  for i:=1 to niveles do begin
    //intensivos
    for j:=1 to matA.nIdiomas do begin
      mostrarC.alumnos[i,j]:=trunc(matA.alumnos[i,j]*vectorB[i]);
      write(mostrarC.alumnos[i,j],' ');
    end;
    //regulares
```

```
    for j:= matA.nIdiomas+1 to matA.nIdiomas*2 do begin
      mostrarC.alumnos[i,j]:= trunc(matA.alumnos[i,j-matA.nIdiomas]*(1-vectorB[i]));
      write(mostrarC.alumnos[i,j],' ');
    end;
    writeln();
  end;
end;

//apartado d, mostrar grupo más rentable (4/10)
procedure masRentable(const matC:tpMatrizAC);
var
  i,j, matricula:integer;
  rentabilidad: tpMatrizAC;
  maxI, maxJ, maxV: integer;
begin
  rentabilidad.nIdiomas:=matC.nIdiomas;
  //crear matriz con ganancia neta por grupo
  for i:=1 to niveles do begin
    for j:=1 to matC.nIdiomas*2 do begin
      if j <= matC.nIdiomas then
        matricula:= 30 //grupos intensivos
      else
        matricula:=20; //grupos regulares
      rentabilidad.alumnos[i,j]:=matC.alumnos[i,j]*matricula;
    end;
  end;
  //buscar el máximo en la matriz de rentabilidad
  //inicializar el máximo
  maxV:=rentabilidad.alumnos[1,1];
  maxI:=1;
  maxJ:=1;
  for i:=1 to niveles do begin
    for j:=1 to matC.nIdiomas*2 do begin
      if rentabilidad.alumnos[i,j] > maxV then begin
        maxV:=rentabilidad.alumnos[i,j];
        maxI:=i;
        maxJ:=j;
      end;
    end;
  end;
  //escribir el resultado por pantalla (formato libre)
  write('El grupo más rentable es el del idioma ',maxJ,' nivel ',maxI,' modalidad ');
  if maxJ <= rentabilidad.nIdiomas then
    writeln('intensiva.')
  else
    writeln('exhaustiva.');

end;

//programa principal. Se valora positivamente el uso de subprogramas.
var
  matA, matC: tpmatrizAC;
  vectorB: tpvectorB;
begin
  leerDatos(matA, vectorB);
  matC:=mostrarC(matA, vectorB);
  masRentable(matC);
end.
```

Ejercicio 3 **[3.5 puntos]**

Se quiere hacer un somero análisis de la evolución y fluctuación de dos de las criptomonedas más populares, Bitcoin (BTC) y Ethereum (ETH), en el año 2023. Dicho análisis supone calcular unas estadísticas simples sobre el histórico del valor de cotización de dichas criptomonedas respecto al dólar estadounidense (USD). La cotización de una moneda frente al dólar es el tipo de cambio, esto es, a cuántos dólares equivale un BTC o un ETH.

Se dispone para ello de un **fichero de texto**, de nombre `'cotizaciones.txt'` que contiene la cotización diaria de cada una de las criptomonedas para cada día del año. El formato de dicho fichero es tal que cada línea del fichero contiene tres datos separados por espacios en blanco, en este orden (ver también Fig. 2): el valor de cotización, tres caracteres indicando la criptomoneda (BTC o ETH), y la fecha en formato `'dd-mm-aaaa'`.

En el fichero, las cotizaciones están ordenadas cronológicamente. Sin embargo, para cada día del año, no se sabe a priori cuál de las dos monedas está escrita antes en el fichero (en el ejemplo de la Fig. 2 se ve que para ciertos días está antes BTC, y para otros está antes ETH). El fichero sólo almacena cotizaciones de esas dos criptomonedas, y todos los días del año tienen valor de cotización para ambas monedas.
Se pide implementar en Pascal un programa que:

- Calcule y muestre por pantalla la volatilidad V de cada una de las dos criptomonedas a lo largo de 2023. Para calcular la volatilidad V se necesita la variación relativa diaria en precio, G. La variación diaria en precio se puede calcular para cada día t como:

$$G(t) = \frac{Z(t+1)\text{-}Z(t)}{Z(t)}$$

  donde t es el día actual, $t+1$ el día siguiente, y Z el valor de cotización correspondiente a ese día. Una vez se tiene el valor de $G(t)$ para cada día del año, la volatilidad anual es la media de $G(t)$ a lo largo de un año, esto es:

$$V = \frac{1}{N}\sum_{t=1}^{t=N} abs(G(t))$$

  donde $N = 365$ es el número de días, y $abs(\cdot)$ es la función valor absoluto.

  Nota: Para el cálculo de G del último día del año, dado que no tenemos la cotización del día siguiente ($Z(t+1)$), cogeremos el mismo valor de G que en el penúltimo día, es decir, $G(365) = G(364)$).

- Calcule y muestre por pantalla la *variación relativa interanual* de cada criptomoneda en 2023. Esto no es más que la variación relativa entre el primer y último día del año:

$$G_{IA} = \frac{Z(365)\text{-}Z(1)}{Z(1)}$$

Importante: Se valorará el uso adecuado de subprogramas, y se valorará negativamente la repetición innecesaria de código, así como recorrer múltiples veces el fichero de texto.
Observaciones:

1. Ignora la posibilidad de que el año sea bisiesto, a efectos de la solución todos los años tienen 365 días.
2. La cotización de una criptomoneda frente al dólar nunca es cero.
3. El formato que debe tener la salida por pantalla se puede ver en la Figura 3.

Figura 2: Ejemplo de fichero `'cotizaciones.txt'` (se muestra sólo un fragmento del fichero)

```
17438.70 BTC 01-01-2023
1335.55 ETH 01-01-2023
1388.37 ETH 02-01-2023
17936.70 BTC 02-01-2023
1416.79 ETH 03-01-2023
18847.70 BTC 03-01-2023
19922.90 BTC 04-01-2023
1451.22 ETH 04-01-2023
1551.71 ETH 05-01-2023
20979.70 BTC 05-01-2023
...
```

Figura 3: Ejemplo de salida por pantalla (a efectos de ilustrar el formato de la salida)

```
Volatilidad BTC: 0.048
Volatilidad ETH: 0.045
Interanual BTC: 0.203
Interanual ETH: 0.162
```

Propuesta de solución:

```
program criptomonedas_conVec;

const
   DIASANIO = 365;

type
   tpVecReales = array[1..DIASANIO] of real;

function variacionRelativa(cSig, cAnt: real): real;
begin
   variacionRelativa := (cSig - cAnt) / cAnt;
end;

function volatilidadAnual(v: tpVecReales): real;
var
   vol: real; i: integer;
begin
   vol := 0.0;
   for i:=1 to (DIASANIO-1) do
   begin
      vol := vol + abs(variacionRelativa(v[i+1], v[i]));
   end;
   vol := vol + variacionRelativa(v[DIASANIO], v[DIASANIO-1]);
   vol := vol / DIASANIO;
   volatilidadAnual := vol;
end;

var
   vETH, vBTC: tpVecReales;
   f: text;
   volETH, volBTC: real;
   interanualETH, interanualBTC: real;
   i: integer;
   mon1, mon2, c: char;
   cot1, cot2: real;
begin
   // 0. Leer de fichero y almacenar en vectores correspondientes }
   assign(f, 'cotizaciones.txt');
   reset(f);
   i := 1;
   while not eof(f) do
   begin
      readln(f, cot1, c, mon1);
      readln(f, cot2, c, mon2);
      { Se asume que mon1 siempre sera B o E, si no, con else-if }
      if (mon1 = 'B') then
      begin
         vBTC[i] := cot1;
         vETH[i] := cot2;
      end
      else // if (mon1 = 'E') then
      begin
         vBTC[i] := cot2;
         vETH[i] := cot1;
      end;
      i := i + 1;
   end;
   close(f);

   // 1. Volatilidad
   volETH := volatilidadAnual(vETH);
   volBTC := volatilidadAnual(vBTC);

   // 2. Variacion interanual
   interanualETH := variacionRelativa(vETH[DIASANIO], vETH[1]);
   interanualBTC := variacionRelativa(vBTC[DIASANIO], vBTC[1]);

   writeln('Volatilidad BTC: ', volBTC:1:3);
   writeln('Volatilidad ETH: ', volETH:1:3);

   writeln('Interanual BTC: ', interanualBTC:1:3);
   writeln('Interanual ETH: ', interanualETH:1:3);
 end.
```