NOMBRE.....

NÚM. de MATRÍCULA..... GRUPO.....

Examen Final. Informática. Septiembre de 2007

Instrucciones

- El examen consta de **diez** preguntas, cada una de las cuales puntuará **cero** o **un** punto.
- Se calificará con **un punto** si la respuesta correcta se indica en la forma **más simple**.
- La duración total del examen será de **dos horas y media**

---

1. Un alumno dispone de un dispositivo de memoria USB (*Universal Serial Bus*, en inglés *pendrive* o *USB flash drive*) con capacidad de almacenamiento de **128 Megabytes** y de otro dispositivo similar de **256 Megabytes**. Indicar la capacidad **total** de almacenamiento de datos de ambos en **bits**. Nota: es suficiente indicar la expresión numérica decimal correspondiente.

   bits

2. Completar el programa `verinif` para que, dado un **número** de Documento Nacional de Identidad (**DNI**) y una **letra** del alfabeto (valores introducidos por teclado durante la ejecución del programa), **calcule** el valor de `ok` e **indique** por pantalla si la letra **es** o **no correcta** para el Número de Identificación Fiscal (**NIF**) correspondiente. *Sugerencia*: El procedimiento de verificación puede consistir en calcular el **módulo** (resto de la división por) **23** del número correspondiente. El resultado da una posición en una secuencia de 23 letras (`TRWAGMYFPDXBNJZSQVHLCKE`) empezando a contar desde el **cero**. La letra situada en dicha posición será la letra **correcta** del NIF. Por ejemplo, a los números `23` ó `6900000` les corresponde la letra `T`, mientras que al número `46000001` le corresponde la letra `R`.

```
program verinif;
```

```
if ok then writeln('El NIF es correcto') else writeln('El NIF es incorrecto')
end.
```

3. Construir una función que devuelva `true` si **todos** los valores de los elementos de un vector de números enteros dado como parámetro de la rutina son **distintos** y `false` en caso contrario. Nota: deben emplearse las declaraciones indicadas a continuación.

```
const n = 100;
type indice = 1..n;
     vector = array [indice] of integer;
```

4. Construir la declaración de un **procedimiento** `adelanta` para que al ejecutarse la llamada a `adelanta(v,c);` se modifique el contenido previo del parámetro real `v` de tipo `vector`, que almacena la referencia horaria (hora y minuto) de varias ciudades del mundo en horas y minutos: los horarios de las ciudades con *tipo de uso horario* **igual** al valor del parámetro `c` han de adelantarse **una hora**. Nota: el valor del campo `hh` debe mantenerse entre 0 y 23 para que sea correcto.

```
const maximo = 50;
type indice = 1..maximo;
     horas = 0..23;
     minutos = 0..59;
     horario = record
        ciudad : string;
        tipoUsoHorario : char;
        hh : horas;
        mm : minutos
        end;
     vector = array[indice] of horario;
```

5. Construir una rutina **recursiva** para calcular y devolver el valor del **coeficiente binomial** definido de forma recurrente mediante la expresión mostrada en la figura de la derecha, considerando $0 \leq k \leq n$.

$$\binom{n}{0} = \binom{n}{n} = 1$$

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

6. Completar la función `ct` para que devuelva el **coste total** de un pedido de productos cuyos datos se almacenan en un archivo de **texto** (cuyo nombre se indica en el parámetro `s` de tipo `string`) con el siguiente formato:

```
2007011  25  35.99<eoln>
2007036  1  1249.00<eoln>
. . .
. . .
. . .
2007084  120  0.75<eof>
```

Donde, en cada línea, el primer valor es el número de **identificación** del producto (siempre de siete cifras), el segundo valor es el **número** de **unidades** y el tercer valor es el **precio** por **unidad**. En el caso anterior, el coste total es 25 x 35.99 + 1 x 1249.00 + . . . + 120 x 0.75

```
function ct(s:string):real;
```

7. Construir una **unidad** que incluya en su parte **pública** una **función** que devuelva una cadena de caracteres con la calificación **final**, '*suspenso*' (x<5), '*aprobado*' (5≤x<7), '*notable*' (7≤x<9) ó '*sobresaliente*' (x≥9), de un alumno dada su calificación numérica **real** (`x` con un valor entre 0 y 10) como parámetro.

8. Dada una estructura de datos dinámicos construida con las siguientes declaraciones:

```
type puntero = ^string;
     vector  = array[1..100] of puntero;
     parrafo = ^vector;
```

Construir un **procedimiento** que modifique **todas** las cadenas almacenadas en la estructura dinámica a la que apunta el parámetro formal `p` de tipo `parrafo`, de forma que convierta en **mayúsculas** el **primer** carácter de la cadena y **el** carácter que se encuentre **inmediatamente** a continuación de un carácter de espacio en **blanco**. Por ejemplo: la cadena `'el libro 2 de Job'` debe convertirse en `'El Libro 2 De Job'`. Notas: Debe comprobarse la existencia de las variables dinámicas antes de tratar de operar con ellas. Si existe, la cadena almacenada contiene al menos una letra. Puede emplearse la función `upcase(c:char)` (que devuelve el carácter en mayúsculas del parámetro c)

9. Dada una estructura de tipo **lista dinámica** construida con las siguientes declaraciones:

```
type ptr = ^elemento;
     elemento = record dato: integer; sig : ptr end;
```

Construir una función que devuelva `true` si **no** hay dos valores en el campo `dato` de elementos **contiguos** que sean **iguales** y `false` en caso contrario. El parámetro formal `p` de la función debe indicar la dirección de memoria del primer elemento de la lista. Si la lista está **vacía** o tiene únicamente **un** elemento la función debe devolver el valor `true`. Nota: se recomienda una función recursiva.

10. Dada una estructura de tipo **lista** dinámica **circular** construir un **procedimiento** que **introduzca** un **nuevo** elemento al **final** de la estructura a cuyo primer elemento apunta el parámetro `p` de tipo `ptr`, si **no existe** ningún elemento en la lista con el valor **`0`** en el campo `dato`. El valor del campo `dato` del nuevo elemento ha de tomar el valor **`0`**. Nota: pueden emplearse las declaraciones de tipos del ejercicio anterior y debe considerarse el caso de lista vacía