NOMBRE.....

NÚM. de MATRÍCULA..... GRUPO.....

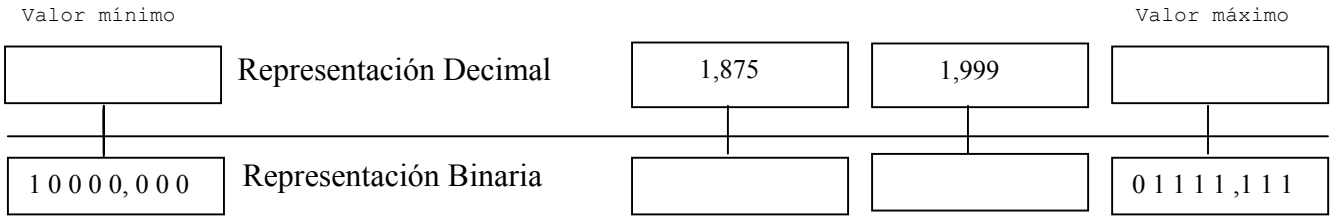
Examen Final. Informática. Febrero 2006

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. En una representación en formato de **punto fijo y complemento a 2** con 8 bits (5 bits para la parte entera y 3 para la parte no entera), indicar el valor mínimo y máximo de los números representados y completar el siguiente diagrama. ¿Cuántas representaciones binarias **distintas** podemos realizar entre los números decimales 1,875 y 1,999 sin contar éstas dos?



Número de **representaciones binarias** diferentes entre 1,875 y 1,999 en el formato anterior: ____

2. Declarar e implementar el procedimiento `digdis`. Dado un número entero positivo `n (longint)`, el procedimiento debe calcular y devolver como resultado el número de **dígitos distintos** que contiene `n`. Por ejemplo, si el número `n` es 0, 5, ó 333 el número de dígitos es 1, si `n` es 21537 el número de dígitos distintos de `n` es 5.

```
Program Numero_digitos;
TYPE Digitos = Array[0..9] of Byte;
```

```
Var n: LongInt; k: Byte;
BEGIN
Readln(n); digdis(n,k);
Writeln('El numero de dígitos distintos de n es  ',k)
END.
```

3. Declarar e implementar un **procedimiento**, `suma`, que devuelva como resultado la suma de otras dos matrices rectangulares dadas como parámetros. El procedimiento **NO** puede utilizar sentencias `read` o `write`.

```
CONST N=8; M = 12;
TYPE Matriz = array[1..N,1..M] of Real;
```

4. Completar la función **booleana** `dentro`. Dado un rectángulo y un punto, la función devuelve verdadero si un punto está **estrictamente** dentro del rectángulo y falso en caso contrario. Los lados del rectángulo son paralelos a los ejes de coordenadas. Un rectángulo se define por su esquina superior izquierda y su esquina inferior derecha.

```
TYPE Punto = Array['x'..'y'] of Real;
     Rectangulo = record TL, BR: punto; end;
Function dentro(p: Punto; rect: Rectangulo): Boolean;
Begin
```

```
End;
```

5. Declarar e implementar una rutina que devuelva la suma del valor numérico de los caracteres correspondientes a dígitos decimales que se encuentren en una cadena alfanumérica. Por ejemplo, si la cadena es 'ACB12m4XyZ' entonces la rutina debe devolver el valor numérico entero 7.

6. Completar el procedimiento, `aleatoria`, para que **asigne** un valor aleatorio 0, 1 ó 2 a las componentes de una matriz N x N, `mat`. La probabilidad de que el valor de cada una de las componentes sea 0 deber ser el 50%, para 1 el 30% y para 2 el 20%. Por ejemplo:

0	0	0	1	1	1
0	1	0	2	0	2
2	0	2	0	1	0
2	2	0	1	0	1
0	1	0	0	2	1
0	2	0	0	1	1

```
CONST N=6; TYPE Matriz = array[1..N,1..N] of Byte;
Procedure aleatoria(var mat: Matriz);
```

7. Dado un archivo de enteros donde se almacena una sucesión de enteros, completar el procedimiento, `escribir_suma`, para que lea ese archivo y cree otro archivo de texto donde se guardan todas las sumas de los primeros elementos hasta el leído. El programa, que se da a continuación, debe leer correctamente el archivo de texto y mostrarlo por pantalla. Por ejemplo, si el archivo de enteros contiene: 3 6 8 6 1 1 4, el programa deberá escribir: "3 9 17 23 24 25 29 ". Si tiene 4 5 2 debe escribir "4 9 11 ". Si el archivo de enteros está vacío se debe crear un archivo de texto vacío.

```
TYPE Enteros = File of Integer;
Procedure escribir_suma(Var f: Enteros; Var t: Text);
```

```
VAR s1,s2: String; f: Enteros; t: Text; num: Integer;
BEGIN
readln(s1);readln(s2);assign(f,s1);assign(t,s2);
escribir_suma(f,t);
reset(t);
while not seekeof(t) do Begin read(t,num); write(num,' '); End;
close(t);
END.
```

8. Dada una lista simple enlazada completar el procedimiento, `eliminar_repetidos`, para que la modifique **eliminando** los elementos que son **iguales al primero**. Se supone que la lista no está vacía. Por ejemplo, si la lista original es: 8 2 8 6 0 5 6 8, la lista final deberá ser: 8 2 6 0 5 6.

```
TYPE Lista=^Elemento; Elemento = record Dato: Byte; sig: Lista; end;
Procedure eliminar_repetidos(p: Lista);
```

9. Dado un vector de punteros que apuntan a una cadena alfanumérica, completar el procedimiento **recursivo**, `mostrar`, para que muestre por pantalla las cadenas. El procedimiento debe mostrar las cadenas (una por línea) empezando por la correspondiente al índice `i` y terminando cuando el puntero no apunte a una variable dinámica o `i` sea mayor que `N`. La llamada a `mostrar(v,1)` debe mostrar por pantalla todas las cadenas desde la primera.

```
CONST N = 100;
TYPE Puntero = ^String; Vector = Array [1..N] of Puntero;
Procedure mostrar(v: Vector, i: Integer);
Begin
```

```
End;
```

10. Dada una lista de palabras completar el procedimiento, escribir_palabras, para que las escriba en un archivo de texto. Las palabras se deben separar con espacios en blanco. Las líneas del archivo resultante no pueden superar 80 caracteres, las palabras no se pueden partir. El parámetro `ruta` contiene el nombre del archivo. Si la lista está vacía se debe crear un archivo vacío.

```
TYPE Lista=^Elemento; Elemento=record palabra: String[30]; sig: Lista; end;
Procedure escribir_palabras(p: Lista; ruta: String);
```