NOMBRE.....

NÚM. de MATRÍCULA..... GRUPO.....

Examen Final. Informática. Septiembre 2004

Instrucciones

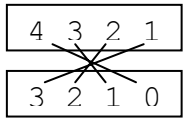
- El examen consta de **diez** preguntas, cada una de las cuales puntuará **cero** o **un** punto.
- Se calificará con **un punto** si la respuesta correcta se indica en la forma **más simple**.
- La duración total del examen será de **dos horas y media**

---

1. Dados dos números codificados en binario punto flotante con 8 bits para la mantisa y cuatro bits para el exponente, ambos en complemento a 2 (C2), representar dicho número en las siguientes codificaciones: punto fijo en complemento a 2, entero en complemento a 2 y en sistema decimal. Puede utilizarse el número de bits que se considere más adecuado para cada codificación.

Punto flotante en C2	Punto fijo C2	Entero C2	Decimal
01110100 0101			
11110100 0101			

2. Dados dos vectores `v1` y `v2` con `n` componentes cada uno, completar el programa para que el resultado `r` acumule el **producto cruzado** de ambos vectores. Se entiende por producto cruzado la suma de los productos del primer elemento de `v1` por el último de `v2`, del segundo de `v1` por el penúltimo de `v2` y sucesivamente.



`r = 4·0 + 3·1 + 2·2 + 1·3 = 10`

```
program cruzado;
var v1,v2:array[1..100] of integer; i,r,n:integer;
begin
randomize;
read(n);
for i:=1 to n do v1[i]:=random(10);
for i:=1 to n do v2[i]:=random(10);
```


```
write('el resultado del producto cruzado es',r);
end.
```

3. Dado un vector `v` de `n` elementos enteros ordenados de menor a mayor, **completar** el procedimiento **`cambiar`** para que cambie el orden del vector `v`, dado como argumento, ordenándolo de mayor a menor.

   Ejemplo: si `v` es [0,2,2,3,7,7,8,9] después del cambio de orden será : [9,8,7,7,3,2,2,0]

```
program ejer3;
const n=8;
type vector=array[1..n] of integer;
procedure cambiar(var v:vector; n:integer);
var i:integer;
begin
```


```
end;
```

4. Dada la matriz `m` creada en el siguiente programa, completar el mismo para que muestre por pantalla la matriz originalmente creada, seguidamente la trasponga almacenándola en la propia variable y a continuación se muestre de nuevo por

Ejemplo de trasposición:

1 0 2 **3 5 6** **4 1 2**	**=>**	**1 3 4** **0 5 1** **2 6 2**

```
program ejer4;
const n=3;
type matriz=array[1..n,1..n] of integer;
var m:matriz; i,j,k:integer;
begin
randomize;
for i:=1 to n do for j:=1 to n do m[i,j]:=random(10);
```

```
end.
```

5. Se considera un archivo de enteros que representa una matriz cuadrada, el primer registro es la dimensión de la matriz y los siguientes son los elementos ordenados por filas. Completar el procedimiento **`trasponer`** cuya finalidad es **crear un nuevo archivo** cuyo nombre se forma a partir del nombre del archivo original, dado a través del argumento `mor`, y cuyo contenido sea la traspuesta de la matriz contenida en el archivo original con la misma organización de los registros (dimensión seguida de los elementos por filas). La dimensión y elementos de la matriz original pueden tomar cualquier valor.

Ejemplo:

1 0 2 **3 5 6** **4 1 2**	**=>**	**1 3 4** **0 5 1** **2 6 2**

Si archivo original:

3	1	0	2	3	5	6	4	1	2

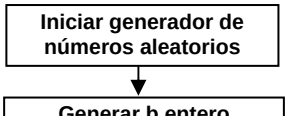
Archivo destino:

3	1	3	4	0	5	1	2	6	2

```
type ts=string[3];
procedure trasponer(mor:ts);
var f,g:file of integer; i,j,k,n:integer;
begin
assign(f,mor); reset(f); read(f,n);
assign(g,'mtr'+mor);
```

```
end;
```

6. Completar la función `aleatorio` que admite un parámetro `a` entero positivo o cero y cuyo comportamiento viene dado por el diagrama de flujo de la figura.



```
function aleatorio(a:word):word;
```

7. Completar la función `leer` cuya finalidad es leer **los registros a partir de la posición actual del cursor**, del archivo dado como argumento, el cual se considera previamente abierto. Los datos leídos son alojados en el vector `b` (buffer). La función devuelve el número de registros leídos, pudiendo no llenar el buffer si no quedan suficientes registros, y siendo cero cuando el cursor está al final del archivo en el momento de la llamada o el archivo está vacío.

```
type archivo = file of real;
     buffer = array[1..100] of real;

function leer(var f:archivo; var b:buffer):word;
```

8. Completar el procedimiento `copia` que realiza la copia **exacta** del archivo cuyo nombre viene dado por el parámetro `entrada` al archivo cuyo nombre viene dado por el parámetro `salida`. El archivo destino si existiera previamente será reescrito. El archivo origen es un archivo con tipo real. Se han de utilizar **obligatoriamente** los tipos de datos y la función `leer` del ejercicio anterior.

```
procedure copia(entrada:string; salida:string);
```

9. Completar la función `compara` que devuelva –1, 0 ó +1 si la duración del vídeo 1 dado por el parámetro `v1` es menor, igual o mayor respectivamente que el vídeo 2 dado por el parámetro `v2`. Nota: no se necesita el uso del puntero `sig` en este ejercicio.

```
type ptrvideo = ^video;
     video = record
                 horas:integer; minutos:integer;
                 sig:ptrvideo;
                 end;

function compara(v1:video; v2:video):integer;
```

10. Completar la función `maximo` que devuelve `true` si el vídeo dado como el argumento `v` tiene una duración **mayor estrictamente** que todos los vídeos de la lista dinámica simple dada por el parámetro `prim`. Devolver `false` en caso contrario. La lista puede ser nula, en ese caso devolver `true`. Es **obligatorio** usar la función `compara` del ejercicio anterior.

```
function maximo(prim:ptrvideo; v:video):boolean;
```