

## Instituto de Computación

### Curso de Programación 1 - Práctico 7

#### 1. Dadas las declaraciones:

```

CONST
  N = 500;
VAR
  puntos: ARRAY [1..N] OF Integer;
  prueba, menor, indice: Integer;

```

¿Cuál segmento de programa encontrará el valor más pequeño de este arreglo, y almacenará el subíndice del elemento donde está guardado este valor?

- a. 

```
FOR prueba := 1 TO N DO
  IF (puntos[prueba] < menor) THEN
    menor := puntos [menor];
```
- b. 

```
menor := puntos[1];
FOR prueba := 2 TO N DO
  IF (puntos[prueba] < menor) THEN
    menor := puntos [prueba];
```
- c. 

```
indice := 1;
FOR prueba := 2 TO N DO
  IF (puntos[prueba] < puntos[indice]) THEN
    indice := prueba;
```
- d. 

```
FOR prueba := 1 TO N DO
  IF (puntos[prueba] < menor) THEN
    menor := prueba;
```
- e. 

```
indice := 1;
FOR prueba := 2 TO N DO
  IF (puntos[prueba] < índice) THEN
    indice := prueba;
```

#### 2. ¿Cuáles de los siguientes segmentos de programa invertirán un arreglo de caracteres llamado *cadena* que consta de N elementos?

- a. 

```
FOR conmuta := 1 TO N DO BEGIN
  temp := cadena[conmuta];
  cadena[(N+1) - conmuta] := temp
END
```
- b. 

```
FOR conmuta := 1 TO N DO BEGIN
  temp := cadena[conmuta];
  cadena[conmuta] := cadena[(N+1) - conmuta];
  cadena[(N+1) - conmuta] := temp
END
```
- c. 

```
FOR conmuta := 1 TO (N DIV 2) DO BEGIN
  temp := cadena[conmuta];
  cadena[conmuta] := cadena[(N+1) - conmuta];
  cadena[(N+1) - conmuta] := temp
END
```
- d. 

```
FOR conmuta := 1 TO (N DIV 2) DO BEGIN
  temp := cadena[conmuta];
```

```

        cadena[conmuta] := cadena[(N+1) - conmuta];
        cadena[conmuta] := temp
    END

```

```

e. FOR conmuta := 1 TO (N DIV 2) DO Begin
    temp := cadena[conmuta];
    cadena[conmuta] := cadena[(N DIV 2) - conmuta];
    cadena [(N DIV 2) - conmuta] := temp
END

```

### 3. Dadas las siguientes declaraciones:

```

CONST N = ...;
...
TYPE
    RangoN = 1..N;
    Tabla = Array[RangoN] OF Integer;

```

- Escriba un programa *HallaIndMax* que imprima el índice del valor más grande de un arreglo.
- Escriba un programa *HallaMax* que imprima el valor más grande de un arreglo.
- Escriba un programa *Ordenado* que imprima una salida indicando si los elementos del arreglo se encuentran en orden ascendente.

En todos los casos se deben obtener los valores del arreglo desde la entrada estándar.

### 4. Examine la siguiente declaración :

```

TYPE
    Tabla = Array [0..9, 1..9] OF Boolean;
VAR
    matriz : Tabla;

```

- ¿ Cuántos componentes contiene esta tabla ?
- ¿ Cuáles de las siguientes proposiciones de asignación son válidas ?
  - matriz [9,9] := 0
  - matriz [0,1] := matriz [9,0] AND matriz [7,7]
  - matriz [1,1] := 1 < 1

### 5. Escriba un programa en Pascal llamado *Cambio* que lea de la entrada estándar una *matriz* (arreglo bidimensional de enteros de diez renglones y diez columnas) y dos variables enteras *m* y *n*. El programa *Cambio* intercambia los renglones *m* y *n* de *matriz*. Incluya las definiciones de tipo necesarias. El resultado se debe imprimir en la salida estándar.

### 6. Escriba un programa *flipflop* en Pascal que dada la matriz de booleanos de tipo:

```

CONST
    M = ...;
    N = ...;
    P = ...;
TYPE
    RangoM = 1..M;
    RangoN = 1..N;
    RangoP = 1..P;
    MatrizBool = ARRAY[RangoM,RangoN,RangoP] OF Boolean;

```

cambie todos los *true* a *false* y viceversa, en cada uno de los componentes de la matriz *lógica*. Los valores de la matriz se leen desde la entrada estándar. El resultado se debe imprimir en la

salida estándar.

7. Dada la definición de tipo para representar cadenas de caracteres de largo N y M:

```
CONST N = ...;
CONST M = ...; { M < N }
...
TYPE
    CadenaN = ARRAY[1..N] Of Char;
    CadenaM = ARRAY[1..M] Of Char;
```

Implementar un programa que lea dos cadenas de la entrada estándar de largo M y N respectivamente, y determine si la primer cadena ocurre como parte de la segunda cadena. El programa debe funcionar para cualquier valor positivo que puedan tomar M y N.

Ejemplo de entrada para N=6, M=3:

```
tor
totora
```

Ejemplo de salida:

El texto 'tor' se encuentra dentro del texto 'totora'.

Ejemplo de entrada:

```
tos
totora
```

Ejemplo de salida:

El texto 'tos' no se encuentra dentro del texto 'totora'.

8. La transpuesta de un arreglo bidimensional cuadrado  $a$  es un arreglo  $b$  del mismo tipo cuyos componentes satisfacen la relación  $b[i,j] = a[j,i]$  para todos los valores posibles de  $i$  y  $j$ . Escriba un programa que calcule la transpuesta de un arreglo de números reales con cinco renglones y cinco columnas. Los valores de la matriz se leen desde la entrada estándar. El resultado se debe imprimir en la salida estándar. Obs: Intente hacerlo sobre la misma matriz.

9. Dadas las siguientes declaraciones:

```
CONST
    M = ...;
    N = ...;
TYPE
    RangoM = 1..M;
    RangoN = 1..N;

    MatrizMxN = ARRAY[RangoM,RangoN] OF Integer;
    MatrizMxM = ARRAY[RangoM,RangoM] OF Integer;
```

Escribir un programa *ProdMatriz* que calcule el producto de una matriz  $M \times N$  con una matriz  $M \times M$ . Las matrices se inicializan con valores leídos de la entrada estándar. El resultado se debe imprimir en la salida estándar.

10. Dada la definición de tipo para representar cadenas de caracteres de largo N y M:

```
CONST N = ...;
CONST M = ...;
...
TYPE
    CadenaN = ARRAY[1..N] Of Char;
```

```
CadenaM = ARRAY[1..M] Of Char;
```

- a. Escriba un programa que lea dos cadenas de largo  $M$  y  $N$  respectivamente, e imprima un mensaje en la salida estándar indicando si alguna letra en la primer palabra ocurre en la segunda.
- b. Escriba un programa que lea dos cadenas de largo  $M$  y  $N$  respectivamente, y determine si todas las letras en la primer palabra ocurren en la segunda.

---

In.Co. - Curso de Programación 1