



# INTRODUCCIÓN A LA PROGRAMACIÓN

TEORÍA (11/02/2002)

NOMBRE:

DNI:

ESPECIALIDAD:

Responda adecuadamente a los siguientes apartados. **Si se aciertan menos de 8 cuestiones propuestas, no se corregirá la parte práctica del examen** (Tiempo de realización: 15 minutos. Puntos: 1.5)

**A.-** Conteste verdadero(V) o falso(F) a las cuestiones que aparecen a continuación :

A.1.-Los operadores de relación permiten comparar expresiones de distinto tipo, siendo el resultado de tipo booleano.

A.2.-Los compiladores traducen instrucción tras instrucción, ejecutándose cada instrucción tras su traducción.

A.3.-Se pueden leer desde teclado valores de tipo enumerado.

A.4.-En la instrucción iterativa repetir, si la condición es falsa al principio no se ejecuta el cuerpo del bucle.

A.5.-En tiempo de ejecución se substituirán las llamadas a las funciones por el valor devuelto por ellas.

A.6.-Cualquier operación que se realice sobre una tabla ha de estar referenciada sobre las variables indexadas y no sobre la tabla completa.

A.7.-Los parámetros por referencia son aquellos en los que los cambios en el valor del parámetro formal afectan al del parámetro real.

A.8.- En las acciones todos los parámetros son de entrada, excepto uno que es de salida.

**B.-** Realice correctamente las siguientes declaraciones :

B.1.-Declare una tabla A tridimensional de  $30 \times 40 \times 50$  de tipo entero.

B.2.- Declare una tabla B bidimensional de  $40 \times 10$  del mismo tipo que la tabla previa A

B.3.-Declare la cabecera de una acción *calcular* que tome como parámetros de entrada una tabla de mismo tipo que A y otra del mismo tipo que B y devuelva como salida una tabla del tipo de A.

B.4.-Declare la cabecera de una función *cuenta* en la que dada una entrada del tipo de B devuelva un entero.



## INTRODUCCIÓN A LA PROGRAMACIÓN

### PROBLEMAS (11/02/2002)

**Duración: 2 horas.**

**Dispone de 45 min. para realizar preguntas sobre el enunciado.**

**Cada problema deberá resolverlo en hojas separadas.**

1.-**(3,5 puntos)** Suponga que tiene las 48 cartas de una baraja desde el 1 al 12 de cada palo dispuestas en una cola, además tendremos 4 pilas(vacías) una por cada palo de la baraja .**Diseñe un algoritmo** que haciendo uso de los módulos que vienen a continuación simule el juego del solitario de la siguiente forma:

Se sacará una carta de la baraja y si es oros se pondrá en la pila de oros, pero siempre en orden del 1 al 12, es decir que primero se pondrá la carta del 1, después la carta del dos...

En el caso de que no se pudiera poner en la pila que corresponde a su palo se volvería a guardar en la cola de la baraja.

Todo este proceso se repite hasta que no quede ninguna carta en la baraja.

Se tienen disponibles los siguientes módulos:

Modulo Cartas;

exporta tipo carta;

(\* información sobre una carta de la baraja \*)

función numero ( c:carta ): entero;

(\* dada una carta c devolverá el numero de la misma, 1,2,3,4,5,6,7,8,9,10,11 o 12\*)

función palo ( c:carta ): carácter;

(\* dada una carta c devolverá el palo que corresponde a la carta, O = oros, C = copas, E=espadas, B=bastos\*)

fexporta

fmodulo

Modulo pilas;

Importa Cartas Fimporta

Exporta tipo pila;

(\* para guardar cartas\*)

acción inicializarp( var p: pila);

(\* antes de poder sacar o meter cartas en p es obligatoria la llamada a esta acción , pone la pila p como si estuviera vacía\*)

función vaciap (p:pila): booleano;

(\* devolverá cierto si la pila p está vacía, en caso contrario devolverá falso\*)

acción sacar ( Var p: pila; var c: carta);

(\*Pondrá en c la carta que lleva menos tiempo almacenada en p\*)

acción meterp ( var p: pila; c:carta);

(\*guardará en la pila p la carta c\*)

función topep ( p: pila ): carta;

(\* devolverá la carta que lleva menos tiempo en la pila p, pero sin quitarla de p\*)

Fexporta

Fmódulo

Modulo colas;

importa Cartas fimporta

exporta tipo cola; (\* para guardar cartas\*)

acción inicializarc( var col : cola);

(\* antes de poder sacar o meter cartas en col es obligatoria la llamada a esta acción , pone la cola col como si estuviera vacía\*)

función vaciac (col:cola): booleano;

(\* devolverá cierto si la cola col está vacía, en caso contrario devolverá falso\*)

acción sacarc ( var col: cola; var c: carta);

(\*Pondrá en c la carta que lleva más tiempo almacenada en col\*)

acción meterc ( var col: cola; c:carta);

(\* guardará en la cola col la carta c\*)

función topec ( col: cola): carta;

(\* devolverá la carta que lleva más tiempo en la cola col, pero sin quitarla de col\*)

acción cargar\_cola ( var col:cola);

(\* cargará la cola col con todas las cartas de la baraja de forma aleatoria\*)

Fexporta

Fmodulo

2.-(2 puntos). Suponga que tiene almacenada en una tabla una serie de números enteros. Diseñe una acción que permita calcular el mínimo número entero almacenado en una tabla,

así como el número de veces que aparece dicho mínimo. La acción a implementar tendrá la siguiente cabecera:

Acción minimo(t: datos; n:entero; var num:entero; var nveces: entero);

(\* guarda en num el valor mínimo existente entre las casillas num y n (ambas inclusive) de la tabla t y guarda en nveces el número de veces que aparece dicho mínimo entre esas celdas \*)

(\*tipo datos=tabla[1..50] de entero;\*)

Ej: Supongamos que la tabla t tiene los siguientes elementos

20	7	1	9	22	7	9	7	14	7	3	7
1	2	3	4	5	6	7	8	9	10	11	12

Valor mínimo: 7

Veces que aparece: 3

Para calcular el mínimo valor que hay entre las casillas 4 y 10 la llamada a realizar sería la siguiente:

m:=4;

minimo(t,10,m,nveces);

Una vez ejecutado m vale 7 y nveces vale 3

3.-(1.5 puntos) Implementar la acción “**Rellenar**” de manera que dada una tabla de 40 enteros, la rellene con la serie de números que se puede obtener a partir de la siguiente expresión:

$$T[n] = \begin{cases} T[n-1] + T[n-2] + n & \text{si } n > 2 \\ 2 & \text{si } n = 2 \\ 1 & \text{si } n = 1 \end{cases}$$

Un ejemplo de tabla (con 10 enteros) es el siguiente:

1	2	3	4	5	6	7	8	9	10
1	2	6	10	21	37	65	110	184	304

(1.5 puntos) Construir la función “**Comprobar\_inversa**” de manera que dada una tabla de 40 enteros nos devuelva verdadero si contiene una serie construida mediante la expresión anterior, pero en orden inverso.

Un ejemplo de tabla (con 10 enteros) invertida es el siguiente:

(\*  
**ERROR**\*\*\*\*\*  
 \*\*)

1	2	3	4	5	6	7	8	9	10
304	184	110	65	37	21	10	6	2	1