

Carrera: **E. T. S. I. I.**
Asignatura: **Informática I**
Material auxiliar: Ninguno

Fecha: 7 de Septiembre de 2000
Hora: 11:30 h.
Duración: 2 horas

RELLENE EN ESTA HOJA Y EN LA HOJA DE LECTURA ÓPTICA LOS SIGUIENTES DATOS:

Apellidos:.....Tlfno.:.....
Nombre:.....D.N.I.:.....
Código Carrera: **10** Código Asignatura: **206** Convocatoria: **Septiembre 2ºPP**
Semana: **1ª**
Tipo de Examen: **A**

- El **test** debe ser contestado **en la hoja de lectura óptica**. Sólo una de las cuatro respuestas posibles de cada pregunta es correcta.
- El test es eliminatorio y aporta un 40% de la nota final. Son necesarias 7 respuestas correctas para que se corrija el ejercicio.
- La solución del ejercicio se realizará en el reverso de esta hoja. **No se corregirán hojas auxiliares.**

ENTREGUE ÚNICAMENTE ESTA HOJA Y LA HOJA DE LECTURA ÓPTICA sin grapar

TEST (cada respuesta correcta: 1 punto; respuesta incorrecta o en blanco: 0 puntos)

1. Dado el siguiente fragmento de código:

```
CASE j OF
  enero..junio:INC(j);|
  diciembre..julio:DEC(j);
ELSE
END;
```

- A.- **la variable j tiene que ser de tipo ordinal para ser correcto**
B.- es necesaria alguna sentencia tras ELSE
C.- la sentencia ELSE no se puede usar dentro de una instrucción CASE
D.- los rangos establecidos son incorrectos

2. Para que en Modula-2, la siguiente definición de tipo sea la de una tabla:

```
TYPE TipVect= ARRAY TipInd OF TipElem
```

A.- TipInd debe ser un tipo ordinal definido por el usuario y TipElem de cualquier tipo

B.- TipInd debe ser cualquier tipo predefinido y TipElem RECORD

C.- TipInd debe ser cualquier tipo ordinal y TipElem RECORD

D.- TipInd debe ser RECORD y TipElem de cualquier tipo

3. Dado el siguiente procedimiento:

```
PROCEDURE proc(a,b:INTEGER);
```

```
VAR aux:INTEGER;
```

```
BEGIN
```

```
  aux:=a+b+c;
```

```
  c:=aux;
```

```
END proc;
```

Para que fuese puro:

A.- Todas las variables, incluida aux, deben pasarse por referencia

B.- Es ya un procedimiento puro

C.- bastaría con pasar las variables a y b por referencia

D.- la variable c se debería pasar por referencia

4. Sabiendo que en Modula-2 existe el tipo predefinido BOOLEAN:

```
TYPE BOOLEAN=(FALSE, TRUE)
```

Se puede afirmar:

A.- ORD(FALSE)=0

B.- ORD(FALSE)=1

C.- El tipo BOOLEAN no es un tipo ordinal

D.- Con los tipos predefinidos no se puede utilizar la ORD

5. La compilación segura:

A.- Mejora la reutilización

B.- Produce un programa objeto más eficiente

C.- Tiene como objetivo comprobar la compatibilidad de tipos

D.- Necesita un módulo de definición

6. Dado el siguiente fragmento de código:

```
TYPE t1=RECORD c1, c2:REAL; END;
```

```
  t2=POINTER TO REAL;
```

```
  t3=POINTER TO t1;
```

```
VAR a:t1; b:t2; c:t3;
```

```
...
```

```
NEW(b); NEW(c);
```

La asignación correcta es:

A.- c^.c1:=a.c2;

B.- b:= c^.c1;

C.- a:=c;

D.- c.c1:=a.c2;

7. Dado la siguiente declaración de variables:

```
VAR c1:T1; c2:T2;
```

Después de ejecutar

```
Leer(c1); Leer(c2);
```

las variables c1 y c2 toman los valores “esto” y “aquello” respectivamente.

A la vista del resultado, el procedimiento Leer tendrá como argumento:

A.- (VAR a: ARRAY OF CHAR)

B.- (a:T1,T2)

C.- (VAR a:T1,T2)

D.- (a: ARRAY OF CHAR)

8. Después de ejecutar el siguiente fragmento de código:

```
VAR pt1, pt2, pt3: POINTER TO REAL;
```

```
...
```

```
NEW(pt1); pt1^:=3.0; pt2:=pt1; pt3:=pt2;
```

```
DISPOSE(pt2);
```

A.- Sólo se libera pt3

B.- Se liberan los punteros pt1, pt2 y pt3

C.- Sólo se libera pt3 y pt2

D.- Se produce un error, y no se libera la memoria

9. La definición completa de un tipo opaco que se declara en un módulo de

A.- La declaración también se hace en el módulo de implementación

B.- El módulo principal del programa

C.- El propio módulo de definición

D.- El módulo de implementación correspondiente

10. Dado el siguiente procedimiento:

```
PROCEDURE
```

```
Calcular (VAR A: INTEGER; B: INTEGER) : INTEGER;
```

```
BEGIN
```

```
  A := B MOD 2; B := A * A + 3 * B - 6; RETURN A+B
```

```
END Calcular;
```

El valor de X tras ejecutar X:=4; X:=Calcular(X,X); será:

A.- 4

B.- 0

C.- 6

D.- 12

EJERCICIO DE PROGRAMACIÓN (10 puntos)

Construir un dato encapsulado que sea una tabla de 50 elementos, en la que se almacenen los datos de una persona: nombre, apellido1, apellido2, dirección y teléfono. Las operaciones serán añadir un elemento, eliminar un elemento y un procedimiento de búsqueda selectiva por nombre o por número de teléfono. Si se encuentra la persona en la tabla se devolverá cierto y se mostrará toda la información disponible de esa persona y si no se devolverá falso.

RECUERDE: La solución del ejercicio se realizará en el reverso de esta hoja. NO se corregirá lo que exceda de este espacio.