| FUNDAMENTOS DE PROGRAMACION | MODELO 1 | *CONV. EXTRAORD. 15/16* |
|---|---|---|
| GRADO DE INFORMÁTICA y GRADO DE TECNOLOGÍAS DE LA INFORMACIÓN-ETSII-UNED | | |
| MATERIAL AUTORIZADO: **NINGUNO** | | DURACIÓN: **2 HORAS** |

| INSTRUCCIONES: | **1) El test debe ser contestado en la hoja de marcas. Sólo una de las cuatro respuestas posibles de cada pregunta es correcta.**<br>**2) El test es eliminatorio y aporta un 30% de la nota final. Son necesarias 8 preguntas correctas (6 con las prácticas aprobadas) para que se corrija el ejercicio.**<br>**3) Cada respuesta correcta 1 pto. Respuesta incorrecta, doble o en blanco: 0 pto.**<br>**4) Puede quedarse, si lo desea, con esta hoja de examen.** |
|---|---|

## CUESTIONES DE TEST

1. El tipo carácter (char), lo constituye una tabla que….

A. asocia un número real con cada carácter
B. depende del lenguaje de programación
C. asocia cada carácter con su posición en la tabla
D. depende del paradigma de programación

2. Dado el siguiente fragmento de programa correcto en el lenguaje C ±:

```
algo -> uno = dos
```

Siempre se puede afirmar que:

A. algo es un puntero
B. uno y dos son punteros
C. algo, uno y dos son punteros
D. uno es un puntero

3. El tratamiento de excepciones asegura que un programa…

A. es más robusto
B. tiene menor complejidad algorítmica
C. se verifica de manera más simple
D. es menos claro

4.- Cuando la velocidad de ejecución de un programa es un factor crítico, debe usarse:

A. Un algoritmo con un orden de complejidad alto
B. Un intérprete
C. Un algoritmo con un orden de complejidad bajo
D. Una máquina virtual

5.- La ejecución del siguiente fragmento de código en el lenguaje C ±:

```
int procedimiento(int &i) {
  j = i * j;
  i = j + 1;
}
i = 2;
j = 3;
procedimiento(i);
printf("%d %d", j, i);
```

Imprime:

A. 6 2
B. 6 7
C. 3 7
D. 3 2

6.- Seleccione la afirmación correcta:

A. Un programa es totalmente correcto si (i) siempre que termina el resultado es correcto, y (ii) termina siempre.
B. Un programa es totalmente correcto si (i) siempre que termina el resultado es correcto, y (ii) para todo dato de entrada válido termina siempre.
C. Un programa es parcialmente correcto si a veces termina produciendo un resultado incorrecto.
D. Un programa es parcialmente correcto si es sintácticamente correcto, pero no semánticamente.

7.- En C+/-, cómo se realizaría la declaración de un tipo para representar los cuatro puntos cardinales:

A. typedef MiTipo { Norte, Sur, Este, Oeste };
B. typedef enum MiTipo { Norte, Sur, Este, Oeste };
C. typedef set MiTipo { Norte, Sur, Este, Oeste };
D. typedef MiTipo = { Norte, Sur, Este, Oeste };

8. Dada la siguiente sentencia en el lenguaje C ±:

DatoUno = DatoDos;

A. Es incorrecta para formaciones
B. Es siempre correcta
C. Es incorrecta para registros
D. Es incorrecta para tipos abstractos de datos (TAD)

9. La sentencia:

```
return resultado;
```

A. Sólo se puede utilizar en el bloque de una función en una única ocasión
B. Se debe utilizar en el bloque de un subprograma en múltiples ocasiones
C. Sólo se puede utilizar en el bloque de un subprograma en una única ocasión
D. Se puede utilizar en el bloque de una función en múltiples ocasiones

10. La ejecución del siguiente fragmento de código en el lenguaje C ±:

```
for (int i=3;i>=0;i--) {
  printf("%d ", i);
  j = i;
  while (j>1) {
    printf("%d ", i);
    j=i-1;
  }
}
```

A. No termina nunca
B. Imprime 3 3 3 2 2 1 0
C. Imprime 3 3 3 3 2 2 2 1 1 0
D. Imprime 3 3 2 2 1 0

# EJERCICIO DE PROGRAMACIÓN

Realizar un tipo abstracto de datos bicicleta para representar las características de una bicicleta. La bicicleta que se quiere representar se describe con el perímetro de las ruedas en metros (PER), un grupo de platos caracterizados cada uno de ellos por el número de dientes (como máximo 5 platos de distinto número de dientes) y otro grupo de piñones traseros caracterizados por el número de dientes (como máximo 10 coronas de piñones de distinto número de dientes). Las operaciones que se deben realizar son: 1) "incluir un plato nuevo" en la bicicleta que es un procedimiento que recibe un plato caracterizado por su número de dientes y lo añade a la bicicleta si hay sitio libre y en la posición de orden que le corresponda (de menor a mayor o de mayor a menor) y 2) "calcular el desarrollo métrico" que es una operación que recibe el plato y la corona que se está usando y devuelve la medida en metros que se desplaza la bicicleta por cada revolución de los pedales (Desarrollo métrico = PER * n / m, n el número de dientes del plato en uso, y m el número de dientes del piñón en uso).