



## Representación de Datos Ingeniería Técnica de Telecomunicación – Telemática

Leganés, 13 de Junio de 2008  
Duración de la prueba: 45 min.

No se permite el uso de libros ni apuntes.  
Puntuación: 4 puntos sobre 10 del examen.

Cada pregunta tiene una única respuesta correcta. En cada pregunta sólo se puede marcar una opción. La contribución de esta parte a la nota del examen estará entre 0 y 4 puntos. Se calcula dividiendo el número de preguntas acertadas entre 6. **Cada pregunta contestada y fallada resta media pregunta acertada.** Las preguntas no contestadas no restan nada.

### Modelo B

1.- Indica qué afirmación acerca de interfaces y clases abstractas es correcta:

- (a) Una clase sólo puede implementar una interfaz, pero puede heredar de varias clases abstractas.
- (b) Una clase sólo puede implementar una interfaz y sólo puede heredar una clase abstracta.
- (c) Una clase sólo puede heredar de una clase abstracta, pero puede implementar varias interfaces.

2.- El siguiente algoritmo de búsqueda:

```
private int search(Comparable[] a, Comparable x, int low, int high) {  
    if (low > high)  
        return -1;  
    int mid = ( low + high ) / 2;  
    if (a[mid].compareTo(x) < 0)  
        return search(a, x, mid + 1, high);  
    else if (a[mid].compareTo(x) > 0)  
        return search(a, x, low, mid - 1);  
    else  
        return mid;  
}
```

- (a) Es una versión lineal de búsqueda binaria.
- (b) Es una versión recursiva de búsqueda binaria.
- (c) Es una versión recursiva de búsqueda lineal.

3.- Si un programa supera con éxito todas las pruebas de una batería de pruebas cuya cobertura es el 100 %, y suponiendo que no hay ningún error en el diseño ni en la programación de las pruebas:

- (a) No se puede afirmar que el programa esté libre de errores.
- (b) Esto no puede ocurrir, dado que es imposible lograr una cobertura del 100 % en una batería de pruebas.
- (c) Se puede afirmar que el programa no tiene errores.

4.- Una técnica para medir la calidad de un plan de pruebas consiste en introducir errores a propósito en el programa para ver si las pruebas lo detectan. Esta técnica se conoce como:

- (a) Pruebas de solidez.
- (b) Pruebas de regresión.
- (c) Pruebas de mutación.

5.- El tipo de recursión del siguiente método:

```
int size (NodoB a) {
    if (a == null)
        return 0;
    else
        return 1 + size(a.izq) + size(a.der);
}
```

- (a) Es *en cascada*.
- (b) Es *no por la cola*.
- (c) Es *anidada*.

6.- Indica cuál de las siguientes afirmaciones es correcta:

- (a) No se puede acceder a un atributo declarado como *static* desde un método de instancia.
- (b) La palabra reservada *static* aplicada a un atributo indica que su valor es constante.
- (c) Un método declarado con la palabra reservada *static* no se ejecuta en el contexto de ninguna instancia.

7.- La oración “en las tres notaciones estudiadas (prefijo, infijo, postfijo) los símbolos de nodos externos siempre están escritos en el mismo orden” (es decir, “si eliminamos los símbolos correspondientes a los nodos internos, resulta la misma secuencia en las tres notaciones”):

- (a) Es a veces correcta y a veces falsa.
- (b) Es siempre correcta.
- (c) Es siempre falsa.

8.- La palabra clave *final* aplicada a un método de una clase significa que:

- (a) El valor que devuelve el método es constante.
- (b) El método no puede ser sobrescrito en ninguna subclase.
- (c) No se puede programar ninguna subclase de dicha clase.

9.- Si un método *a* llama en su cuerpo al propio método *a*, se dice que tenemos un caso de:

- (a) Recursión lineal.
- (b) Recursión directa.
- (c) Recursión inmediata.

10.- El siguiente fragmento de código en una lista encadenada:

```
Nodo n;
for (n = primero; n != null; n = n.getSiguiete()) {...}
```

- (a) Es correcto.
- (b) No es correcto porque es necesario inicializar la variable *nodo* mediante `Nodo n = new Nodo()`.
- (c) No es correcto porque no se comprueba el caso de que la lista esté vacía.

- 11.- Una de las diferencias entre ordenación por mezcla (*merge-sort*) y ordenación rápida (*quick-sort*) es que:
- (a) *Merge-sort* necesita, al contrario que *quick-sort*, un *array* auxiliar.
  - (b) Ninguna de las otras dos respuestas es correcta.
  - (c) *Quick-sort* es idóneo para ordenación externa, mientras que *merge-sort* lo es para ordenación interna.
- 12.- El algoritmo de ordenación *Shell*, ejecutado sobre cantidades grandes de datos:
- (a) Es, en media, más rápido que ordenación por inserción.
  - (b) Es en media, igual de rápido que ordenación por inserción.
  - (c) Es, en media, más lento que ordenación por inserción.
- 13.- Sea una secuencia que contiene únicamente los símbolos “[” y “]”, que pueden estar repetidos y en cualquier orden. Para comprobar que la secuencia está correctamente construida siguiendo la regla de los paréntesis (un paréntesis se cierra con “]” después de abrirlo con “[” y no antes, y hay tantos “]” como “[”)
- (a) No hace falta utilizar ni una pila ni una cola.
  - (b) Hay que utilizar necesariamente una cola.
  - (c) Hay que utilizar necesariamente una pila.
- 14.- Indica cuál de las siguientes afirmaciones es *falsa*:
- (a) En el caso medio de *Quick-sort*, el tiempo necesario para llevar a cabo la ordenación crece linealmente con el número de datos a ordenar.
  - (b) Tanto *quick-sort* como *merge-sort* (ordenación por mezcla) dividen los datos a ordenar en dos bloques y se invocan recursivamente sobre cada bloque.
  - (c) En el caso peor de *quick-sort*, el tiempo de ordenación crece cuadráticamente con el número de datos a ordenar ( $N^2$ ).
- 15.- Si se ejecuta el método *main* siguiente, en la salida estándar se imprimirá:
- ```
public static void duplicar(int n) {
    n = n * 2;
}
public static void main(String[] args) {
    int n = 5;
    duplicar(n);
    System.out.println(n);
}
```
- (a) Salta una excepción, porque un método no puede modificar el valor de sus parámetros.
  - (b) “5”.
  - (c) “10”.
- 16.- Desde un constructor se puede invocar a un constructor de la superclase:
- (a) Con la palabra reservada *new*, que puede aparecer en cualquier lugar del constructor.
  - (b) Con la palabra reservada *super*, que además debe ser la primera sentencia del constructor.
  - (c) Con la palabra reservada *super*, que puede aparecer en cualquier lugar del constructor.

17.- La altura de un árbol binario de búsqueda es siempre:

- (a) Ninguna de las otras dos respuestas es correcta.
- (b) Esencialmente proporcional al tamaño del árbol.
- (c) Esencialmente proporcional al logaritmo del tamaño del árbol.

18.- En una cola con prioridad (*priority queue*) la estrategia de extracción de datos se conoce también como:

- (a) *Last in, first out* (LIFO).
- (b) Ninguna de las otras dos respuestas es correcta.
- (c) *First in, first out* (FIFO).

19.- Un árbol se llama ordenado si:

- (a) Para cada nodo existe un orden lineal para todos sus hijos.
- (b) Los nodos se han introducido en el árbol según el orden de llegada.
- (c) Los hijos de todos los nodos van de menor a mayor.

20.- Dados los siguientes fragmentos de código, indica qué resultado se mostrará por salida estándar:

```
public class A {
    public String getNombre() {return "Hola";}
}
public class B extends A {
    public String getNombre() {return "Adiós";}
}
public class Prueba {
    public static void main(String[] args) {
        A ref = new B();
        System.out.println(ref.getNombre());
    }
}
```

- (a) “Adiós”
- (b) No compila porque se realiza una asignación entre tipos de datos distintos.
- (c) “Hola”

21.- El siguiente método:

```
public boolean isEmpty() {
    return (top < 0);
}
```

se corresponde con una implementación de pilas basada en:

- (a) Ninguna de las otras dos respuestas es correcta.
- (b) Arrays.
- (c) Listas encadenadas.

22.- Dado un método que implementa búsqueda binaria, ¿cuál es la diferencia, en media, entre el número de comparaciones necesarias para buscar entre  $N$  datos y el número de comparaciones necesarias para buscar entre  $\frac{N}{4}$  datos?

- (a) 2 comparaciones.
- (b)  $\frac{N}{4}$  comparaciones.
- (c)  $\frac{3N}{4}$  comparaciones.

23.- La notación polaca inversa se corresponde con:

- (a) La notación infijo.
- (b) La notación prefijo.
- (c) La notación postfijo.

24.- En Java, los parámetros en una llamada a un método se pasan:

- (a) Todos por valor.
- (b) Todos por referencia.
- (c) Algunos por valor y otros por referencia.

**Soluciones:**

- 1.- c
- 2.- b
- 3.- a
- 4.- c
- 5.- a
- 6.- c
- 7.- b
- 8.- b
- 9.- b
- 10.- a
- 11.- a
- 12.- a
- 13.- a
- 14.- a
- 15.- b
- 16.- b
- 17.- a
- 18.- b
- 19.- a
- 20.- a
- 21.- b
- 22.- a
- 23.- c
- 24.- c