Programación de Sistemas
Examen final (convocatoria ordinaria)

Leganés, 31 de mayo de 2018 — Examen convocatoria ordinaria (teoría)
Duración de la prueba: 40 minutos — Puntuación: 3 puntos sobre 10 del examen

Sólo una opción es correcta en cada pregunta. Cada respuesta correcta suma 0,15 *puntos. Cada respuesta incorrecta resta* 0,05 *puntos. Las preguntas no contestadas no suman ni restan.*

- Marca la respuesta a cada pregunta con una equis ("X") en la tabla de abajo.
- Si marcas más de una opción o ninguna opción, la pregunta se considera no contestada.
- Rellena **tus datos personales** antes de comenzar a realizar el examen.

Nombre:	Grupo:

Firma:

NIA:

	A	B	C	D		A	B	C	D
1					11				
2					12				
3					13				
4					14				
5					15				
6					16				
7					17				
8					18				
9					19				
10					20				

1.- La palabra reservada `this` se utiliza en Java para referirse a:

(a) *** Un objeto.

(b) Un método.

(c) Un atributo.

(d) Una clase.

2.- Dado el siguiente código...

```
public class C{
    private static int a = 0;
    public static void main(String[] args) {
        System.out.println(a);
    }
}
```

(a) *** El código es correcto.

(b) El código es incorrecto porque no se puede llamar a un atributo estático desde un método estático.

(c) El código es incorrecto porque hay que crear un objeto de la clase C antes de poder utilizar el atributo a.

(d) El código es incorrecto porque no puede accederse al atributo a desde el *main()*, al ser a un atributo privado.

3.- El modificador `final` aplicado sobre el parámetro `a` en el siguiente método...

```
public int m(final int a){...}
```

(a) *** Indica que el valor que toma el parámetro a al invocar al método m no puede ser modificado dentro de dicho método.

(b) Indica que el método m no puede devolver a, por lo que la sentencia *return a;* daría un error de compilación.

(c) Indica que el método m no puede ser sobreescrito en las subclases de la clase donde se haya implementado.

(d) Indica que el espacio en memoria ocupado por la variable a no se liberará al finalizar la ejecución del método m.

4.- Dada una clase `A` que implementa la interfaz `I`, y siendo `B` y `C` dos clases derivadas de `A`, y sabiendo que todas las clases tienen constructores que no reciben parámetros, indica cuál de las siguientes afirmaciones es correcta.

(a) *** `I[] array = {new A(), new B(), new C()};`

(b) `B[] array = {new A(), new B(), new C()};`

(c) `C[] array = {new A(), new C(), new I()};`

(d) `A[] array = {new B(), new C(), new I()};`

5.- La estructura de datos `LinkedStack<Person>`, es decir, una pila implementada con listas enlazadas que guarda información sobre personas puede contener:

(a) *** Objetos de la clase `Person` y de cualquier clase no abstracta derivada de ella.

(b) Exclusivamente objetos de la clase `Person`.

(c) Objetos de la clase `Person` y de cualquier interfaz implementada por la clase `Person`.

(d) Objetos de las clases `Person` y `Object`.

6.- ¿Cuál de las siguientes afirmaciones sobre constructores es correcta?

(a) *** Todas las clases cuentan con un constructor por defecto que no recibe parámetros, excepto en el caso de que se hayan programado explícitamente otros constructores.

(b) Los constructores no pueden sobrecargarse.

(c) Los constructores tienen que definir un tipo de retorno, aunque sea *void*.

(d) Un constructor puede invocar a otro constructor de la misma clase utilizando `this` y pasándole el nombre de la clase entre paréntesis.

7.- En la sobrecarga de métodos hay:

(a) *** métodos con mismo nombre, pero distinto número y/o tipo de parámetros

(b) métodos con mismo nombre, número y tipo de parámetros

(c) métodos con distinto nombre, pero mismo número y tipo de parámetros

(d) métodos con distinto nombre, número y tipo de parámetros

8.- Dado el siguiente código y su clase de test, ¿qué cobertura de líneas se alcanza?

```
public class C {
  public int m(int a, int b){
    if(a<0)
      return b;
    if(b<0)
      return a;
    return a+b;
  }
}
public class CTest {
  C c = new C();
  @Test
  public void test() {
    assertEquals(c.m(1, -1), 1);
    assertEquals(c.m(-1, 1), 1);
    assertEquals(c.m(1,1),2);
  }
}
```

(a) *** 100 %.

(b) 25 %.

(c) 50 %.

(d) 75 %.

9.- Indica el resultado de invocar al siguiente método recursivo con valores `n = 10, m = 2`.

```
public int m(int n, int m) {
  if(n<m)
    return n;
  else
    return 2*m(n-m, n+m);
}
```

(a) *** 16.

(b) 4.

(c) 6.

(d) 10.

10.- Indica el tipo de recursión del siguiente método de un árbol binario definido de forma recursiva, según la cual cada nodo raíz tiene referencias a subárboles izquierdo (`getLeft()`) y derecho (`getRight()`), los cuales pueden estar vacíos (`isEmpty()`) o no.

```
public int m() {
  if(isEmpty())
    return 0;
  else
    return root.getLeft().m() + root.getRight().m() + 1;
}
```

(a) *** Recursión no lineal en cascada.

(b) Recursión no lineal anidada.

(c) Recursión lineal por la cola.

(d) Recursión lineal no por la cola.

11.- Dado el siguiente código indique qué devuelve el método *m()*:

```
public class MyBasicLinkedList<E> {
  protected Node<E> first;
  private int mR(Node<E> list) {
    if (list == null) return 0;
    else return 1 + mR(list.getNext());
  }
  public int m() {
    int a = mR(first);
    return a;
  }
}
```

(a) *** El número de nodos de la lista enlazada.

(b) 0

(c) 1

(d) La suma de los valores de todos los elementos de la lista.

12.- Después de ejecutar el siguiente método pasando una pila (*stack*) como parámetro:

```
void process(Stack stack) {
  Stack aux = new Stack();
  while (!stack.isEmpty()) {
    aux.push(stack.pop());
  }

  while (!aux.isEmpty()) {
    stack.push(aux.pop());
  }
}
```

(a) *** La pila (*stack*) contendrá los mismos elementos que antes, en el mismo orden.

(b) La pila (*stack*) contendrá los mismos elementos que antes, en orden inverso.

(c) Los cambios realizados sobre de la pila (*stack*) dentro del método no serán visibles porque en Java los parámetros de los métodos se pasan por valor.

(d) La pila (*stack*) estará vacía.

13.- Para programar una cola de prioridad en Java es necesario que la clase de los objetos que se utilicen como clave en la cola implementen:

(a) *** El interface `Comparable`

(b) El interface `Queue<E>`

(c) El interface `Deque<E>`

(d) El interface `Equals`

14.- ¿Qué métodos de una cola doble (*deque*) podrían usarse para implementar los métodos *push* y *pop* de una pila (stack)?

(a) *** *insertFirst* para *push* y *removeFirst* para *pop*.

(b) *insertLast* para *push* y *removeFirst* para *pop*.

(c) *insertLast* para *pop* y *removeLast* para *push*.

(d) *insertFirst* para *push* y *removeLast* para *pop*.

15.- La interfaz `BTree<E>` contiene un método `toString()` que devuelve el contenido del árbol en inorden ¿Qué hace el exáctamente entonces el método `toString()`?:

(a) *** Devuelve un String con el contenido del hijo izquierdo recursivamente, después el contenido del nodo raíz, y después el contenido del hijo derecho recursivamente

(b) Devuelve un String con el contenido del hijo derecho recursivamente, después el contenido del nodo raíz, y después el contenido del hijo izquierdo recursivamente

(c) Devuelve un String con el contenido de los hijos recursivamente y después el contenido del nodo raíz

(d) Devuelve un String con el contenido del nodo raíz y después el contenido de los hijos recursivamente

16.- Dado el árbol binario de búsqueda representado por el siguiente array: [50,40,60,30,45,55,70] si hacemos un recorrido inorden obtendremos la secuencia:

(a) *** 30, 40, 45, 50, 55, 60, 70

(b) 50, 40, 30, 45, 60, 55, 70

(c) 30, 45, 40, 55, 70, 60, 50

(d) 30, 45, 55, 70, 40, 60, 50

17.- Dado el árbol binario representado por el siguiente array: [1, 2, 4, 3, 5, 6, 9, 8, 7], ¿cúal de las siguientes afirmaciones es correcta?

(a) *** Es un montículo con condición min-heap.

(b) No es montículo, a pesar de que cumple la condición de min-heap.

(c) Es un montículo con condición max-heap.

(d) No un es montículo, a pesar de que cumple la condición de max-heap.

18.- Dado el siguiente array [1, 2, 3, 4, 5, 6, 7, 8, 9]. ¿Cuántas iteraciones da el algoritmo de BinarySearch para encontrar al número 5?

(a) *** Lo encuentra en la primera iteración.

(b) Lo encuentra en la segunda iteración.

(c) Lo encuentra en la tercera iteración.

(d) Lo encuentra en la cuarta iteración.

19.- Dado el algoritmo de ordenación SelectionSort que busca el mínimo de la parte desordenada y lo intercambia por el primer elemento no ordenado, y este array que está siendo ordenado de menor a mayor por el algoritmo, el cual actualmente lleva tres iteraciones [3, 8, 15, 25, 20, 45, 35] ¿Cómo quedará el array en la siguiente iteración?:

(a) *** 3, 8, 15, 20, 25, 45, 35

(b) 3, 8, 15, 25, 20, 45, 35

(c) 3, 8, 15, 20, 25, 35, 45

(d) 3, 8, 20, 15, 25, 45, 35

20.- Dado un array desordenado de mil elementos, ¿cuál de los siguientes algoritmos es, de media, el más eficiente?

(a) *** Merge Sort

(b) Bubble Sort

(c) Selection Sort

(d) Insertion Sort