SISTEMAS INFORMÁTICOS I **Septiembre-Reserva 2005**
INGENIERÍA INFORMÁTICA **Código Carrera: 55 - Código Asignatura: 077** **Tipo de Examen: F**
NO SE PERMITE EL USO DE NINGÚN MATERIAL **Tiempo: 2 horas**

Apellidos: _____ Nombre: _____ DNI: _____

Centro Asociado en el que está MATRICULADO: _____

INSTRUCCIONES: Complete sus datos personales en la cabecera de esta hoja, y **ENTRÉGUELA OBLIGATORIAMENTE** con el resto de hojas de su examen. **Cualquier examen que no venga acompañado de esta hoja de enunciados no será corregido**. Complete **TODOS** los datos que se piden en la hoja de lectura óptica o **en caso contrario su examen no será corregido.** El examen consta de 20 preguntas. Cada respuesta acertada tiene un valor de 0.5 puntos y cada respuesta incorrecta de -0.25 puntos. Para superar el examen es necesario obtener una puntuación de al menos 5 puntos.

Test : Conteste exclusivamente en la HOJA DE LECTURA ÓPTICA, no olvidando marcar que su tipo de examen es F.

1. En el siguiente fragmento de programa, ¿cuántas veces se ejecuta la sentencia `System.out.println` ?

```
a = 9;
for (int i = 0; i < 100; i++)
      if ((a % 4 == 0) || (i % 2) == 0)
            System.out.println(a + " " + i);
```

A) 50
B) 0
C) 25
D) 100

2. ¿Cuál es el resultado de la ejecución del siguiente código?

```
public class matriz1 {
  public matriz1() {
  }
  public static void main(String[] args) {
    Integer[] arrayInteger = new Integer[5];
    arrayInteger[1]= new Integer(3);
    arrayInteger[2]= new Integer(4);
    arrayInteger[3]= new Integer(5);
    arrayInteger[4]= new Integer(6);

    for ( int i = 0; i < arrayInteger.length; i++)
    { System.out.print(arrayInteger[i]);
    }
  }
}
```

A) `01234`
B) `null34567`
C) `3456null`
D) `null3456`

3. ¿Cuál es el resultado de la ejecución del siguiente código?

```
import java.util.*;
public class Iterador {

  public static void main(String[] args) {
    HashSet conjunto = new HashSet();
    conjunto.add("Esto ");
    conjunto.add("es ");
    conjunto.add("una ");
    conjunto.add("prueba ");
    conjunto.add("prueba ");

    Iterator itr = conjunto.iterator();

    while(itr.hasNext()){
      System.out.print(itr.next());
    }
  }
}
```

A) `Esto es una prueba prueba`
B) `Esto es una prueba`
C) En principio no se sabe. Depende de la función de conversión de claves.
D) Al compilar da un error. No se pueden añadir dos elementos iguales a un objeto `HashSet`.

4. ¿Cuál es el resultado de la ejecución del siguiente código?

```
import java.io.*;

public class Valores {

  public Valores() {
  }

  public static void main(String[] args) {
      byte[] byteArray = {0,1,2,3,4,5,6,7,8,9};
      ByteArrayInputStream flujoArrayByte =
              new ByteArrayInputStream(byteArray);
      while(flujoArrayByte.available() != 0){
          byte leido = (byte)flujoArrayByte.read();
          System.out.print(leido);
          flujoArrayByte.skip(1);
      }
      try {
            flujoArrayByte.close();
      } catch (IOException ex) {
      }
  }
}
```

A) `2468`
B) `0123456789`
C) `9876543210`
D) `02468`

5. ¿Cuál de las siguientes afirmaciones es cierta?

   A) `StringTokenizer` devuelve todos los símbolos contenidos en una cadena de caracteres de uno en uno.
   B) `StringTokenizer` devuelve todos los símbolos contenidos en una cadena de caracteres ordenados según su código ASCII.
   C) `StringTokenizer` devuelve sólo los símbolos contenidos en una cadena de caracteres cuyas claves aparecen en una función de conversión de claves.
   D) `StringTokenizer` devuelve todos los símbolos contenidos en una cadena de caracteres ordenados según una función de conversión de claves.

6. Señale cuál es el resultado de la ejecución del código siguiente. Las líneas se han numerado con el propósito de facilitar la referencia a las mismas.

```
1     public class Conversion {
2           public static void main(String args[]) {
3                 int num = 300;
4                 String num_s1 = (String)num;
5                 String num_s2 = String.valueOf(num);
6                 String num_s3 = "" + num;
7                 System.out.println("Resultado: "+num_s1+num_s2+num_s3);
8           }
9     }
```

   A) Se obtiene error en la línea 4.
   B) `Resultado: 900`
   C) `Resultado: 300300300`
   D) Ninguno de los anteriores.

7. Señale cuál es el resultado de la ejecución del método `main` de la clase Ejemplo1. Las líneas se han numerado con el propósito de facilitar la referencia a las mismas.

```
1     class Ejemplo1_A {
2           String mensaje = "A";
3           void mensajeA() { System.out.print(mensaje); }
4     }
5
6     class Ejemplo1_B extends Ejemplo1_A {
7           String mensaje = "B";
8           void mensajeA() {System.out.print(mensaje); }
9           void mensajeB() {System.out.print(mensaje); }
10    }
11
12    public class Ejemplo1 {
13          public static void main(String[] args) {
14                Ejemplo1_B e1 = new Ejemplo1_A();
15                e1.mensajeA();
16                e1.mensajeB();
17          }
18    }
```

   A) `AB`
   B) Se obtiene error en la línea número 14.
   C) Se obtiene error en la línea número 16.
   D) Ninguno de las anteriores.

8. Señale cuál es el resultado de la ejecución del código siguiente. Las líneas se han numerado con el propósito de facilitar la referencia a las mismas.

```
1     public class Ejemplo2 {
2
3           static int a = 1;
4           static int b = 3;
5
6           static int producto(int p1, int p2) {
7                 return p1*p2;
8           }
9
10          int suma(int p1, int p2) {
11                return p1 + p2;
12          }
13
14          public static void main(String[] args) {
15                int a = 2;
16                System.out.print(producto(a,b));
17                System.out.print(suma(a,b));
18          }
19    }
```

A) Se produce error en la línea número 16.
B) Se produce error en la línea número 17.
C) `65`
D) Ninguno de las anteriores.

9. Indique cuál es el resultado de la ejecución del código siguiente.

```
public class Ejemplo3 {
      public static void main(String[] args) {
          int x[] = { 0, 1, 2, 3};
          try {
            double z = x[x.length-1] / x[0];
            System.out.print(z);
            return;
          } catch (ArrayIndexOutOfBoundsException e1) {
            System.out.print("P1");
            x[0] = 1;
          } catch (ArithmeticException e2) {
            System.out.print("P2");
          } catch (Exception e3) {
            System.out.print("P3");
          } finally {
            System.out.print("F");
          }
          System.out.print("OK");
      }
}
```

A) `P2P3FOK`
B) `P2FOK`
C) `3.0F`
D) Ninguno de las anteriores.

10. A continuación se muestra el código de la clase Ejemplo4. Se han numerado las líneas para facilitar la referencia a las mismas.

```
1     import java.util.*;
2     public class Ejemplo4 {
3           public static void main(String[] args) {
4                 ArrayList sumandos   = new ArrayList();
5                 int  suma = 0;
6                 for (int i=0; i<3; i++)
7                       sumandos.add(new Integer(i));
8                 for (int i=0; i<sumandos.size(); i++)
9                       suma += ((Integer)sumandos.get(i)).intValue();
10                System.out.print("Suma: " + suma);
11          }
12    }
```

Señale cuál de las siguientes afirmaciones es correcta:

A) Hay un error en la línea 7.
B) Hay un error en la línea 9.
C) El código de la clase no contiene errores. El resultado de la ejecución de su método `main` es la impresión en el flujo de salida estándar de:
```
Suma: 3
```
D) Ninguna de las afirmaciones anteriores es correcta.

11. ¿Cuál es la diferencia entre bloquear un thread mediante el empleo de `suspend()` y mediante el uso de `wait()`?

A) La diferencia radica en que `suspend()` detiene la ejecución del thread e impide la ejecución de cualquier otro método que forme parte del mismo al no liberar el bloqueo sobre todo el objeto. Por el contrario, `wait()` libera el bloqueo sobre el objeto, lo que significa que otros métodos `synchronized` del objeto thread podrían ser invocados durante `wait()`.
B) La diferencia estriba en la forma en que hay que despertar el thread. Si se utiliza `suspend()` hay que invocar al método `notify()`, mientras que el uso de `wait()` implica la utilización de `resume()` para continuar el procesamiento del thread en el punto en que se detuvo la ejecución. En ambos casos no se produce la liberación del bloqueo sobre el objeto, con lo que no se pueden invocar otros métodos del objeto thread.
C) El método de bloqueo mediante `wait()` se emplea, exclusivamente, para producir la detención del thread durante los milisegundos especificados, mientras que `suspend()` provoca la detención hasta que se produzca la invocación del método `resume()`.
D) Ninguna de las anteriores.

12. El método `wait()` se caracteriza porque:

A) Se puede utilizar dentro de cualquier método de un objeto thread con independencia de los atributos del método al tratarse de un método propio de la clase base `Object` y no de la clase base `Thread`.
B) Se tiene que utilizar siempre dentro de un bloque o método `synchronized` debido a que es un método propio de la clase base `Object` y no de la clase base `Thread`.
C) Provoca la detención del thread e impide la llamada a cualquier método que forme parte del objeto ya que no libera el bloqueo del objeto durante la detención por tratarse de un método propio de la clase base `Thread`.
D) Ninguna de las anteriores.

13. La especificación de la URL que hay que elaborar para acceder a una base de datos mediante JDBC debe constar de los siguientes campos:

A) `jdbc:<nombre_base_de_datos>:<login>:<password>`
B) `jdbc:<subprotocolo>:<nombre_base_de_datos>`
C) `jdbc-odbc:<subprotocolo>:<nombre_base_de_datos>`
D) Ninguna de las anteriores.

14. ¿Cuál de las siguientes afirmaciones es correcta?

A) El resultado de una consulta SQL a la base de datos mediante el método `executeQuery` se almacena en un objeto `ResultSet`.
B) Los resultados de una consulta SQL mediante el método `getConnection` se almacenan en un objeto `ResultSet`.
C) El resultado de una consulta SQL con el método `executeQuery` se almacena en un objeto `String[]`.
D) Ninguna de las anteriores.

15. Tras realizar una consulta SQL a una base de datos, el resultado:

A) Se almacena en un objeto `StringTokenizer` y se recorre el contenido del objeto utilizando, por ejemplo, el método `hasMoreTokens()`. Los registros recuperados se extraen utilizando el método `nextToken()`.
B) Se almacena en un objeto `Vector`. Los registros recuperados de la base de datos se extraen de ese objeto `Vector` mediante el método `elementAt()` que devuelve un objeto genérico `Object` por cada registro recuperado de la base de datos.
C) Se almacena en un objeto `ResultSet`, recorriéndose este objeto con el método `next()` y utilizando métodos `get***()` para recuperar la información de los campos que componen los registros recuperados.
D) Ninguna de las anteriores.

16. Dado el siguiente código

```
public class ejer1 {
      public static void main(String args[]) {
            for (int i=0;i<5; i++){
                  System.out.print(" Numero i= " + i + ":");
                  for(int j=0;j<12;j++){ if (j>15) break;
                        System.out.print("Numero j="+j+":");
                  }
                  System.out.println();
            }
      }
}
```

¿Cuál de las afirmaciones siguientes es correcta?

A) Cuando se ejecuta **break** por primera vez se sale del segundo bucle y finaliza el programa independientemente del valor de `j`.
B) Cuando se ejecuta **break**, si la variable `i` es igual a 10 finaliza el programa.
C) Cada vez que se ejecuta **break** se sale del bucle interior y continúa la ejecución del bucle exterior.
D) Ninguna de las anteriores.

17. ¿Cuál es el resultado de ejecutar el código siguiente?

```
public class Test {
      static boolean b;
      public static void main(String [] args) {
            int x = 0;
            if (b) {
                  x = 1;
            } else if (b) {
                  x = 2;
            } else if (b) {
                  x = 4;
            } else {
                  x = 3;
            }
            System.out.println("x = " + x);
      }
}
```

A) x = 4
B) x = 3
C) x = 2
D) Es un programa erróneo y no se ejecuta.

18. ¿Cuál sería la salida del siguiente fragmento de código?

```
class prueba {
  int i;
  public prueba() {
    i=0; }
}

public class ejer1 {
  public static void main(String args[]) {
    prueba uno = new prueba();
    prueba dos = new prueba();
    uno.i= 20;
    dos.i= 30;
    System.out.println(uno.i);
  }
}
```

A) 50
B) 20
C) 30
D) Ninguna de ellas.

19. ¿Cuál es el resultado de la ejecución del código siguiente?

```
public class ejer1 {
     public static void main(String [] args) {
           int i=3, j=0, result=1;
           result += i-- * --j ;
           System.out.println( result );
     }
}
```

A) 1
B) -2
C) 2
D) Ninguna de las anteriores.

20. Dada la siguiente orden de ejecución en línea de comandos

```
Java Clases para1 para2 para3
```

¿Dónde se almacenan inicialmente los valores de los parámetros?

A) Todos los parámetros se almacenan en una única cadena denominada `arg`.
B) Se almacena en un array indexado entre 0 y 2.
C) Se almacena en un array indexado entre 1 y 3.
D) Ninguna de las anteriores.