dit UPM	ETSIT-UPM. PROGRAMACIÓN. CURSO 2012-2013. EXAMEN 2. APELLIDOS: NOMBRE: EMAIL:	

1. (2 puntos) Escriba un método *imprimeHistograma* para imprimir por pantalla los valores de un array de enteros que se pasa como parámetro. Lanza una excepción si el array es null. Por cada valor, debe haber una línea de texto con tantos caracteres '*' como indique el valor. Se supone que el array contiene valores positivos pequeños. Se muestra un ejemplo a la derecha si se pasa el parámetro {2, 4, 6, 6, 9, 12, 3, 0, 1}

```
**
****
******
******
*********
***********
***

*
```

```
public void imprimeHistograma (int [] valores) throws Exception {
      if (valores == null) {
            throw new Exception("Entero no positivo");
      }
      for (int i = 0; i < valores.length; i++) {
        for (int j = 1; j <= valores[i]; j++) {
          System.out.print ("*");
        }
        System.out.println  ();
      }
}
```

2. (3 puntos). Escriba el cuerpo del método *int[][] mezcla (int[][] m1, int[][] m2)* que combina dos matrices bidimensionales de números enteros y produce una nueva. Las matrices m1 y m2 pueden tener dimensiones diferentes, por ejemplo m1 puede ser 2x3 y m2 puede ser 4x2, pero siempre van a ser rectangulares y nunca serán null. La matriz resultante debe tener tamaño suficiente para contener a las otras dos matrices (m1 y m2). Con m1 de dimensiones 2x3 y m2 de dimensiones 4x2 la matriz resultante r tendrá dimensiones 4x3 y como valores:

- r[i][j]= 0, si la posición (i, j) no existe en m1 ni en m2
- r[i][j]= m1[i][j], si la posición (i, j) no existe en m2 pero sí en m1
- r[i][j]= m2[i][j], si la posición (i, j) no existe en m1 pero sí en m2
- r[i][j]= (m1[i][j] + m2[i][j]) / 2, si la posición (i, j) existe en m1 y en m2

```
int[][] mezcla(int[][] m1, int[][] m2) {
  int filasM1 = m1.length;
  int columnasM1 = m1[0].length;
  int filasM2 = m2.length;
  int columnasM2 = m2[0].length;
  int[][] resultado = new int[Math.max(filasM1, filasM2)][Math.max(columnasM1,
columnasM2)];
  for (int i = 0; i < resultado.length; i++) {
    for (int j = 0; j < resultado[i].length; j++) {
      resultado[i][j] = 0;
      boolean estaEnM1 = (i < filasM1) && (j < columnasM1);
      boolean estaEnM2 = (i < filasM2) && (j < columnasM2);
      if (estaEnM1)
        resultado[i][j] = m1[i][j];
      if (estaEnM2)
        resultado[i][j] += m2[i][j];
      if (estaEnM1 && estaEnM2)
        resultado[i][j] /= 2;
    }
  }
  return resultado;
}
```

3. (3 puntos). Añada un constructor sin parámetros a la clase Coordenada que cree un objeto en el que las coordenadas son 0.0 y 0.0, y el nombre es una *String* compuesta por "Coordenada" y el número de objetos de clase Coordenada que se hayan creado hasta el momento (incluyendo esta). Añada o cambie lo que sea necesario para lograrlo.
Escriba también el método *equals*. Suponga que los métodos accesores y modificadores ya están hechos.

```
class Coordenada {
  private String nombre;
  private double x;
  private double y;
  public Coordenada (String nombre, double x, double y) {
    this.nombre = nombre;
    this.x = x;
    this.y = y;
  }
  // métodos accesores
  // métodos modificadores
}
```

```
class Coordenada {
  private String nombre;
  private double x;
  private double y;
  private static int nCoordenadas = 0;
  public Coordenada () {
    nCoordenadas ++;
    this.nombre = "Coordenada" + nCoordenadas;
    this.x = 0.0;
    this.y = 0.0;
  }
  public Coordenada (String nombre, double x, double y) {
    nCoordenadas ++;
    this.nombre = nombre;
    this.x = x;
    this.y = y;
  }
  public boolean equals (Coordenada otra) {
    return this.nombre.equals (otra.nombre) && (this.x == otra.x) && (this.y == otra.y);
  }
}
```

4. (2 puntos) Escriba un método que tome como parámetro un número entero entre 0 y 15 (ambos válidos), y devuelve un carácter correspondiente a su valor en hexadecimal ('0','1', …,'9','A','B',…'F'). Por ejemplo, si el parámetro es 10, devuelve el carácter 'A'; si el parámetro es 8, devuelve el carácter '8'. Lanza excepción si el valor del parámetro no es válido. En el cuerpo no puede usar más de 3 sentencias *if*.

```
Hay varias soluciones posibles:

public char enLetras (int n) throws Exception {
   if (n<0 || n>15) throw new Exception("fuera de rango");
  char[] numEnLetras={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};
  return numEnLetras[n];
}

public char enLetras (int n) throws Exception {
  switch (n) {
    case 0: return '0'; case 1: return '1'; case 2: return '2'; case 3: return '3'; case 4: return '4';
    case 5: return '5'; case 5: return '5'; case 6: return '6'; case 7: return '7' case 8: return '8';
    case 9: return '9'; case 10: return 'A'; case 11: return 'B'; case 12: return 'C'; case 13: return 'D';
    case 14: return 'E'; case 15: return 'F';
   default: throw new Exception("fuera de rango");
  }
}

public char enLetras (int n) throws Exception {
   if (n>=0 && n<10)
    return (char)((int)'0'+n);
  else if (n>= 10 && n<16)
    return (char)((int)'A'+n-10);
  else throw new Exception("fuera de rango");
}
```