Afianza tus conocimientos de programación mediante la resolución de ejercicios

Ejercicios de Programación en Java

Condicionales, Bucles, Tablas y Funciones

F.M. Pérez Montes

Edita Asoc. Por la Innovación Educativa Eduinnova.
Esta obra se encuentra disponile en: http://www.eduinnova.es/monografias2011/ene2011/java.pdf

Depósito legal: SE 1211-2011.

ISBN: 978-84-614-7595-7.

A mi hija Claudia,
la solución a todos los problemas.

AGRADECIMIENTOS

A todos los que han hecho posible este libro.
En especial a mi amigo y compañero: Alfonso Jiménez,
por sus innumerables correcciones y por la clase Entrada.

ÍNDICE

PRÓLOGO

El libro *Ejercicios de Programación en Java: Condicionales, Bucles, Tablas y Funciones* nace como fruto de años de docencia en materias relacionadas: Algorítmica, Programación, Lenguaje C y Java, entre otros.

Con el paso del tiempo he constatado que aprender un lenguaje de programación es relativamente sencillo y sobre este tema existen muchos y muy buenos textos. Pero aprender a programar es algo totalmente distinto, que necesita de un mayor esfuerzo y abordar el problema desde otra perspectiva. Siempre utilizo la metáfora del pianista para explicar el tándem programar/lenguaje de programación: saber tocar las notas de un piano es relativamente fácil, tan solo debemos anotar en cada una de las teclas a qué nota musical corresponde. Esto sería similar a conocer un lenguaje de programación. Es muy sencillo utilizar un ***if*** o entender la mecánica de un ***while***.

Volviendo al piano: una vez que dominamos la relación tecla/nota, un pianista debe aprender muchas otras cosas para que aquello que está tocando *suene bien*; esto sería saber tocar el piano. Para saber programar, no basta saber cómo funciona una instrucción sino saber

utilizarla conjuntamente con otras, en el orden y la forma adecuadas para que la aplicación que estamos creando *suene bien*.

Esta obra no es un libro para aprender java ni sus numerosas bibliotecas, es un libro que por medio de ejercicios resueltos, desde cero, y con la práctica facilita la asimilación de las técnicas de programación. Para aprender a programar la mejor forma es desvincular la lógica de la aplicación (cómo hacerlo) del lenguaje utilizado para implementarlo. Dicho en otras palabras: lo mejor es utilizar seudocódigo (un lenguaje teórico de alto nivel) donde no tengamos que preocuparnos por las particularidades del lenguaje de programación, ni por la rigidez de su sintaxis. El inconveniente de utilizar seudocódigo es que el lector no tiene nada tangible, nada con lo que se pueda comprobar el buen funcionamiento de la aplicación; por este motivo se ha decidido utilizar Java. Esta elección se justifica frente a otras alternativas, como el lenguaje C, que también es muy didáctico, simplemente por el hecho de que con Java podemos abstraernos un poco más, al ser un lenguaje de más alto nivel. También hay que decir que en la medida de lo posible no profundizaremos en las bibliotecas del lenguaje; en otras ocasiones esto será totalmente imposible de llevar a la práctica y hemos de trabajar con los detalles.

Para finalizar, desearía comentar que el libro se estructura como un conjunto de boletines de ejercicios que se resuelven de la forma más didáctica posible. Un programador

experto seguramente encontrará soluciones mucho más elegantes y eficientes. Aquí nuestro principal objetivo es que el lector entienda qué está haciendo y por qué lo hace.

La dificultad de los boletines crece gradualmente y en cada boletín se trata un tema distinto. Una vez resueltos los ejercicios de un boletín podremos disponer de ellos para utilizarlos en posteriores boletines.

La resolución de los ejercicios no es única, y en ocasiones merece la pena ver otro enfoque distinto. Es por esto por lo que en algunos casos se han incluido varias soluciones.

Si el lector se enfrenta a la tarea de aprender a programar, este libro, junto con las clases que pueda recibir en una facultad, escuela técnica o ciclo formativo de grado superior, serán una ayuda eficaz para llevar a cabo su objetivo. Esta tarea debe tomarse sin prisas, entendiendo los detalles sutiles y dedicando mucho tiempo a la práctica.

Sevilla, octubre de 2010
Francisco M. Pérez Montes

INTRODUCCIÓN

Este libro está compuesto como una colección de boletines de ejercicios (se encuentran disponibles en el Apéndice **I**). En cada boletín se resuelven ejercicios con una temática común, de la siguiente forma:

Boletín 1..... Variables y condicionales
Boletín 2..... Condicionales y bucles
Boletín 3..... Bucles anidados
Boletín 4..... Tablas
Boletín 5..... Tablas n-dimensionales
Boletín 6..... Funciones

Los ejercicios no tienen solución única, aquí se plantea la más didáctica y fácil de entender, dejando de lado la eficiencia. Cuando existen distintas soluciones, utilizando distintos enfoques, se incluye más de una solución por ejercicio.

La resolución de los ejercicios de programación, son el complemento ideal para las clases de programación impartidas en una facultas, escuela técnica o ciclo formativo de grado superior.

Otro aspecto importante es la entrada por teclado, algo primordial para poder introducir datos y probar nuestros programas. En un principio el alumno no debe tener los conocimientos necesarios para escribir el código que le proporcione dicha entrada. Algo similar ocurre en las asignaturas de programación, lo que plantea el problema de empezar a explicar código y funciones que se escapan al programador novel.

Por todo esto, se ha diseñado la clase Entrada, que permite realizar de forma transparente la entrada por teclado. Aprender a utilizarla es sencillo y proporciona una herramienta cómoda y fiable para dejar de preocuparnos por la entrada de datos. La clase Entrada se encuentra en el Apéndice **II**. Las funciones que proporciona la clase Entrada son:

Entrada.entero()	Lee un número entero por teclado y lo devuelve
Entrada.real()	Lee un número real por teclado y lo devuelve
Entrada.cadena()	Lee una cadena de caracteres y la devuelve
Entrada.caracter()	Lee un solo carácter por teclado y lo devuelve

Boletín 1
Variables y condicionales

1. Pedir los coeficientes de una ecuación se 2° grado, y muestre sus soluciones reales. Si no existen, debe indicarlo.

```
package bol01ej01;

public class Main {

    public static void main(String[] args) {
        double a,b,c; // coeficientes ax^2+bx+c=0
        double x1,x2,d; // soluciones y determinante

        System.out.println("Introduzca primer coeficiente (a):");
        a=Entrada.entero();
        System.out.println("Introduzca segundo coeficiente: (b):");
        b=Entrada.entero();
        System.out.println("Introduzca tercer coeficiente: (c):");
        c=Entrada.entero();

        // calculamos el determinante
        d=((b*b)-4*a*c);
        if(d<0)
            System.out.println("No existen soluciones reales");
        else{
            // queda confirmar que a sea distinto de 0.
```

```
            // si a=0 nos encontramos una división por cero.

            x1=(-b+Math.sqrt(d))/(2*a);
            x2=(-b-Math.sqrt(d))/(2*a);
            System.out.println("Solución: " + x1);
            System.out.println("Solución: " + x2);

        }
    }
}
```

2. Pedir el radio de un círculo y calcular su área. A=PI*r^2.

```
package bol01ej02;

public class Main {

    public static void main(String[] args) {
        double a,r; // área y radio

        System.out.print("Introduce el radio de un circulo: ");
        r=Entrada.real();

        a=Math.PI*(r*r); // para elevar al cuadrado otra opción es: Math.pow (r, 2)

        System.out.println("El área de una circunferencia de radio " + r+ " es: " + a);

    }
}
```

3. Pedir el radio de una circunferencia y calcular su longitud.

```
package bol01ej03;

public class Main {
```

```
    public static void main(String[] args) {
        double l,r; // longitud y radio

        System.out.print("Introduce el radio de una circunferencia: ");
        r=Entrada.real();

        l=2*Math.PI*r;

        System.out.println("La longitud de una circunferencia de radio " + r+ " es: " + l);
    }
}
```

4. Pedir dos números y decir si son iguales o no.

```
package bol01ej04;

public class Main {

    public static void main(String[] args) {
        int n1,n2;

        System.out.print("Introduce un número: ");
        n1=Entrada.entero();
        System.out.print("Introduce otro número: ");
        n2=Entrada.entero();

        if(n1==n2)
            System.out.println("Son iguales");
        else
            System.out.println("No son iguales");
     }
}
```

5. Pedir un número e indicar si es positivo o negativo.

```
package bol01ej05;

public class Main {

    public static void main(String[] args) {
        int num;

        System.out.print("Introduce un número: ");
        num=Entrada.entero();

        if( num < 0)
            System.out.println("Negativo");
        else
            // suponemos que el 0 es positivo.
            System.out.println("Positivo");
    }
}
```

6. Pedir dos números y decir si uno es múltiplo del otro.

```
package bol01ej06;

public class Main {

    public static void main(String[] args) {
        int n1,n2;

        System.out.print("Introduce un número: ");
        n1=Entrada.entero();

        System.out.print("Introduce otro número: ");
        n2=Entrada.entero();
```

```
        if(n1%n2==0)
            System.out.println("Son múltiplos");
        else
            System.out.println("No son múltiplos");
    }
}
```

7. Pedir dos números y decir cual es el mayor.

```
package bol01ej07;

public class Main {

    public static void main(String[] args) {
        int n1,n2;

        System.out.print("Introduce un número: ");
        n1=Entrada.entero();
        System.out.print("Introduce otro número: ");
        n2=Entrada.entero();

       // si ambos números son iguales diría que n2 es mayor que n1
       if(n1>n2)
          System.out.println(n1 + " es mayor que " + n2);
       else
          System.out.println(n2 + " es mayor que " + n1);
    }
}
```

8. Pedir dos números y decir cual es el mayor o si son iguales.

```
package bol01ej08;

public class Main {

    public static void main(String[] args) {
        int n1,n2;

        System.out.print("Introduce un número: ");
        n1=Entrada.entero();
        System.out.print("Introduce otro número: ");
        n2=Entrada.entero();

        if(n1==n2)
            System.out.println("Son iguales");
        else
         {
          if(n1>n2)
              System.out.println(n1 + " es mayor que " + n2);
          else
              System.out.println(n2 + " es mayor que " + n1);
        }
    }
}
```

9. Pedir dos números y mostrarlos ordenados de mayor a menor.

```
package bol01ej09;

public class Main {

    public static void main(String[] args) {
        int n1,n2;
```

```
        System.out.print("Introduce un número: ");
        n1=Entrada.entero();
        System.out.print("Introduce otro número: ");
        n2=Entrada.entero();

          if(n1>n2)
              System.out.println(n1 + " y " + n2);
          else
              System.out.println(n2 + " y " + n1);
    }
}
```

10. Pedir tres números y mostrarlos ordenados de mayor a menor.

```
package bol01ej10;

public class Main {

    public static void main(String[] args) {
        int a,b,c;

        System.out.print("Introduzca primer número: ");
        a=Entrada.entero();
        System.out.print("Introduzca segundo número: ");
        b=Entrada.entero();
        System.out.print("Introduzca tercer número: ");
        c=Entrada.entero();

        if(a>b && b>c)
            System.out.println( a+", "+b+", "+c);
        else{
            if(a>c && c>b)
                System.out.println(a+", "+c+", "+b);
```

```
            else{
               if(b>a && a>c)
                   System.out.println(b+", "+a+", "+c);
               else{
                  if(b>c && c>a)
                     System.out.println(b+", "+c+", "+a);
                  else{
                      if(c>a && a>b)
                         System.out.println(c+", "+a+", "+b);
                      else{
                         if(c>b && b>a)
                            System.out.println(c+", "+b+", "+a);
                        }
                  }
               }
            }
        }
    }
}
```

11. Pedir un número entre 0 y 9.999 y decir cuantas cifras tiene.

```
package bol01ej11;

public class Main {

    public static void main(String[] args) {
        int num;

        System.out.print("Introduzca un número entre 0 y 99.999: ");
        num=Entrada.entero();

        if(num<10)
            System.out.println("tiene 1 cifra");
```

```
        else{
            if(num<100)
                System.out.println("tiene 2 cifras");
            else{
                if(num<1000)
                    System.out.println("tiene 3 cifras");
                else{
                    if(num<10000)
                        System.out.println("tiene 4 cifras");
                    else{
                        if(num<100000)
                            System.out.println("tiene 5 cifras");
                        }
                    }
                }
            }
    }
}
```

12. Pedir un número entre 0 y 9.999 y mostrarlo con las cifras al revés.

```
package bol01ej12;

public class Main {

    public static void main(String[] args) {
        int num;
        int dm, um, c, d, u;

        // 9 9 . 9 9 9  a cada guarismo lo llamaremos:
        //dm um  c d u: dm (decenas de millar), um:(unidades de millar)
        //              c: (centenas), d: (decenas), u: (unidades)
```

```
        System.out.print("Introduzca un número entre 0 y 99.999: ");
        num=Entrada.entero();

        // unidad
        u = num % 10;
        num = num / 10;

        // decenas
        d = num % 10;
        num = num / 10;

        // centenas
        c = num % 10;
        num = num / 10;

        // unidades de millar
        um = num % 10;
        num = num / 10;

        // decenas de millar
        dm = num;

        // lo imprimimos al revés:
        System.out.println (u + " " + d + " " + c + " " + um + " " + dm);

        // otra forma de hacerlo es
        num = 10000*u + 1000*d + 100*c + 10*um + dm;
        System.out.println (num);
    }
}
```

13. Pedir un número entre 0 y 9.999, decir si es capicúa.

```
package bol01ej13;
```

```
public class Main {

    public static void main(String[] args) {
        int num;
        int dm, um, c, d, u;

        // 9 9 . 9 9 9  a cada guarismo lo llamaremos:
        //dm um  c d u: dm (decenas de millar), um:(unidades de millar)
        //              c: (centenas), d: (decenas), u: (unidades)

        System.out.print("Introduzca un número entre 0 y 99.999: ");
        num=Entrada.entero();

        // unidad
        u = num % 10;
        num = num / 10;

        // decenas
        d = num % 10;
        num = num / 10;

        // centenas
        c = num % 10;
        num = num / 10;

        // unidades de millar
        um = num % 10;
        num = num / 10;

        // decenas de millar
        dm = num;

        // el número será capicúa si las cifras son iguales dos a dos por los extremos
        // las centenas no las tenemos en cuenta
```

```
        if (dm == u && um == d)
            System.out.println ("el número es capicúa");
        else
            System.out.println ("el número NO es capicúa");

        // hay que tener en cuenta que en este ejercicio el número 121 es similar al 00121 y
        // resulta que 121 es capicúa, pero nuestro código lo identifica como NO capicúa. Ya
        // que trabajamos con el 00121. No tendremos en cuenta este pequeño error.
    }
}
```

```
package bol01ej13;

public class Main {

    public static void main(String[] args) {
        int num;
        int dm, um, c, d, u;

        boolean capicua = false; // suponemos que el número no es capicúa;

        // 9 9 . 9 9 9  a cada guarismo lo llamaremos:
        //dm um  c d u: dm (decenas de millar), um:(unidades de millar)
        //              c: (centenas), d: (decenas), u: (unidades)

        // En esta versión haremos que el número 121 ó el 33 sea visto como capicúa.
        // La idea es no tener en cuenta los ceros por la derecha.

        System.out.print("Introduzca un número entre 0 y 99.999: ");
        num=Entrada.entero();

        // unidad
        u = num % 10;
        num = num / 10;
```

```
        // decenas
        d = num % 10;
        num = num / 10;

        // centenas
        c = num % 10;
        num = num / 10;

        // unidades de millar
        um = num % 10;
        num = num / 10;

        // decenas de millar
        dm = num;

        //si el número tiene 5 cifras (dm, um, c, d, u)
        if (dm == u && um == d)
            capicua = true;

        //si el número tiene 4 cifras (0, um, c, d, u)
        if (dm == 0 && um == u && c == d)
            capicua = true;

        //si el número tiene 3 cifras (0, 0, c, d, u)
        if (dm == 0 && um==0 && c == u)
            capicua = true;

        //si el número tiene 2 cifras (0, 0, 0, d, u)
        if (dm == 0 && um == 0 && c == 0 && d == u)
            capicua = true;

        // se entiende que un número de una cifra no es capicúa

        if (capicua)
            System.out.println ("el número es capicúa");
```

```
        else
            System.out.println ("el número NO es capicúa");

    }
}
```

14. Pedir una nota de 0 a 10 y mostrarla de la forma: Insuficiente, Suficiente, Bien...

```
package bol01ej14;

public class Main {

    public static void main(String[] args) {
        int nota;

        System.out.print("Introduzca una nota: ");
        nota=Entrada.entero();


        // tanto los if's como los else's encierran a una sola instrucción
        // y no es necesario utilizar llaves { }

        if(nota>=0 && nota<5)
            System.out.println("INSUFICIENTE");
        else
            if(nota==5)
                System.out.println("SUFICIENTE");
            else
                if(nota==6)
                    System.out.println("BIEN");
                else
                    if(nota==7 || nota==8)
                        System.out.println("NOTABLE");
                    else
```

```
                        if(nota==9 || nota==10 )
                            System.out.println("SOBRESALIENTE");
        }
    }
```

```
package bol01ej14b;

public class Main {

    public static void main(String[] args) {
        int nota;

        System.out.print("Introduzca una nota: ");
        nota=Entrada.entero();

        switch(nota){
            case 0:
            case 1:
            case 2:
            case 3:
            case 4:
                System.out.println("INSUFICIENTE");
                break;
            case 5:
                System.out.println("SUFICIENTE");
                break;
            case 6:
                System.out.println("BIEN");
                break;
            case 7:
            case 8:
                System.out.println("NOTABLE");
                break;
            case 9:
```

```
            case 10:
                System.out.println("SOBRESALIENTE");
                break;
            default:
                System.out.println("ERROR");
                break;
        }
    }
}
```

15. Pedir el día, mes y año de una fecha e indicar si la fecha es correcta. Suponiendo todos los meses de 30 días.

```
package bol01ej15;

public class Main {

    public static void main(String[] args) {
        int dia,mes,año;

        // para que una fecha sea correcta se tiene que cumplir
        // día en el rango 1..30
        // mes en el rango 1..12
        // año cualquiera distinto del 0

        System.out.print("Introduzca día: ");
        dia=Entrada.entero();
        System.out.print("Introduzca mes: ");
        mes=Entrada.entero();
        System.out.print("Introduzca año: ");
        año=Entrada.entero();

        if (dia >= 1 && dia <=30)
            if (mes >= 1 && mes <= 12)
```

```
                if (año != 0)
                    System.out.println ("Fecha correcta");
                else
                    System.out.println ("Año incorrecto");
            else
                System.out.println("Mes incorrecto");
        else
            System.out.println("Día incorrecto");

    }
}
```

16. Pedir el día, mes y año de una fecha e indicar si la fecha es correcta. Con meses de 28, 30 y 31 días. Sin años bisiestos.

```
package bol01ej16;

public class Main {

    public static void main(String[] args) {
        int dia,mes,año;

        System.out.print("Introduzca día: ");
        dia=Entrada.entero();
        System.out.print("Introduzca mes: ");
        mes=Entrada.entero();
        System.out.print("Introduzca año: ");
        año=Entrada.entero();

        // el único año que no existe es el 0
        if(año==0)
            System.out.println("Fecha incorrecta");
        else{
            if(mes==2 && (dia>=1 && dia<=28))
```

```
                System.out.println(dia + "/" + mes + "/" + año+": Fecha correcta");
            else{
                if((mes==4 || mes==6 || mes==9 || mes==11) &&
                                            (dia>=1 && dia<=30))
                    System.out.println(dia + "/" + mes + "/" + año+": Fecha correcta");
                else{
                    if( (mes==1 || mes==3 || mes==5 || mes==7 || mes==8 || mes==10 || mes==12) &&
                                        (dia>=1 && dia<=31))
                        System.out.println(dia + "/" + mes + "/" + año+": Fecha correcta");
                    else
                        System.out.println("Fecha incorrecta");

                }
            }
        }
    }
}
```

```
package bol01ej16;

public class Main {

    public static void main(String[] args) {
        int dia,mes,año;

        int dias_del_mes;
        boolean fecha_correcta=true;

        System.out.print("Introduzca día: ");
        dia=Entrada.entero();
        System.out.print("Introduzca mes: ");
        mes=Entrada.entero();
        System.out.print("Introduzca año: ");
        año=Entrada.entero();
```

```
        dias_del_mes = 0; // si se utiliza un mes fuera del rango 1..12
                          // supondremos que los días del mes son 0.

        if(año==0) // el único año que no existe es el 0
            fecha_correcta = false;
        if (dia<1 || dia >31) // un día fuera del rango 1..31 no tiene sentido
            fecha_correcta = false;
        if (mes<1 || mes >12) // un mes fuera del rango 1..12 no tiene sentido
            fecha_correcta = false;

        if(mes==2 )
            dias_del_mes = 28;
        if(mes==4 || mes==6 || mes==9 || mes==11)
            dias_del_mes = 30;
        if(mes==1 || mes==3 || mes==5 || mes==7 || mes==8 || mes==10 || mes==12)
            dias_del_mes = 31;

        if (dia > dias_del_mes)
            fecha_correcta = false;

        if (fecha_correcta)
            System.out.println(dia + "/" + mes + "/" + año+": Fecha correcta");
        else
            System.out.println("Fecha incorrecta");
    }
}
```

17. Pedir el día, mes y año de una fecha correcta y mostrar la fecha del día siguiente. suponer que todos los meses tienen 30 días.

```
package bol01ej17;

public class Main {
```

```
public static void main(String[] args) {
    int dia,mes,año;

    System.out.print("Introduzca día: ");
    dia=Entrada.entero();
    System.out.print("Introduzca mes: ");
    mes=Entrada.entero();
    System.out.print("Introduzca año: ");
    año=Entrada.entero();

    // suponemos que la fecha introducida es correcta

    // incrementamos el día
    dia ++;

    // si el día supera 30, lo reiniciamos a 1 e incrementamos el mes
    if (dia >= 30)
    {
        dia = 1;
        mes ++;

        // si el mes supera 12, lo reiniciamos a 1 e incrementamos el año
        if (mes >= 12)
        {
            mes = 1;
            año ++;
        }
    }
    // habría que tener en cuenta que el año pasa del -1 al +1
    // en este código pasaríamos del año -1 al 0 (que nunca existió)
    // para corregirlo:

    if (año == 0)
        año = 1;
```

```
        System.out.println (dia + "/"+ mes + "/" + año);
    }
}
```

18. Ídem que el ej. 17, suponiendo que cada mes tiene un número distinto de días (suponer que febrero tiene siempre 28 días).

```
package bol01ej18;

public class Main {

    public static void main(String[] args) {
        int dia,mes,año;
        int dias_del_mes=0; // guardaremos el número de días que tiene el mes

        System.out.print("Introduzca día: ");
        dia=Entrada.entero();
        System.out.print("Introduzca mes: ");
        mes=Entrada.entero();
        System.out.print("Introduzca año: ");
        año=Entrada.entero();

        // suponemos que la fecha introducida es correcta

        if(mes==2 )
            dias_del_mes = 28;
        if(mes==4 || mes==6 || mes==9 || mes==11)
            dias_del_mes = 30;
        if(mes==1 || mes==3 || mes==5 || mes==7 || mes==8 || mes==10 || mes==12)
            dias_del_mes = 31;

        // incrementamos el día
        dia ++;
```

```
        // si el día supera el número de días del mes,
        // lo reiniciamos a 1 e incrementamos el mes

        if (dia >= dias_del_mes)
        {
            dia = 1;
            mes ++;

            // si el mes supera 12, lo reiniciamos a 1 e incrementamos el año
            if (mes >= 12)
            {
                mes = 1;
                año ++;
            }
        }
        // habría que tener en cuenta que el año pasa del -1 al +1
        // en este código pasaríamos del año -1 al 0 (que nunca existió)
        // para corregirlo:

        if (año == 0)
            año = 1;

        System.out.println (dia + "/"+ mes + "/" + año);
    }
}
```

19. Pedir dos fechas y mostrar el número de días que hay de diferencia. Suponiendo todos los meses de 30 días.

```
package bol01ej19;

public class Main {
```

```
    public static void main(String[] args) {
        int dia1,mes1,año1;
        int dia2,mes2,año2;
        int total_dias;

        System.out.println ("Fecha 1:");
        System.out.print("Introduzca día: ");
        dia1=Entrada.entero();
        System.out.print("Introduzca mes: ");
        mes1=Entrada.entero();
        System.out.print("Introduzca año: ");
        año1=Entrada.entero();

        System.out.println ("Fecha 2:");
        System.out.print("Introduzca día: ");
        dia2=Entrada.entero();
        System.out.print("Introduzca mes: ");
        mes2=Entrada.entero();
        System.out.print("Introduzca año: ");
        año2=Entrada.entero();

        // suponemos que las fecha introducidas son correctas

        // convertimos las dos fechas a días y calculamos la diferencia
        total_dias = dia2-dia1 + 30*(mes2-mes1)+365*(año2-año1);

        System.out.println ("Días de diferencia: " + total_dias);
    }
}
```

20. Pedir una hora de la forma hora, minutos y segundos, y mostrar la hora en el segundo siguiente.

```
package bol01ej20;
```

```
public class Main {

    public static void main(String[] args) {
        int h,m,s; // hora, minutos y segundos

        System.out.print("Introduzca hora: ");
        h=Entrada.entero();
        System.out.print("Introduzca minutos: ");
        m=Entrada.entero();
        System.out.print("Introduzca segundos: ");
        s=Entrada.entero();

        // suponemos que la hora introducida es correcta

        // incrementamos los segundos
        s ++;

        // si los segundos superan 59, los reiniciamos a 0 e incrementamos los minutos
        if (s >= 60)
        {
            s = 0;
            m ++;

            // si los minutos superan 59, los reiniciamos a 0 e incrementamos la hora
            if (m >= 60)
            {
                m = 0;
                h ++;
                // si la hora supera 23, la reiniciamos a 0
                if (h>=24)
                    h=0;
            }
        }
        System.out.println ("Fecha: "+ h + ":"+ m + ":" + s);
    }
}
```

21. Pedir una nota numérica entera entre 0 y 10, y mostrar dicha nota de la forma: cero, uno, dos, tres...

```
package bol01ej21;

public class Main {

    public static void main(String[] args) {
        int num;
        System.out.print("Introduzca una nota numérica entre 0 y 10: ");
        num=Entrada.entero();
        switch(num){
            case 0:
                System.out.println("CERO");
                break;

            case 1:
                System.out.println("UNO");
                break;

            case 2:
                System.out.println("DOS");
                break;

            case 3:
                System.out.println("TRES");
                break;

            case 4:
                System.out.println("CUATRO");
                break;

            case 5:
                System.out.println("CINCO");
                break;
```

```
            case 6:
                System.out.println("SEIS");
                break;

            case 7:
                System.out.println("SIETE");
                break;

            case 8:
                System.out.println("OCHO");
                break;

            case 9:
                System.out.println("NUEVE");
                break;

            case 10:
                System.out.println("DIEZ");
                break;
        }
    }
}
```

22. Pedir un número de 0 a 99 y mostrarlo escrito. Por ejemplo, para 56 mostrar: cincuenta y seis.

```
package bol01ej22;

public class Main {

    public static void main(String[] args) {
        int num;
        int unidades, decenas;
        // esta versión muesrta 11 como diez y uno.
        // es una forma de hacerlo bastante burda.
```

```
        // se puede poner algunos condicionales para los números especiales: 11,12,...
        // y otro condicional para mostrar "y"

        System.out.print("Introduzca un número (0 a 99): ");
        num=Entrada.entero();

        unidades = num % 10;
        decenas = num / 10;

        switch(decenas){
            case 0:
                System.out.print("");
                break;

            case 1:
                System.out.print("diez");
                break;

            case 2:
                System.out.print("veinte");
                break;

            case 3:
                System.out.print("treinta");
                break;

            case 4:
                System.out.print("cuarenta");
                break;

            case 5:
                System.out.print("cincuenta");
                break;

            case 6:
                System.out.print("sesenta");
```

```
                break;

            case 7:
                System.out.print("setenta");
                break;

            case 8:
                System.out.print("ochenta");
                break;

            case 9:
                System.out.print("noventa");
                break;
        }

        System.out.print (" y ");

        switch(unidades){
            case 0:
                System.out.println("");
                break;

            case 1:
                System.out.println("uno");
                break;

            case 2:
                System.out.println("dos");
                break;

            case 3:
                System.out.println("tres");
                break;

            case 4:
                System.out.println("cuatro");
```

```
                break;

            case 5:
                System.out.println("cinco");
                break;

            case 6:
                System.out.println("seis");
                break;

            case 7:
                System.out.println("siete");
                break;

            case 8:
                System.out.println("ocho");
                break;

            case 9:
                System.out.println("nueva");
                break;
        }
    }
}
```

Boletín 2
Condicionales y bucles

1. Leer un número y mostrar su cuadrado, repetir el proceso hasta que se introduzca un número negativo.

```
package bol02ej01;

public class Main {

    public static void main(String[] args) {
        int num,cuadrado;
        // num guardará el número que leamos
        // y cuadrado guardará el cuadrado de num

        System.out.print("Introduzca número: ");
        num=Entrada.entero();

        while(num>=0){ // repetimos el proceso mientras el número leído no sea negativo
            cuadrado=num*num;
            System.out.println(num+ "² es igual a "+ cuadrado);
            System.out.print("Introduzca otro número: ");
            num=Entrada.entero(); // volvemos a leer num
        }
    }
}
```

2. Leer un número e indicar si es positivo o negativo. El proceso se repetirá hasta que se introduzca un 0.

```
package bol02ej02;

public class Main {

    public static void main(String[] args) {
        int num;
        System.out.print("Introduzca un número: ");
        num=Entrada.entero();

        while(num!=0) // mientras num sea distinto de 0
        {
            if(num>0)
                // mayor que cero: positivo
                System.out.println("Positivo");
            else
                // si no es positivo: es negativo
                System.out.println("Negativo");

            // repetimos el proceso y volvemos a leer num
            System.out.print("Introduzca otro número: ");
            num=Entrada.entero();
        }
        // al salir del mientras tenemos la certeza que num es 0
    }
}
```

3. Leer números hasta que se introduzca un 0. Para cada uno indicar si es par o impar.

```
package bol02ej03;
```

```
public class Main {

    public static void main(String[] args) {
        int num;
        System.out.print("Introduzca un número: ");
        num=Entrada.entero();

        while(num!=0) // mientras num sea distinto de 0
        {
            if(num%2 == 0)
                // si el resto de dividir entre dos es cero: esto indica que es par
                System.out.println("Par");
            else
                // en caso contrario: impar
                System.out.println("Impar");

            // repetimos el proceso y volvemos a leer num
            System.out.print("Introduzca otro número: ");
            num=Entrada.entero();
        }
        // al salir del mientras tenemos la certeza que num es 0
    }
}
```

4. Pedir números hasta que se teclee uno negativo, y mostrar cuántos números se han introducido.

```
package bol02ej04;

public class Main {

    public static void main(String[] args) {
        int num, contador;
```

```
            // num guardará los números introducidos
            // y contador se incrementará para llevar la cuenta de los números introducidos

        System.out.print("Introduzca un número: ");
        num=Entrada.entero();

        contador=0;   // al comienzo el número de números introducidos es 0

        while(num>0) // mientras num sea positiva
        {
            contador =contador+1; // contador toma el valor que tuviera en este momento más uno
                                  // en definitiva: contador se incrementa en uno

            System.out.print("Introduzca otro número: ");
            num=Entrada.entero();
        }

        System.out.println("Se han introducido: " +contador + " números");
        // sin tener en cuenta el último número negativo.
    }
}
```

5. Realizar un juego para adivinar un número. Para ello pedir un número N, y luego ir pidiendo números indicando "mayor" o "menor" según sea mayor o menor con respecto a N. El proceso termina cuando el usuario acierta.

```
package bol02ej05;

public class Main {

    public static void main(String[] args) {
        int n, num;
```

```
        // n es el número que hay que acertar
        // num guarda los números introducidos

        System.out.print("Introduce N: ");
        n =Entrada.entero();

        System.out.print("Introduce número: ");
        num=Entrada.entero();

        while(num!=n) // mientras no coincidan ambos números
        {
            if(num>n)
                System.out.println("menor");
            else
                System.out.println("mayor");

            System.out.print("Introduce número: ");
            num=Entrada.entero();
        }

        // al salir del mientras tenemos la certeza que num es igual a n

        System.out.println("acertaste...");
    }
}
```

```
package bol02ej05;

public class Main {

    public static void main(String[] args) {
        int n, num;
        // n es el número que hay que acertar
        // num guarda los números introducidos
```

```
        n=(int)(Math.random()*100)+1;
        // en lugar de pedir n... podemos hacer que se n tome un valor
        // aleatorio entre 1 y 100.
        // Así el juego es algo más entretenido.


        System.out.print("Introduce número: ");
        num=Entrada.entero();

        while(num!=n) // mientras no coincidan ambos números
        {
            if(num>n)
                System.out.println("menor");
            else
                System.out.println("mayor");

            System.out.print("Introduce número: ");
            num=Entrada.entero();
        }

        // al salir del mientras tenemos la certeza que num es igual a n

        System.out.println("acertaste...");
    }
}
```

6. Pedir números hasta que se teclee un 0, mostrar la suma de todos los números introducidos.

```
package bol02ej06;

public class Main {

    public static void main(String[] args) {
```

```
        int num,suma;
        suma=0;

        do
        {
            System.out.print("Introduzca un número: ");
            num=Entrada.entero();

            suma=suma+num;
        }

        while(num!=0);


        System.out.println("La suma de todos los números es: "+suma);

    }
}
```

7. Pedir números hasta que se introduzca uno negativo, y calcular la media.

```
package bol02ej07;

public class Main {

    public static void main(String[] args) {
        int num, suma, elementos;
        float media; // la media puede tener decimales


        // num: guardará los números introducidos por el usuario
        // suma: almacenará la suma de todos los números introducidos
        // elementos: será un contador que indicará el números de números 8o elementos) introducidos
```

```
        System.out.print("Introduzca un número: ");
        num=Entrada.entero();

        suma= 0;
        elementos= 0;

        while(num>=0) // nos interesan los positivos y el cero
        {
            suma+=num;
            elementos++;

            System.out.print("Introduzca otro número: ");
            num=Entrada.entero();
        }


        if (elementos == 0) // daría un error de división por cero
            System.out.println("Imposible hacer la media");
        else
        {
            media= (float)suma/elementos;
            System.out.println("La media es de: " + media);
        }
    }
}
```

8. Pedir un número N, y mostrar todos los números del 1 al N.

```
package bol02ej08;

public class Main {

    public static void main(String[] args) {
```

```
         int i,num;

        System.out.print("Introduce un número: ");
        num=Entrada.entero();

        i=1;
        // i es el contador que tomará los valores de 1 a n

        while(i<=num){
            System.out.println(i);
            i++;
        }
    }
}
```

9. Escribir todos los números del 100 al 0 de 7 en 7.

```
package bol02ej09;

public class Main {

    public static void main(String[] args) {

        // inicializamos la i a 100
        // mientras la i sea mayor o igual a 0
        // y en cada vuelta del for la i se decrementa en 7
        for (int i=100;i>=0;i-=7)
            System.out.println(i);

        // el for al llevar una sola instrucción en su cuerpo de ejecución
        // no precisa de llaves { }
    }
}
```

10. Pedir 15 números y escribir la suma total.

```
package bol02ej10;

public class Main {

    public static void main(String[] args) {
        int num,suma_total;

        suma_total=0;

        for (int i=1;i<=15;i++)
        {
            System.out.print("Introduzca número: ");
            num=Entrada.entero();

            suma_total=suma_total+num;
        }
        System.out.println("La suma total es de: "+suma_total);
     }
}
```

11. Diseñar un programa que muestre el producto de los 10 primeros números impares.

```
package bol02ej11;

public class Main {

    public static void main(String[] args) {
        long producto=1;  // producto guardará la multiplicación de los 10 primeros números impares.
                          // es muy importante acordarse de inicializarlo a 1. Ya que si lo hacemos a 0,
```

```
el producto siempre valdrá 0.


        // para calcular los 10 primeros números impares utilizamos un for que:
        //
        // comience en 1
        // y en cada vuelta se incremente en 2, así obtenemos
        // 1, 3, 5, 7, 9, 11, 13, 15, 17, 19.

        for (int i=1; i<20; i+=2) {
            producto=producto*i;

        }

        System.out.println("La multiplicación de los 10 primeros impares: " + producto);

    }
}
```

12. Pedir un número y calcular su factorial.

```
package bol02ej12;

public class Main {

    public static void main(String[] args) {
        // long factorial; con long se puede calcular hasta el factorial de 25
        double factorial;
        int num;


        System.out.print("Introduce un número: ");
        num=Entrada.entero();
```

```
        factorial=1; // es importante inicializarlo a 1, ya que multiplicará-


        // por ejemplo: el factorial de 10 es:
        // 10*9*8*7*6*5*4*3*2*1

        for (int i=num;i>0;i--)
        {
            factorial=factorial*i;

        }

        System.out.println("El factorial de " + num + " es: " + factorial);
    }
}
```

13. Pedir 10 números. Mostrar la media de los números positivos, la media de los números negativos y la cantidad de ceros.

```
package bol02ej13;

public class Main {

    public static void main(String[] args) {
        int num;
        int cont_ceros; // el contador de ceros
        int cont_pos; // contador de positivos
        int cont_neg; // contador de negativos
        int suma_pos,suma_neg; // suma de los números positivos y negativos

        float media_pos,media_neg; // las medias 8positivas y negativa9 pueden tener decimales
```

```
cont_ceros=0;
cont_pos=0;
cont_neg=0;

suma_pos=0;
suma_neg=0;

for (int i=1;i<=10;i++)
{
    System.out.print("Introduce número: ");
    num=Entrada.entero();

    if(num==0)
        cont_ceros++;
    else
    {
        if(num>0)
        {
            cont_pos++;
            suma_pos+=num;
        }
        else
        {
            cont_neg++;
            suma_neg+=num;
        }
    }
}

// tratamos los ceros
System.out.println("El número de ceros introducidos es de: "+cont_ceros);

//Tratamos los positivos
if (cont_pos ==0)
    System.out.println("No se puede hacer la media de los positivos");
```

```
        else
        {
            media_pos= (float)suma_pos/cont_pos;
            System.out.println("Media de los positivos: "+ media_pos);
        }

        // tratamos los negativos
        if (cont_pos ==0)
            System.out.println("No se puede hacer la media de los negativos");
        else
        {
            media_neg= (float)suma_neg/cont_neg;
            System.out.println("Media de los negativos: "+ media_neg);
        }
    }
}
```

14. Pedir 10 sueldos. Mostrar su suma y cuantos hay mayores de 1000€.

```
package bol02ej14;

public class Main {

    public static void main(String[] args) {
         int sueldo,suma,mayor_1000;
        suma=0;
        mayor_1000=0;
        for (int i=1;i<10;i++)
        {
            System.out.print("Escribe un sueldo: ");
            sueldo =Entrada.entero();
            if(sueldo>1000)
                mayor_1000++;
```

```
            suma=suma+sueldo;
        }
        System.out.println("Mayores de 1000 hay: "+mayor_1000);
        System.out.println("la suma es de: "+suma);
    }
}
```

15. Dadas las edades y alturas de 5 alumnos, mostrar la edad y la estatura media, la cantidad de alumnos mayores de 18 años, y la cantidad de alumnos que miden más de 1.75.

```
package bol02ej15;

public class Main {

    public static void main(String[] args) {
        int edad,media_edad,suma_edad,mayor_edad,mayor_175; //mayor_edad: mayores de 18 años
        double altura,media_altura,suma_alt;

        mayor_edad=0;
        media_altura=0;
        mayor_175=0;
        suma_edad=0;
        suma_alt=0;

        for (int i=1;i<5;i++){
            System.out.println("Alumno " +i);
            System.out.print("Introduzca edad: ");
            edad=Entrada.entero();
            System.out.print("Introduzca altura: ");
            altura=Entrada.real();
            if(edad>18)
                mayor_edad++;
```

```
            if(altura>1.75)
                mayor_175++;

            suma_edad=suma_edad+edad;
            suma_alt=suma_alt+altura;
        }

        media_edad=suma_edad/5;
        media_altura=suma_alt/5;

        System.out.println("\n\nLa edad media es de: " +media_edad);
        System.out.println("La altura media es de: " +media_altura);
        System.out.println("Mayor de 18 años: " +mayor_edad);
        System.out.println("Mayor de 1.75: " +mayor_175);
    }
}
```

16. Pide un número (que debe estar entre 0 y 10) y mostrar la tabla de multiplicar de dicho número.

```
package bol02ej16;

public class Main {

    public static void main(String[] args) {
        int num;

        do
        {
            System.out.print("Introduce número (de 0 a 10): ");
            num=Entrada.entero();
        }
```

```
        while ( !  (0<=num && num<=10));

        System.out.println("\n\nTabla del " + num);

        for (int i=1;i<=10;i++)
        {
            System.out.println(num + " x " + i + " = " + num*i);

        }
    }
}
```

17. Una empresa que se dedica a la venta de desinfectantes necesita un programa para gestionar las facturas. En cada factura figura: el código del artículo, la cantidad vendida en litros y el precio por litro.
Se pide de 5 facturas introducidas: Facturación total, cantidad en litros vendidos del artículo 1 y cuantas facturas se emitieron de más de 600 €.

```
package bol02ej17;

public class Main {

    public static void main(String[] args) {
        int codigo;     // el código del artículo en cada factura
        int litros;     // la cantidad de litros en cada factura
        float precio;   // el precio por litro en cada factura

        float importe_factura;  // guardará el importe de la factura con la que estemos trabajando

        float facturacion_total;    // el importe de todas las facturas
        int litros_cod1;            // el total de litros vendidos del producto 1 en todas las facturas
```

```
        int mas_600;                // contador que sirve para llevar la cuenta de cuantas facturas hay de
más de 600 €


        facturacion_total = 0;
        litros_cod1 = 0;
        mas_600 = 0;

        for (int i=1;i<=5;i++)
        {
            System.out.println("Factura nº " + i);
            System.out.print("código de producto: ");
            codigo=Entrada.entero();
            System.out.print("cantidad (litros): ");
            litros=Entrada.entero();
            System.out.print("precio (litro): ");
            precio= (float)Entrada.real();


            importe_factura = litros*precio;

            facturacion_total += importe_factura;

            if (codigo == 1)
                litros_cod1 += litros;

            if(importe_factura >= 600)
                mas_600 ++;

        }

        System.out.println("\n\n\nResumen de ventas\n");
        // facturación total
        System.out.println("La facturación total es de: " +facturacion_total + "€");
```

```
        // litros del articulo 1
        System.out.println("Ventas del producto 1: " + litros_cod1 + " litros");


        // factura de mas de 600 euros
        System.out.println("Factura superior a 600€: " + mas_600);
    }
}
```

18. Igual que el anterior pero suponiendo que no se introduce el precio por litro. Solo existen tres productos con precios:
    1- 0,6 €/litro, 2- 3 €/litro y 3- 1,25 €/litro.

```
package bol02ej18;

public class Main {

    public static void main(String[] args) {
        int codigo;     // el código del artículo en cada factura
        int litros;     // la cantidad de litros en cada factura
        float precio;   // ahora el precio no se pide por teclado

        float importe_factura;  // guardará el importe de la factura con la que estemos trabajando

        float facturacion_total;    // el importe de todas las facturas
        int litros_cod1;            // el total de litros vendidos del producto 1 en todas las facturas

        int mas_600;                // contador que sirve para llevar la cuenta de cuantas facturas hay de
más de 600 €


        facturacion_total = 0;
```

```
        litros_cod1 = 0;
        mas_600 = 0;

        for (int i=1;i<=5;i++)
        {
            System.out.println("Factura nº " + i);
            System.out.print("código de producto: ");
            codigo=Entrada.entero();
            System.out.print("cantidad (litros): ");
            litros=Entrada.entero();


            switch (codigo)
            {
                case 1:
                    precio = 0.6f;
                    break;

                case 2:
                    precio = 3f;
                    break;

                case 3:
                    precio = 1.25f;
                    break;
                default:
                    precio = 0; // este caso no debe darse
            }



            importe_factura = litros*precio;

            facturacion_total += importe_factura;

            if (codigo == 1)
```

```
                litros_cod1 += litros;

            if(importe_factura >= 600)
                mas_600 ++;
        }

        System.out.println ("\n\n\nResumen de ventas\n");
        // facturación total
        System.out.println("La facturación total es de: " +facturacion_total + "€");


        // litros del articulo 1
        System.out.println("Ventas del producto 1: " + litros_cod1 + " litros");


        // factura de mas de 600 euros
        System.out.println("Factura superior a 600€: " + mas_600);

    }
}
```

19. Dadas 6 notas, escribir la cantidad de alumnos aprobados, condicionados (=4) y suspensos.

```
package bol02ej19;

public class Main {

    public static void main(String[] args) {
        int nota,aprobados,suspensos,condicionados;

        aprobados=0;
        suspensos=0;
```

```
        condicionados=0;

        for (int i=1;i<=6;i++)
        {
            System.out.print("Introduzca nota entre 0 y 10: ");
            nota=Entrada.entero();

            if(nota == 4)
                condicionados++;
            else
                if(nota >= 5)
                    aprobados++;
                else
                    if(nota < 4) // este if sobra, ya que es el único caso posible
                        suspensos++;


        }

        System.out.println("Aprobados: " +aprobados);
        System.out.println("Suspensos: " +suspensos);
        System.out.println("Condicionados: "+condicionados);
    }
}
```

20. Pedir un número N, introducir N sueldos, y mostrar el sueldo máximo.

```
package bol02ej20;

public class Main {

    public static void main(String[] args) {
```

```
        int sueldo, sueldo_max;
        int n;


        sueldo_max = 0; // como los sueldos son positivos, inicializamos el sueldo
                        // máximo a cero.

        System.out.print("Número de sueldos: ");
        n = Entrada.entero();

        System.out.println("--------");

        for (int i=1;i<=n;i++)
        {
            System.out.print("Introduce sueldo: ");
            sueldo=Entrada.entero();

            if (sueldo > sueldo_max)
                sueldo_max = sueldo;
                // si leemos un sueldo mayor que el máximo, este sueldo será el nuevo máximo.
        }

        System.out.println("\nEl sueldo máximo es: " +sueldo_max);
    }
}
```

```
package bol02ej20;

public class Main {

    public static void main(String[] args) {
        int sueldo, sueldo_max=0;
        int n;
```

```
        boolean primer_sueldo_asignado;
                        // esta bandera indica si hemos asignado el primer sueldo como sueldo máximo
                        // con esto podremos tener sueldos negativos

        primer_sueldo_asignado = false;


        System.out.print("Número de sueldos: ");
        n = Entrada.entero();

        System.out.println("--------");

        for (int i=1;i<=n;i++)
        {
            System.out.print("Introduce sueldo: ");
            sueldo=Entrada.entero();

            if (primer_sueldo_asignado == false)
            {
                // asignamos como máximo el primer sueldo leído
                sueldo_max = sueldo;
                primer_sueldo_asignado = true;
            }

            if (sueldo > sueldo_max)
                sueldo_max = sueldo;
                // si leemos un sueldo mayor que el máximo, este sueldo será el nuevo máximo.
        }

        System.out.println("\nEl sueldo máximo es: " +sueldo_max);
    }
}
```

21. Pedir 10 números, y mostrar al final si se ha introducido alguno negativo.

```
package bol02ej21;

public class Main {

    public static void main(String[] args) {
        int num;
        boolean hay_negativo;
            // la variable hay_negativo según los dos posibles valores indica:
            // false: no se ha introducido ningún número negativo
            // true: al menos hay un número negativo

        hay_negativo =false;
        // suponemos que no habrá ningún negativo

        for (int i=1;i<=10;i++)
        {
            System.out.print("Introduce número: ");
            num=Entrada.entero();

            if(num<0)
                hay_negativo =true;
                // si num es menor que cero, cambiamos el valor de hay_negativo a true

        }

        if(hay_negativo == true)
            System.out.println("Se ha introducido algún número negativo");
        else
            System.out.println("No hay ningún número negativo");

    }
}
```

22. Pedir 5 calificaciones de alumnos y decir al final si hay algún suspenso.

```
package bol02ej22;

public class Main {

    public static void main(String[] args) {
        int notas;
        boolean suspensos;

        suspensos=false; // suponemos que en principio no hay ningún suspenso

        for (int i=0;i<5;i++)
        {
            System.out.print("Introduzca nota (de 0 a 10): ");
            notas=Entrada.entero();

            if(notas<5)
                suspensos=true;
        }

        if(suspensos)
            System.out.println("Hay alumnos suspensos");
        else
            System.out.println("No hay suspensos");
    }
}
```

23. Pedir 5 números e indicar si alguno es múltiplo de 3.

```
package bol02ej23;

public class Main {
```

```
    public static void main(String[] args) {
        int num;
        boolean multiplo_3;

        multiplo_3=false;

        for (int i=0;i<5;i++){
            System.out.print("Introduzca número: ");
            num=Entrada.entero();

            if(num %3 == 0)
                multiplo_3=true; // si se ejecuta esta instrucción significa que al menos hay un múltiplo
de 3.
        }

        if(multiplo_3 == false)
            System.out.println("no existen múltiplos de 3");
        else
            System.out.println("Hay múltiplos de 3");
    }
}
```

Boletín 3
Bucles anidados

1. Realiza detenidamente una traza al siguiente programa y muestra cual seria la salida por pantalla:

```
PROGRAMA ej_1
      VARIABLES
            suma, i, j: ENTERO
      COMIENZO
            PARA i <- 1 HASTA 4
                  PARA j <- 3 HASTA 0 INC -1
                        suma <- i*10+j
                        escribir (suma)
                  FIN PARA
            FIN PARA
FIN
```

```
package bol03ej01;

public class Main {
    public static void main(String[] args) {
        int suma;
        for (int i=0;i<4;i++){
            for (int j=3;j>0;j--){
                suma=i*10+j;
                System.out.println(suma);
            }
        }
    }
}
```

2. Realiza una traza del siguiente algoritmo y muestra la salida generada por pantalla.

```
PROGRAMA ej_1
      VARIABLES
            i, j: ENTERO
      COMIENZO
            PARA i <- 1 HASTA 3
                  j <- i+1
                  MIENTRAS j < 4
                        escribir (j-i)
                        j <- j+1
                  FIN MIENTRAS
            FIN PARA
FIN
```

```
package bol03ej02;

public class Main {

    public static void main(String[] args) {
        int j;
        for (int i=0;i<3;i++){
            j=i+1;
            while(j<4){
                System.out.println(j-i);
                j++;
            }
        }
    }
}
```

3. Diseña una aplicación que muestre las tablas de multiplicar del 1 al 10.

```
package bol03ej03;

public class Main {

    public static void main(String[] args) {
        int tabla,i;

        for (tabla=1; tabla<=10; tabla++)
        {
            System.out.println ("\n\nTabla del " +tabla);
            System.out.println ("----------------");

            for (i=1;i<=10;i++)
```

```
            {
                System.out.println (tabla + " x " + i + " = " + tabla*i);
            }
        }
    }
}
```

4. Dibuja un cuadrado de n elementos de lado utilizando *.

```
package bol03ej04;

public class Main {

    public static void main(String[] args) {
        int n; // tamaño del lado
        int fila, col;


        System.out.print ("Lado del cuadrado: ");
        n = Entrada.entero();

        for (fila=1; fila<=n; fila++)
        {
            for (col=1; col<=n; col++)
                System.out.print ("* ");
            System.out.println ("");
        }
    }
}
```

5. Necesitamos mostrar un contador con 5 dígitos (X-X-X-X-X), que muestre los números del 0-0-0-0-0 al 9-9-9-9-9, con la particularidad que cada vez que aparezca un 3 lo sustituya por una **E**.

```
package bol03ej05;

public class Main {

    public static void main(String[] args) {
        for (int i=0;i<=9;i++){
            for (int j=0;j<=9;j++){
                for (int k=0;k<=9;k++){
                    for (int l=0;l<=9;l++){
                        for (int m=0;m<=9;m++){

                            if(i==3)
                                System.out.print("E");
                            else
                                System.out.print(i);

                            if(j==3)
                                System.out.print("E");
                            else
                                System.out.print(j);

                            if(k==3)
                                System.out.print("E");
                            else
                                System.out.print(k);

                            if(l==3)
                                System.out.print("E");
                            else
                                System.out.print(l);

                            if(m==3)
```

```
                                System.out.print("E");
                            else
                                System.out.print(m);

                            System.out.println(" ");

                        }
                    }
                }
            }
        }
    }
}
```

6. Realizar un programa que nos pida un número n, y nos diga cuantos números hay entre 1 y n que son primos.

```
package bol03ej06;

public class Main {

    public static void main(String[] args) {
        int j,num,cont_pri;
        boolean primo;

        cont_pri=0;
        System.out.print("Introduce numero: ");
        num=Entrada.entero();

        // vamos procesando todos los números entre 1..num
        for(int i=1;i<=num;i++){
```

```
            // para cada número i, calcularemos si es primo
            // veremos si el número i es divisible en el rango 2..i-1
            // El bucle while se puede hacer con menos vuelta... consultar algoritmos para primos

            primo=true;
            j=2;
            while (j<=i-1 && primo==true)
            {
                if (i%j==0)
                    primo=false;
            j++;
            }

            if (primo==true){
                cont_pri++;  // si es primo incrementamos el contador de primos
                System.out.println(i+(" es primo"));
            }
        }
        System.out.println("En el rango 1.." + num + ", hay "+ cont_pri + " números primos");
    }
}
```

Boletín 4
Tablas

1. Leer 5 números y mostrarlos en el mismo orden introducido.

```
package bol04ej01;

public class Main {

    public static void main(String[] args) {
         int t[];

         t = new int[5];

        for (int i=0;i<5;i++)
        {
            System.out.print("Introduzca un número: ");
            t[i]=Entrada.entero();
        }

        System.out.println("Los números son:");

        for (int i=0;i<5;i++)
            System.out.println(t[i]);
   }
}
```

2. Leer 5 números y mostrarlos en orden inverso al introducido.

```
package bol04ej02;

public class Main {

    public static void main(String[] args) {
        int t[]=new int[5];

        for (int i=0;i<5;i++)
        {
            System.out.print("Introduzca un número: ");
            t[i]=Entrada.entero();
        }

        System.out.println("Los números (en orden inverso):");
        for (int i=4;i>=0;i--)
            System.out.println(t[i]);
   }
}
```

3. Leer 5 números por teclado y a continuación realizar la media de los números positivos, la media de los negativos y contar el número de ceros.

```
package bol04ej03;

public class Main {

    public static void main(String[] args) {
        int t[]=new int [5];
        int suma_pos,cont_pos,suma_neg,cont_neg,cont_cero;
```

```
        suma_pos=0;
        cont_pos=0;
        suma_neg=0;
        cont_neg=0;
        cont_cero=0;


        // utilizamos un bucle para leer los datos y otro para procesarlos
        // se podrían hacer ambas operaciones, leer y procesar, en un solo bucle
        for (int i=0;i<5;i++){
            System.out.print("Introduzca un número: ");
            t[i]=Entrada.entero();
        }

        for (int i=0;i<5;i++){
            if(t[i]==0)
                cont_cero++;
            else{
                if(t[i]>0){
                    suma_pos=suma_pos+t[i];
                    cont_pos++;
                }
                else{
                    suma_neg=suma_neg+t[i];
                    cont_neg++;
                }
            }
        }
        if(cont_pos==0)
            System.out.println("No se puede realizar la media de números positivos");
        else
            System.out.println("La media de los positivos: "+(float)suma_pos/cont_pos);

        if(cont_neg==0)
            System.out.println("No se puede realizar la media de números negativos");
        else
```

```
            System.out.println("La media de los negativos: " + (float)suma_neg/cont_neg);


        System.out.println("La cantidad de cero es de: " + cont_cero);
    }
}
```

4. Leer 10 números enteros. Debemos mostrarlos en el siguiente orden: el primero, el último, el segundo, el penúltimo, el tercero, etc.

```
package bol04ej04;

public class Main {

    public static void main(String[] args) {
        int i, t[];

        t = new int[10];


        for (i=0;i<10;i++){
            System.out.print("Introduzca numero: ");
            t[i]=Entrada.entero();
        }


        System.out.println("El resultado es:");

        for (i=0;i<=4;i++){
            System.out.println (t[i]); // mostramos el i-ésimo número por el principio
            System.out.println(t[9-i]); // y el i-ésimo por el final
        }
```

```
        // como en cada vuelta de for se muestran dos números
        // para mostrarlos todos, solo necesitaremos la mitad de vueltas.

    }
}
```

5. Leer por teclado dos tablas de 10 números enteros y mezclarlas en una tercera de la forma: el 1° de A, el 1° de B, el 2° de A, el 2° de B, etc.

```
package bol04ej05;

public class Main {

    public static void main(String[] args) {
        int a[], b[], c[];
        int i,j;

        a=new int[10];
        b=new int[10];

        // la tabla c tendrá que tener el doble de tamaño que a y b.
        c = new int [20];


        // leemos la tabla a
        System.out.println("Leyendo la tabla a");

        for (i=0;i<10;i++){
            System.out.print("número: ");
            a[i]=Entrada.entero();
        }

        // leemos la tabla b
```

```
        System.out.println("Leyendo la tabla b");

        for (i=0;i<10;i++){
            System.out.print("número: ");
            b[i]=Entrada.entero();
        }

        // asignaremos los elementos de la tabla c
        // para las tablas a y b utilizaremos como índice i
        // y para la tabla c utilizaremos como índice j.

        j=0;
        for (i=0;i<10;i++){
            c[j]=a[i];
            j++;
            c[j]=b[i];
            j++;
        }

        System.out.println("La tabla c queda: ");

        for (j=0;j<20;j++) // seguimos utilizando j, para la tabla c. Aunque se podría utilizar i.
            System.out.print(c[j]+"  ");

        System.out.println("");
    }
}
```

6. Leer los datos correspondiente a dos tablas de 12 elementos numéricos, y mezclarlos en una tercera de la forma: 3 de la tabla A, 3 de la B, otros 3 de A, otros 3 de la B, etc.

```
package bol04ej06;
```

```
public class Main {

    public static void main(String[] args) {
        int a[], b[], c[];
        int i,j;


        a=new int[12];
        b=new int[12];

        // la tabla c tendrá que tener el doble de tamaño que a y b.
        c = new int [24];


        // leemos la tabla a
        System.out.println("Leyendo la tabla a");

        for (i=0;i<12;i++){
            System.out.print("número: ");
            a[i]=Entrada.entero();
        }

        // leemos la tabla b
        System.out.println("Leyendo la tabla b");

        for (i=0;i<12;i++){
            System.out.print("número: ");
            b[i]=Entrada.entero();
        }

        // asignaremos los elementos de la tabla c
        // para las tablas a y b utilizaremos como índice i
        // y para la tabla c utilizaremos como índice j.

        j=0;
        i=0;
```

```
        while (i<12)
        {
            // copiamos 3 de a
            for (int k=0;k<3; k++)
            {
                c[j]=a[i+k];
                j++;
            }

            // copiamos 3 de b
            for (int k=0;k<3;k++)
            {
                c[j]=b[i+k];
                j++;
            }

            // como hemos copiado 3 de a y b, incrementamos la i en 3.
            i+=3;

            // la j se incrementa cada vez que se añade un elemento a la tabla c.
        }

        System.out.println("La tabla c queda: ");

        for (j=0;j<24;j++) // seguimos utilizando j, para la tabla c. Aunque se podría utilizar i.
            System.out.print(c[j]+"  ");

        System.out.println("");
    }
}
```

7. Leer por teclado una serie de 10 números enteros. La aplicación debe indicarnos si los números están ordenados de forma creciente, decreciente, o si están desordenados.

```
package bol04ej07;

public class Main {

    public static void main(String[] args) {
        int numeros[];
        int i;

        boolean creciente, decreciente;
            // creciente indicará si los números están ordenados de forma creciente
            // decreciente indicará si la serie está ordenada de forma decreciente

            // los posible valores para creciente y decreciente son:

            /*        creciente         decreciente
             *          false              false    -> cuando todos los números sean idénticos
             *          false              true     -> orden decreciente
             *          true               false    -> orden creciente
             *          true               true     -> desordenado
             *
             *  si, para algún i, se cumple t[i]>t[i+1]: la serie t[i], t[i+1] es decreciente
             *
             * o el caso contrario.
             *
             */

        numeros = new int [10];

        creciente = false;
        decreciente = false;
```

```
        // leemos los números
        System.out.println("Leyendo números:");

        for (i=0;i<10;i++){
            System.out.print("número: ");
            numeros[i]=Entrada.entero();
        }


        // comprobaremos el orden
        for (i=0;i<9;i++)   // usamos i e i+1, por lo que la i solo podrá llegar hasta 8 (ó <9)
        {
            if (numeros[i] > numeros[i+1])  // en este momento es decreciente
                decreciente = true;

            if (numeros[i] < numeros[i+1])  // en este momento es creciente
                creciente = true;
        }

        // dependiendo de los valores de creciente y decreciente daremos un tipo de ordenación


        if (creciente ==true && decreciente ==false) //toda las parejas están en orden creciente
            System.out.println("Serie creciente.");

        if (creciente ==false && decreciente ==true) // todas la parejas están en orden decreciente
            System.out.println("Serie decreciente.");

        if (creciente ==true && decreciente ==true) // si ha tenido momentos creciente y decrecientes
            System.out.println("Serie desordenada.");

        if (creciente ==false && decreciente ==false) // no hay parejas crecientes ni decrecientes
            System.out.println("Todos los números iguales."); // lo que significa que todos son iguales

    }
}
```

8. Diseñar una aplicación que declare una tabla de 10 elementos enteros. Leer mediante el teclado 8 números. Después se debe pedir un número y una posición, insertarlo en la posición indicada, desplazando los que estén detrás.

```
package bol04ej08;

public class Main {

    public static void main(String[] args) {
        int t[]=new int[10];
        int elemento, posicion;

        // leemos 8 números
        System.out.println("Leyendo datos...");
        for (int i=0;i<8;i++){
            System.out.print("Introduzca número: ");
            t[i]=Entrada.entero();
        }


        // pedimos el nuevo elemento y la posición
        System.out.print("Nuevo elemento: ");
        elemento = Entrada.entero();

        System.out.print("Posición donde insertar (de 0 a 8): ");
        posicion = Entrada.entero();

        // supondremos que la posición estará entre 0 y 8.
        // un valor distinto podría dar un error en tiempo de ejecución
        // una posible mejora, propuesta para el lector, es comprobar esto.


        // ahora desplazaremos los elementos de la tabla
        // desde posición hasta el último (en este caso 7)
```

```
        for (int i=7;i>=posicion;i--)
            t[i+1]=t[i];

        //insertamos el nuevo elemento
        t[posicion] =elemento;

        System.out.println("La tabla queda:");
        for (int i=0;i<9;i++)
            System.out.println(t[i]);
    }
}
```

9. Crear un programa que lea por teclado una tabla de 10 números enteros y la desplace una posición hacia abajo: el primero pasa a ser el segundo, el segundo pasa a ser el tercero y así sucesivamente. El último pasa a ser el primero.

```
package bol04ej09;

public class Main {

    public static void main(String[] args) {
        int t[]=new int[10];
        int ultimo;

        // leemos la tabla
        for (int i=0;i<10;i++){
            System.out.print("Introduzca número: ");
            t[i]=Entrada.entero();
        }

        // guardamos el último elemento de la tabla
        ultimo = t[9];
```

```
        // desplazamos hacia abajo (de 0 hacia la última posición)
        // al desplazar perdemos el último valor, por eso lo hemos guardado antes.
        for (int i=8;i>=0;i--)
            t[i+1]=t[i];

        // el último valor pasa a ser el primero
        t[0] =ultimo;

        System.out.println("La tabla queda:");
        for (int i=0;i<10;i++)
            System.out.println(t[i]);
    }
}
```

10. Ídem, desplazar N posiciones (N es introducido por el usuario).

```
package bol04ej10;

public class Main {

    public static void main(String[] args) {
        int t[]=new int[10];
        int ultimo;
        int n;

        // leemos la tabla
        for (int i=0;i<10;i++){
            System.out.print("Introduzca número: ");
            t[i]=Entrada.entero();
        }

        // preguntamos cuantas posiciones se desea desplazar
        System.out.print("Posiciones a desplazar:" );
```

```
        n = Entrada.entero();



        // del ejercicio anterior tenemos una versión que desplaza una sola
        // posición. Si repetimos este proceso n veces, conseguiremos
        // desplazar n veces.
        // este algoritmo es muy fácil de implementar, pero es muy costoso en tiempo.

        for (int vueltas=1;vueltas <=n; vueltas++)
        {
            // guardamos el último elemento de la tabla
            ultimo = t[9];

            // desplazamos hacia abajo (de 0 hacia la última posición)
            // al desplazar perdemos el último valor, por eso lo tenemos guardado.
            for (int i=8;i>=0;i--)
                t[i+1]=t[i];

            // el último valor pasa a ser el primero
            t[0] =ultimo;
        }


        System.out.println("La tabla queda:");
        for (int i=0;i<10;i++)
            System.out.println(t[i]);
    }
}
```

```
// un algoritmo más eficiente
package bol04ej10;

public class Main {
```

```
    public static void main(String[] args) {
        int t[]=new int[10];
        int n;

        // leemos la tabla
        for (int i=0;i<10;i++){
            System.out.print("Introduzca número: ");
            t[i]=Entrada.entero();
        }

        // preguntamos cuantas posiciones desea desplazar
        System.out.print("Posiciones a desplazar:" );
        n = Entrada.entero();


        // en el caso de desplazar una posición: necesitamos guardar solo un elemento (el último que
        // pasa a ser el primero)
        // en el caso de desplazar dos posiciones: necesitamos guardar dos elementos (penúltimo y
        // último, que pasarán a ser primero y segundo)
        // como a priori no sabemos cuantos elementos vamos a desplazar, ni cuantos elementos tenemos
        // que guardar. Una buena solución es hacer una copia de la tabla completa

        int copia[] = t.clone();


        // desplazamos hacia abajo n posiciones
        for (int i=0;i<10;i++)
            t[i] =copia[(i+n)%10];
            // la tabla funciona como si fuese circular por eso utilizamos el módulo

        System.out.println("\n\nLa tabla queda:");
        for (int i=0;i<10;i++)
            System.out.println(t[i]);
    }
}
```

11. Leer 5 elementos numéricos que se introducirán ordenados de forma creciente. Éstos los guardaremos en una tabla de tamaño 10. Leer un número N, e insertarlo en el lugar adecuado para que la tabla continúe ordenada.

```
package bol04ej11;

public class Main {

    public static void main(String[] args) {
        int t[]=new int[10];
        int num,sitio_num,j;

        for (int i=0;i<5;i++)
        {
            System.out.print("Introduzca número (ordenado crecientemente): ");
            t[i]=Entrada.entero();
        }

        System.out.println();
        System.out.print("Número a insertar entre los anteriores: ");
        num=Entrada.entero();



        sitio_num=0;
        j=0;

        // buscaremos el sitio donde debería ir num
        while(t[j]<num && j<=4){
            sitio_num ++;
            j++;
        }


        // desplazaremos los elementos desde el sitio_num hasta el final
        // así haremos un hueco para num
```

```
        for (int i=4; i>=sitio_num; i--)
            t[i+1]=t[i];

        // por último ponemos num en su sitio para que todo siga ordenado
        t[sitio_num]=num;

        System.out.println("La nueva serie ordenada queda: ");
        for (int i=0;i<5+1;i++)
            System.out.println(t[i]);


    }
}
```

12. Leer por teclado una tabla de 10 elementos numéricos enteros y una posición (entre 0 y 9). Eliminar el elemento situado en la posición dada sin dejar huecos.

```
package bol04ej12;

public class Main {

    public static void main(String[] args) {
        int t[]=new int[10];
        int posicion;


        // leemos los 10 números
        for (int i=0;i<10;i++)
        {
            System.out.print("Elemento ("+i+"): ");
            t[i]=Entrada.entero();
        }
```

```
        System.out.println();

        // leemos la posición que nos interesa
        // suponemos que la posición está en el rango 0..9
        System.out.print("Posición a eliminar: ");
        posicion=Entrada.entero();



        // desplazamos desde posición hasta el final todos los elementos un lugar hacia la izquierda
        // con lo que el elemento que está en posición se pierde (se borra)

        for (int i=posicion;i<9;i++) // la i llega hasta la penúltima posición,
            t[i] = t[i+1];           // ya que dentro usamos (i+1) que es la última posición
                                     // así evitamos salirnos de la tabla

        System.out.println("La tabla queda: ");
        for (int i=0;i<9;i++)       // hay que tener cuidado que ahora hay un
            System.out.println(t[i]);  // elemento útil menos en la tabla

    }
}
```

13. Leer 10 enteros. Guardar en otra tabla los elementos pares de la primera, y a continuación los elementos impares.

Realizar dos versiones: una trabajando con los valores y otra trabajando con los índices.

```
package bol04ej13;

public class Main {
     /**
     * En esta versión utilizaremos para ver los elementos pares o impares
     * los valores de la tabla, es decir t[0], t[1],...
```

```
 */

public static void main(String[] args) {
    int t[];
    int cont_par, par[]; // contador de números pares y tabla para guardarlos
    int cont_impar,impar[]; // ídem para los impares

    t = new int[10];

    // Leemos los valores de la tabla
    for (int i=0;i<10;i++){
        System.out.print("Introduzca un número: ");
        t[i]=Entrada.entero();
    }


    // contamos la cantidad de elementos pares e impares
    // también se podía contar solo lo pares y calcular los impares=10 -pares
    cont_par=0;
    cont_impar=0;

    for (int i=0;i<10;i++)
        if(t[i]%2==0)
            cont_par++;
        else
            cont_impar++;

    // creamos las tablas par e impar del tamaño adecuado
    par=new int[cont_par];
    impar=new int[cont_impar];

    // volvemos a procesar para copiar cada elemento en la tabla adecuada
    cont_par=0;
    cont_impar=0;

    for (int i=0;i<10;i++)
```

```
            if(t[i]%2==0)
            {
                par[cont_par]=t[i];
                cont_par++;
            }
            else
            {
                impar[cont_impar]=t[i];
                cont_impar++;
            }

        System.out.println("\n\nTabla par:");
        for (int i=0;i<cont_par;i++)
            System.out.println(par[i]);

        System.out.println("Tabla impar:");
        for (int i=0;i<cont_impar;i++)
            System.out.println(impar[i]);
    }
}
```

```
package bol04ej13;

public class Main {

    /*
     * En esta versión nos fijaremos en los índices pares e impares
     * en par se incluirá t[2],t[4]...
     * en impar t[1], t[2],...
     */

    public static void main(String[] args) {
        int t[];
        int par[];
```

```
        int impar[];

        t = new int[10];

        // Leemos los valores de la tabla
        for (int i=0;i<10;i++){
            System.out.print("Introduzca un número: ");
            t[i]=Entrada.entero();
        }


        // creamos las tablas par e impar del tamaño adecuado (5)
        par=new int[5];
        impar=new int[5];

        // copiamos cada elemento en la tabla adecuada

        for (int i=0;i<10;i+=2)
            par[i/2] = t[i];

        for (int i=1;i<10;i+=2) // la división entera redondeada hacia el entero más próximo por
            impar[i/2] = t[i];  // debajo: 1/2 =0, 3/2 =1, etc.

        System.out.println("\n\nTabla par:");
        for (int i=0;i<5;i++)
            System.out.println(par[i]);

        System.out.println("Tabla impar:");
        for (int i=0;i<5;i++)
            System.out.println(impar[i]);
    }
}
```

14. Leer dos series de 10 enteros, que estarán ordenados crecientemente. Copiar (fusionar) las dos tablas en una tercera, de forma que sigan ordenados.

```
package bol04ej14;

public class Main {

    public static void main(String[] args) {
        int a[], b[], c[];
        int i,j,k;


        a =new int[10];
        b =new int[10];

        // leemos a
        System.out.println("Datos para a:");
        for (i=0;i<10;i++){
            System.out.print("Introduzca numero (orden creciente): ");
            a[i]=Entrada.entero();
        }


        // leemos b
        System.out.println("\nDatos para b:");
        for (i=0;i<10;i++){
            System.out.print("Introduzca numero (orden creciente): ");
            b[i]=Entrada.entero();
        }


        System.out.println();

        // creamos c
```

```
        c = new int [20];

        // comenzamos a fusionar a y b en c
        i=0;   // utilizaremos i como índice de a;
        j=0;   // utilizaremos j como índice de b;
        k=0;   // utilizaremos k como índice de c

        while(i<10 && j<10)
        {
            if (a[i] < b[j])    // nos interesa el elemento de a
            {
                c[k] = a[i];
                i++; // incrementamos i para tomar el siguiente elemento de a
            }
            else
            {
                c[k] = b[j];
                j++;  // incrementamos j para tomar el siguiente elemento de b
            }

            k++;  // como hemos copiado a c[k], incrementamos k, para
                  // en la siguiente vuelta, utilizar el siguiente hueco de la tabla
        }

        // cuando salimos de while es por que alguna de las tablas (a o b) ha llegado al final


        if(i==10)   // hemos copiado toda la tabla a en c, queda por copiar un resto de b
            while (j<10)
            {
                c[k] = b[j];
                j++;
                k++;
            }
        else    // hay que copiar el resto de a en c
            while (i<10)
```

```
            {
                c[k] = a[i];
                i++;
                k++;
            }

        System.out.println("Mostramos la tabla c:");

        for (k=0;k<20;k++)
            System.out.print(c[k] + "   ");

        System.out.println("");
    }
}
```

15. Leer 10 enteros ordenados crecientemente. Leer N y buscarlo en la tabla. Se debe mostrar la posición en que se encuentra. Si no está, indicarlo con un mensaje.

```
package bol04ej15;

public class Main {

    public static void main(String[] args) {
        int t[]=new int[10];
        int num,j;

        for (int i=0;i<10;i++){
            System.out.print("Introduzca numero (orden creciente): ");
            t[i]=Entrada.entero();
        }


        System.out.println();
```

```
        System.out.println("Introduzca numero a buscar: ");
        num=Entrada.entero();
        j=0;
        while(j<10 && t[j]<num){
            j++;
        }

        // cuando me salgo del mientras puede ser por dos motivos:
        // - que j llegue a 10 ó
        // - que encuentre el donde iría num en la tabla

        if(j==10) // hemos llegado al final y no hemos encontrado nada.
            System.out.println("No encontrado");
        else{
            // si t[j] < num, hemos sobrepasado el lugar donde debería estar num, sin encontrarlo
            if(t[j]==num) // num está en la posición i. Entonces si está ahí
                System.out.println("Encontrado en la posición " + j);
            else
                System.out.println("No encontrado");
        }
    }
}
```

16. Queremos desarrollar una aplicación que nos ayude a gestionar las notas de un centro educativo. Cada grupo (o clase) está compuesto por 5 alumnos. Se pide leer las notas del primer, segundo y tercer trimestre de un grupo. Debemos mostrar al final: la nota media del grupo en cada trimestre, y la media del alumno que se encuentra en la posición N (N se lee por teclado).

```
package bol04ej16;

public class Main {
```

```
    public static void main(String[] args) {
        int primer[], segundo[], tercer[]; // notas del primer, segundo y tercer trimestre
        int num,i;

        int suma_primer, suma_segundo, suma_tercer;
        double media_alumno;



        // creamos las tablas necesarias
        primer = new int [5];
        segundo = new int [5];
        tercer = new int [5];

        // el programa consta de dos partes: entrada de datos y procesado. Se podrían procesar las
        // notas mientras se leen. Preferimos utilizar dos bloques por tener un código más legible.

        // leemos las notas del primer trimestre
        System.out.println("Notas de primer trimestre:");
        for (i=0;i<5;i++){
            System.out.print("Alumnos ("+i+"): ");
            primer[i]=Entrada.entero();
        }

        // leemos las notas del segundo trimestre
        System.out.println("Notas del segundo trimestre:");
        for (i=0;i<5;i++){
            System.out.print("Alumno ("+i+"): ");
            segundo[i]=Entrada.entero();
        }

        // leemos las notas del tercer trimestre
        System.out.println("Notas del tercer trimestre:");
        for (i=0;i<5;i++){
            System.out.print("Alumnos ("+i+"): ");
            tercer[i]=Entrada.entero();
```

```
        }


        // calculamos las medias
        suma_primer = 0;    // ponemos a 0 los acumuladores
        suma_segundo = 0;
        suma_tercer = 0;


        for (i =0; i < 5; i++)
        {
            suma_primer += primer[i];
            suma_segundo += segundo[i];
            suma_tercer += tercer[i];
        }

        // mostramos datos
        System.out.println("Media primer trimestre: "+ suma_primer/5.0);
        System.out.println("Media segundo trimestre: "+ suma_segundo/5.0);
        System.out.println("Media tercer trimestre: "+ suma_tercer/5.0);
        System.out.println();


        // leemos la posición del alumnos que nos interesa
        // una posible mejora es comprobar que el índice se encuentre
        // entre 0 y 4
        System.out.print ("Introduzca posición del alumno (de 0 a 9): ");
        num=Entrada.entero();

        // la media del alumno es la suma de sus notas entre 3
        media_alumno = (double) (primer[num]+segundo[num]+tercer[num])/3;

        System.out.println("La media del alumno es: " + media_alumno);
    }
}
```

Boletín 5
Tablas n-dimensionales

1. Crear una tabla bidimensional de tamaño 5x5 y rellenarla de la siguiente forma: la posición T[n,m] debe contener n+m. Después se debe mostrar su contenido.

```
package bol05ej01;

public class Main {

    public static void main(String[] args) {
        int t[][]; // definimos t como una tabla bidimensional

        t = new int [5][5]; // creamos la tabla de 5x5



        for (int i=0;i<5;i++) // utilizamos i para la primera dimensión
        {
            for (int j=0;j<5;j++)  // utilizamos j para la segunda dimensión
            {
                t[i][j]=i+j;
            }
        }

        System.out.println("TABLA: ");
```

```
        for (int i=4;i>=0;i--)
        {
            System.out.println();
            for (int j=0;j<5;j++)
            {
                System.out.print(t[i][j]+" ");

            }
        }
    }
}
```

2. Crear y cargar una tabla de tamaño 4x4 y decir si es simétrica o no, es decir si se obtiene la misma tabla al cambiar las filas por columnas.

```
package bol05ej02;

public class Main {

    public static void main(String[] args) {
        int t[][];
        boolean simetrica;
        int i,j;


        t = new int[4][4];

        for (i=0;i<4;i++)
        {
            for (j=0;j<4;j++)
            {
                System.out.print("Introduzca elemento ["+i+"]["+j+"]: ");
                t[i][j]=Entrada.entero();
            }
```

```
        }

        simetrica=true;  // suponemos que la matriz es simétrica, y en caso de
                         // encontrar un caso donde t[i][j] sea distinta de t[j][i] pondremos
                         // simétrica a falso.

        //una solución es mirar todos los elementos de la matriz, pero se hacen comprobaciones
        // dobles, un ejemplo: comprobamos t[1][2] con t[2][1]... pero más tarde comprobaremos
        // t[2][1] con t[1][2]

        // la solución será mirar solo la zona inferior o superior a la diagonal principal.
        // En el momento que tengamos la constancia de que no es simétrica, pararemos
        // todas las comprobaciones

        i=0;
        while(i<4 && simetrica==true){
            j=0;
            while(j<i && simetrica==true){
                if(t[i][j]!=t[j][i])
                      simetrica=false;

                j++;
            }
            i++;
        }

        // si en algún momento se da: t[i][j]!=t[j][i] es que la matriz no es simétrica.
        // si al llegar aquí y la variable simétrica vale true, indica que no hemos encontrado
        // ningún valor que indique que la matriz no es simétrica.

        if(simetrica)
            System.out.println("SIMETRICA");
        else
            System.out.println("NO ES SIMETRICA");
    }
}
```

3. Crear y cargar dos matrices de tamaño 3x3, sumarlas y mostrar su suma.

```
package bol05ej03;

public class Main {

    public static void main(String[] args) {
        int a[][], b[][], suma[][];
        int i,j;

        a = new int[3][3];
        b = new int[3][3];

        // Leemos los datos
        System.out.println ("Matriz A:");

        for (i=0;i<3;i++)
        {
            for (j=0;j<3;j++)
            {
                System.out.print("A["+i+"]["+j+"]: ");
                a[i][j]=Entrada.entero();
            }
        }

        System.out.println ("Matriz B:");

        for (i=0;i<3;i++)
        {
            for (j=0;j<3;j++)
            {
                System.out.print("B["+i+"]["+j+"]: ");
                b[i][j]=Entrada.entero();
            }
        }
```

```
        // hacemos la suma
        suma = new int[3][3];

        for (i=0;i<3;i++)
        {
            for (j=0;j<3;j++)
            {
                suma[i][j] = a[i][j] + b[i][j];
            }
        }


        // mostramos los resultado
        System.out.println ("Matriz Suma:");

        for (i=0;i<3;i++)
        {
            for (j=0;j<3;j++)
            {
                System.out.print (suma[i][j] + " ");
            }
            System.out.println ();
        }
    }
}
```

4. Crear y cargar una tabla de tamaño 3x3, trasponerla y mostrarla.

```
package bol05ej04;

public class Main {

    public static void main(String[] args) {
        int t[][]=new int[3][3];
```

```
        int aux;

        for (int i=0;i<3;i++){
            for (int j=0;j<3;j++){
                System.out.print("Introduzca elemento["+i+"]["+j+"]: ");
                t[i][j]=Entrada.entero();
            }
        }

        // mostramos la matriz original

        System.out.print ("Matriz original:");
        for (int i=0;i<3;i++){
            System.out.println();
            for (int j=0;j<3;j++){
                System.out.print(t[i][j]+" ");

            }
        }

        // trasponemos la matriz
        // no podemos transponer todos los elementos. Un ejemplo el elemento t[1][2]
        // se convierte en el [2][1]... pero cuando transpongamos el elemento [2][1] se convierte
        // en el [1][2]. Al intercambiar dos veces los elemento, la matriz se quedaría exactamente igual.


        // solo traspondremos los elementos por debajo de la diagonal principal.

        for (int i=1;i<3;i++){
            for (int j=0;j<i;j++){
                aux=t[i][j];
                t[i][j]=t[j][i];
                t[j][i]=aux;
            }
        }
```

```
        // mostramos la matriz transpuesta

        System.out.println();
        System.out.println("----------------------");
        System.out.println ("Matriz transpuesta");

        for (int i=2;i>=0;i--){
            System.out.println();
            for (int j=0;j<3;j++){
                System.out.print(t[i][j]+" ");
            }
        }
    }
}
```

5. Crear una tabla de tamaño 7x7 y rellenarla de forma que los elementos de la diagonal principal sean 1 y el resto 0.

```
package bol05ej05;

public class Main {

    public static void main(String[] args) {
        int t[][]=new int[7][7];
        int i,j;


        for (i=0;i<7;i++)
            for (j=0;j<7;j++)
                if (i==j)
                    t[i][j] = 1;
                else
                    t[i][j] = 0; // en java, al crear una tabla de enteros, todos los elementos se
```

```
                                // inicializan a 0. Por lo que esta instrucción no es necesaria. Se aña_
                                // de para que el código sea más comprensible.

        // mostramos la matriz
        // la forma de ver la matriz no es la típica que estamos acostumbrados en
        // matemática... pero desde el punto de vista del algoritmo no es relevante.

        System.out.println ("Matriz:");
        for (i=0;i<7;i++)
        {
            for (j=0;j<7;j++)
                System.out.print(t[i][j]+" ");
            System.out.println ();
        }
    }
}
```

6. Crear y cargar una tabla de tamaño 10x10, mostrar la suma de cada fila y de cada columna.

```
package bol05ej06;

public class Main {

    public static void main(String[] args) {
        int t[][]=new int[4][4];
        int suma_fila,suma_col;
        int i,j;
        final int tamaño=4;

        t = new int[tamaño][tamaño];

        for (i=0;i<tamaño;i++){
            for (j=0;j<tamaño;j++){
                System.out.print("Elemento ["+i+"]["+j+"]: ");
```

```
                t[i][j]=Entrada.entero();
            }
        }

        // sumamos columna a columna
        System.out.println();
        for (i=0;i<tamaño;i++){
            suma_col=0;
            for (j=0;j<tamaño;j++){
                suma_col=suma_col+t[i][j];
            }
            System.out.println("Columna"+" "+i+":"+" "+suma_col);
        }

        // sumamos fila a fila
        for (j=0;j<tamaño;j++){
            suma_fila=0;
            for (i=0;i<tamaño;i++){
                suma_fila=suma_fila+t[i][j];
            }
            System.out.println("Fila"+" "+j+":"+" "+suma_fila);
        }
    }
}
```

7. utilizando dos tablas de tamaño 5x9 y 9x5, cargar la primera y trasponerla en la segunda.

```
package bol05ej07;

public class Main {

    public static void main(String[] args) {
        int a[][], b[][];
```

```
        int i,j;

        a = new int[5][9];
        b = new int [9][5];


        for (i=0;i<5;i++){
            for (j=0;j<9;j++){
                System.out.print("Elemento ["+i+"]["+j+"]: ");
                a[i][j]=Entrada.entero();
                // si queremos ahorrarnos introducir 5x9 (45) números, podemos comentar las
                // dos lineas anteriores y utilizar (por ejemplo):
                // a[i][j] = 10*i+j;
            }
        }

        // trasponemos
        for (i=0;i<5;i++){
            for (j=0;j<9;j++){
                b[j][i] = a[i][j];
            }
        }

        // mostramos la matriz traspuesta
        System.out.println("Matriz traspuesta");

        for (i=0;i<9;i++){
            for (j=0;j<5;j++)
                System.out.print (b[i][j] + " ");
            System.out.println ();

        }
    }
}
```

8. Crear una matriz "marco" de tamaño 8x6: todos sus elementos deben ser 0 salvo los de los bordes que deben ser 1. Mostrarla.

```
package bol05ej08;

public class Main {

    public static void main(String[] args) {
        int i,j;
        int t[][];

        t = new int[8][6]; // se inicializa toda la tabla a 0.

        // rellenamos la matriz marco
        for (i=0;i<8;i++)
            for (j=0;j<6;j++) {
                if(i==0 || i==7) // si nos encontramos en la primera o última columna
                    t[i][j]=1;
                if(j==0 || j==5) // si nos encontramos en la primera o última fila
                    t[i][j]=1;
            }

        System.out.print("Matriz marco: ");

        for (i=0;i<8;i++){
            System.out.println();
            for (j=0;j<6;j++){
                System.out.print (t[i][j]+" ");
            }
        }
        System.out.println ();
    }
}
```

9. Hacer lo mismo que el ejercicio anterior, pero con una matriz 9x9x9. Creamos un cubo con las caras puestas a 1 y el interior a 0.

```
package bol05ej09;

public class Main {

    public static void main(String[] args) {
        int t[][][];
        int i,j,k;

        t = new int[9][9][9];

        for (i=0;i<9;i++){
            for (j=0;j<9;j++){
                for (k=0;k<9;k++){
                    if(i==0 || i==8 || j==0 || j==8 || k==0 || k==8)
                        // si estamos en la primera o última columna, fila o capa de la matriz
                        t[i][j][k]=1;
                }
            }
        }
        // Mostramos la matriz capa a capa
        System.out.println("Matriz: ");

        for (i=0;i<9;i++)
        {
            System.out.println("Capa: " +i);
            for (j=0;j<9;j++)
            {
                for (k=0;k<9;k++)
                {
                    System.out.print(t[i][j][k] + " ");
                }
                System.out.println ();
            }
            System.out.println (" -------------- ");
        }
```

```
    }
}
```

10. Los siguientes programas piden una serie de datos y tras procesarlos ofrecen unos resultados por pantalla. Mostrar el resultado:

```
PROGRAMA Ej10a
VARIABLES
    i, m, a: ENTEROS
    t: TABLA [5] ENTEROS
COMIENZO
    PARA i←0 HASTA 4
        leer (t[i])
    FIN PARA
    m ← 0
    PARA i ← 0 HASTA 4
        SI t[i] > m
            m ← t[i]
         FIN SI
    FIN PARA
    a ← t[4-m]
    t[4-m] ← t[m]
    t[m] ← a
    PARA i ← 0 HASTA 4
        escribir (t[i])
    FIN PARA
FIN PROGRAMA
```

Datos de entrada: -4, 0, 1, 3 y 2.

```
PROGRAMA Ej10b
VARIABLES
    n, i: ENTEROS
    a, b: TABLA [100] ENTEROS
COMIENZO
    n ← 10
    PARA i←0 HASTA n-1
        leer (a[i])
    FIN PARA
    PARA i ← 0 HASTA n/2
        b[i] ← a[n-1-i]
        b[n-1-i] ← a[i]
    FIN PARA
    PARA i ← 0 HASTA n-1
        SI i mod 2 = 0
            escribir (a[i])
        SINO
            escribir (b[i])
        FIN SI
    FIN PARA
FIN PROGRAMA
```

Datos de entrada:
6, 2, 8, 9, 2, 5, 8, 2, 6 y 1.

```
package bol05ej10;

public class Main {

    public static void main(String[] args) {
        int i, m, a;

        // la idea de este ejercicio es hacer una traza de forma manual, y tras
        // ésta, escribir el código equivalente para comprobar el resultado.
        // Debemos destacar que este algoritmo no tiene sentido ni hace nada en concreto
        // incluso con otros datos de entrada el algoritmo puede dar un error, al
        // utilizar datos como índices de tablas sin las pertinentes
        // comprobaciones.

        int t = new int [5];

        for (i=0; i<=4; i++)
            t[i] = Entrada.entero();

        // podemos sustituir la lectura de datos por la siguiente línea:
        //int t[] = {-4, 0, 1, 3, 2};

        m = 0;
        for (i=0; i<=4; i++)
            if (t[i] > m )
                m = t[i];

        a = t[4-m];
        t[4-m] = t[m];
        t[m] = a;

        for (i=0; i<=4; i++)
            System.out.println (t[i]);
    }
}
```

```
package bol05ej10;

public class Main {

    public static void main(String[] args) {
        int n, i;

        /* la idea de este ejercicio es hacer una traza de forma manual, y tras ésta, escribir el código
           equivalente para comprobar el resultado. Debemos destacar que este algoritmo no tiene sentido
           ni hace nada en concreto incluso con otros datos de entrada el algoritmo puede dar un error,
           al utilizar datos como índices de tablas sin las pertinentes comprobaciones.
        */

        int a[] = new int [10];
        int b[] = new int [10];

        for (i=0; i<=n; i++)
            a[i] = Entrada.entero();

        // podemos sustituir la declaración de a y la lectura de datos por la siguiente línea:
        // int a[] = {6, 2, 8, 9, 2, 5, 8, 2, 6, 1};

        n = 10;

        for (i=0; i<=n/2; i++) {
            b[i] = a[n-1-i];
            b[n-1-i] = a[i];
        }

        for (i=0; i<n; i++)
            if (i % 2 == 0)
                System.out.println (a[i]);
            else
                System.out.println (b[i]);
    }
}
```

11-Se pretende realizar un programa para gestionar la lista de participaciones en una competición de salto de longitud. El número de plazas disponible es de 10. Sus datos se irán introduciendo en el mismo orden que vayan inscribiéndose los atletas. Diseñar el programa que muestre las siguientes opciones:

1- Inscribir un participante.
2- Mostrar listado de datos.
3- Mostrar listado por marcas.
4- Finalizar el programa.

Si se selecciona 1, se introducirán los datos de uno de los participantes: Nombre, mejor marca del 2002, mejor marca del 2001 y mejor marca del 2000.
Si se elige la opción 2, se debe mostrar un listado por número de dorsal.
La opción 3 mostrará un listado ordenado por la marca del 2002, de mayor a menor.
Tras procesar cada opción, se debe mostrar de nuevo el menú inicial, hasta que se seleccione la opción 4, que terminará el programa.

```
package bol05ej11;

public class Main {

    public static void main(String[] args) {
        final int TAM=10,D=0,M0=1,M1=2,M2=3;
        // TAM: Número máximo de participantes
        // D: número de dorsal
        // M0, M1, M2: Marca del 2000, 2001, y 2002

        int opc,numc,dorsal,i,aux;
        boolean d_rep,inter;

        int part[][]=new int[TAM][4];
        numc=0;
        opc=0;
```

```
do{
   System.out.println();
   System.out.println("---------------------------------");
   System.out.println("1. Inscribir participantes");
   System.out.println("2. Mostrar listado por datos");
   System.out.println("3. Mostrar listado por marcas");
   System.out.println("4. Salir");
   System.out.println("---------------------------------");
   System.out.print("Por favor, introduzca una opción: ");
   opc=Entrada.entero();
   System.out.println();
   switch(opc){
       case 1:
           if(numc==20)
               System.out.println("Listado completo");
           else{
               do{
                   System.out.print("Introduzca dorsal: ");
                   dorsal=Entrada.entero();

                   d_rep=false;
                   i=0;
                   while(i<numc && d_rep==false){
                       if(part[i][D]==dorsal){
                           System.out.print("Dorsal registrado.");
                           System.out.println("Por favor intente de nuevo");
                           d_rep=true;
                       }
                       i++;
                   }
               }while(d_rep==true);
               if(d_rep==false){
                   part[numc][D]=dorsal;
                   System.out.print("Introduzca marca del 2000: ");
                   part[numc][M0]=Entrada.entero();
```

```
                System.out.print("Introduzca marca del 2001: ");
                part[numc][M1]=Entrada.entero();
                System.out.print("Introduzca marca del 2002: ");
                part[numc][M2]=Entrada.entero();
                System.out.println();
                numc++;
            }
        }
       break;

    case 2: // método de ordenación por burbuja, controlado por intercambio
        inter=true;
        while(inter==true){
            inter=false;
            for (int j=0;j<=numc-1-1;j++){
                if(part[j][D]>part[j+1][D]){
                    for (int k=0;k<4;k++){
                        aux=part[j][k];
                        part[j][k]=part[j+1][k];
                        part[j+1][k]=aux;
                    }
                    inter=true;
                }
            }
        }
        System.out.println("LISTADO DE DATOS,SEGUN DORSAL:");
        System.out.println ("dorsal - marcas");
        for (int j=0;j<numc;j++){
            System.out.println();
            for (int k=0;k<4;k++){
                System.out.print(part[j][k]+" ");
            }
        }
       break;

    case 3:
```

```
                    inter=true;
                    while(inter==true){
                        inter=false;
                        for (int j=0;j<=numc-1-1;j++){
                            if(part[j][M2]>part[j+1][M2]){
                                for (int k=0;k<4;k++){
                                    aux=part[j][k];
                                    part[j][k]=part[j+1][k];
                                    part[j+1][k]=aux;
                                }
                                inter=true;
                            }
                        }
                    }
                    System.out.println("LISTADO POR MARCAS :");
                    System.out.println ("dorsal - marcas");
                    for (int j=0;j<numc;j++){
                        System.out.println();
                        for (int k=0;k<4;k++){
                            System.out.print(part[j][k]+" ");
                        }
                    }
                   break;
               }
           }
           while(opc!=4);
       }
}
```

Boletín 6
Funciones

1. Realizar una función, a la que se le pase como parámetro un número N, y muestre por pantalla N veces, el mensaje: "**Módulo ejecutándose**"

```
package bol06ej01;

public class Main {

    public static void main(String[] args) {
         int num;

        System.out.print("Introduzca un numero: ");
        num=Entrada.entero();
        System.out.println("----------------------------");
        mostrar(num);
        System.out.println("----------------------------");
    }


    static void mostrar(int num){
        for (int i=0;i<num;i++){
            System.out.println("Módulo ejecutándose");
        }
    }
}
```

2. Diseñar una función que tenga como parámetros dos números, y que calcule el máximo.

```
package bol06ej02;

public class Main {

     static int maximo(int a, int b){  // suponemos que los tres números serán distintos
        int max;

        if(a>b)
            max=a;
        else
            max=b;

        return(max);
    }


    public static void main(String[] args) {
        int max;
        int a,b;

        System.out.print("Introduzca un numero: ");
        a=Entrada.entero();
        System.out.print("Introduzca otro numero: ");
        b=Entrada.entero();


        max =maximo (a, b);
        System.out.println("El número mayor es: " +max);
        }
}
```

3. Ídem una versión que calcule el máximo de 3 números.

```
package bol06ej03;

public class Main {

     static int maximo(int a, int b, int c){
        int max;

        if(a>b && a>c) // si a es mayor que b y c, entonces a es el máximo
            max=a;
        else
            if(b>a && b>c) // si b es el mayor de todos, entonces b es el máximo
                max=b;
            else
                // si el máximo no es a ni b, será c
                max=c;

        return(max);
    }


    public static void main(String[] args) {
        int max;
        int a, b, c;

        System.out.print("Introduzca un numero: ");
        a=Entrada.entero();

        System.out.print("Introduzca otro numero: ");
        b=Entrada.entero();

        System.out.print("Introduzca el último: ");
        c=Entrada.entero();
```

```
        System.out.println("");

        max =maximo (a, b, c);
        System.out.println("El número mayor es: " +max);
        }
}
```

4. Ídem una versión que calcule el máximo de una tabla de n elementos.

```
package bol06ej04;

public class Main {

   /**
    * Esto funciona solo para tablas con un tamaño mínimo de 1
    *
    */
     static int maximo(int t[]){
        int max;

        max = t[0];

        for (int i = 0; i < t.length; i++)
            if (t[i]>max) // si t[i] es mayor que max, entonces t[i] es el nuevo máximo
                max=t[i];

        return(max);
    }


    public static void main(String[] args) {
        int max;
        int t[];
```

```
        t=new int [6];

        for (int i = 0; i < t.length; i++) // llenamos la tabla con valores aleatorios entre 1 y 100
            t[i]=(int)(Math.random()*100+1);

        System.out.println("Los valores son:");
        for (int i = 0; i < t.length; i++)
            System.out.print(t[i] +"   ");;


        max = maximo (t);
        System.out.println("\nEl número mayor es: " +max);
        }
}
```

5. Función a la que se le pasan dos enteros y muestra todos los números comprendidos entre ellos, inclusive.

```
package bo01ej05;

public class Main {

    static void mostrar(int a,int b){
        int mayor, menor;
        // desconocemos el orden en el que vienen a y b.
        // Lo que haremos es poner los valores correctos en mayor, menor.

        if(a>b){ // a es el mayor. Se podría utilizar la función maximo() implementada anteriormente.
            mayor=a;
            menor=b;
        }
        else{   // en este caso b será el mayor
```

```
            mayor=b;
            menor=a;
        }


        for (int i=menor;i<=mayor;i++)
            System.out.print(i+"  ");


        System.out.println();
    }

    public static void main(String[] args) {
        int a,b;

        System.out.print("Introduzca primer numero: ");
        a=Entrada.entero();
        System.out.print("Introduzca segundo numero: ");
        b=Entrada.entero();

        mostrar(a,b);

    }
}
```

6. Función que muestra en pantalla el doble del valor que se le pasa como parámetro.

```
package bol06ej06;

public class Main {

    static void doble(int num)
    {
```

```
        int doble;

        doble=2*num; // calculamos el doble de num

        System.out.println("El doble es: " +doble);
    }

    public static void main(String[] args) {
        int num;

        System.out.print("Introduzca un número: ");
        num=Entrada.entero();
        doble(num);
    }
}
```

7. Realizar una función que calcule (muestre en pantalla) el área o el volumen de un cilindro, según se especifique. Para distinguir un caso de otro se le pasará el carácter 'a' (para área) o 'v' (para el volumen). Además hemos de pasarle a la función el radio y la altura.

```
package bol06ej07;

public class Main {

     static void area_o_volumen_cilindro(double radio, double altura, char opcion){

         double volumen,area;

         switch (opcion)
         {
                 case 'v':
                     volumen =Math.PI*radio*radio*altura; // radio*radio es el radio al cuadrado
                     System.out.println("El volumen es de: " +volumen);
```

```
                    break;
                case 'a':
                    area = 2*Math.PI*radio*altura + 2*Math.PI*radio*radio;
                    System.out.println("El área es de: "+area);
                    break;
                 default:
                     System.out.println("Indicador del cálculo erróneo");
        }
    }

    public static void main(String[] args) {
        double radio,alt;
        char tipo_calculo;

        System.out.print("Introduzca radio: ");
        radio=Entrada.real();
        System.out.print("Introduzca altura: ");
        alt=Entrada.real();
        System.out.print("Que desea calcular (a/v): ");
        tipo_calculo =Entrada.caracter();

        System.out.println("");

        area_o_volumen_cilindro(radio,alt,tipo_calculo);
    }
}
```

8. Ídem que devuelva una tabla con el área y el volumen.

```
package bol06ej08;

public class Main {
```

```
    static double[] area_y_volumen_cilindro(double radio, double altura)
    {
        double volumen,area;
        double calculo[];

        calculo = new double [2]; // [0] para el volumen y [1] para el área

        calculo[0] =Math.PI*radio*radio*altura; // radio*radio es el radio al cuadrado

        calculo[1] =2*Math.PI*radio*altura + 2*Math.PI*radio*radio;

        return (calculo);
    }

    public static void main(String[] args) {
        double radio,alt;
        double resultado[]; //esta tabla no necesita new, ya que apunta a
                            // la tabla creada dentro de la función

        System.out.print("Introduzca radio: ");
        radio=Entrada.real();
        System.out.print("Introduzca altura: ");
        alt=Entrada.real();

        resultado =area_y_volumen_cilindro(radio,alt); // resultado hace referencia a la tabla devuelta
                                                       // por la función.

        System.out.println("El volumen es de: " +resultado[0]);
        System.out.println("El área es de: " +resultado[1]);

    }
}
```

9. Módulo al que se le pasa un número entero y devuelve el número de divisores primos que tiene.

```
package bol06ej09;

public class Main {

    // la función es_primo indica si el número pasado es o no primo
    // recordamos que un número primo es solo divisible por el mismo y 1
    static boolean es_primo(int num)
    {
        boolean primo;
        int i;

        primo=true; // suponemos que el número es primo


        // este algoritmo se puede mejorar sabiendo que si un número no es
        // divisible entre 2 y su raíz cuadrada, entonces ya no será divisible
        // por ningún otro números -> será primo
        //
        // con esta mejora podemos ahorrar muchas vueltas del while para
        // números grandes

        i=2;
        while(i<num && primo==true) // en realidad bastaría probar hasta la raíz cuadrada de num
        {
            if( num %i == 0) // si es divisible
                primo=false; // si hemos entrado aquí significa que el número no es primo

            i++;
        }

        return(primo);
    }
```

```
    // esta función devuelve el número de divisores primos del número pasado como parámetro.
    //
    // un ejemplo:
    // los divisores de 24 son: 2 y 3
    // aunque 4 y 6 también dividen a 24, no se consideran divisores primos, al no ser primos
    // por lo que 24 tiene tres divisores primos: el 1, el 2 y el 3.

    static int num_divisores (int num){
        int cont;

        cont=1; // siempre habrá un divisor seguro, el 1.

        for (int i=2;i<=num;i++)
            if(es_primo (i) && num %i == 0) // si i es primo y divide a num
                    cont++;     // incrementamos el número de divisores primos


        return(cont);
    }


    public static void main(String[] args) {
        int num,div;
        System.out.print("Introduce numero: ");
        num=Entrada.entero();
        div=num_divisores(num);

        System.out.println("Tiene " +div+ " divisores");
    }
}
```

10.Ídem diseñar una función que devuelve una tabla con los divisores.

```
package bol06ej10;

public class Main {

    // la función es_primo indica si el número pasado es o no primo
    // recordamos que un número primo es solo divisible por el mismo y 1
    static boolean es_primo(int num)
    {
        boolean primo;
        int i;

        primo=true; // suponemos que el número es primo


        // este algoritmo se puede mejorar sabiendo que si un número no es
        // divisible entre 2 y su raíz cuadrada, entonces ya no será divisible
        // por ningún otro números -> será primo
        //
        // con esta mejora podemos ahorrar muchas vueltas del while para
        // números grandes

        i=2;
        while(i<num && primo==true)
        {
            if( num %i == 0) // si es divisible
                primo=false; // si hemos entrado aquí significa que el número no es primo

            i++;
        }

        return(primo);
    }
```

```
// esta función me devuelve el número de divisores del número
// los divisores a tener en cuenta solo son aquellos que son primos
//
// un ejemplo:
// los divisores de 24 son: 2 y 3
// aunque 4 y 6 también dividen a 24, no se consideran divisores, al no ser primos
// por lo que 24 tiene tres divisores (el 1, el 2 y el 3)

static int num_divisores(int num){
    int cont;

    cont=1; // siempre habrá un divisor seguro, el 1.

    for (int i=2;i<=num;i++)
        if(es_primo (i) && num %i == 0) // si i es primo y divide a num
                cont++;     // incrementamos el número de divisores


    return(cont);
}



static int [] divisores(int num){
    int cont=0;
    int div[];  // tabla donde guardaremos los divisores;
    int num_div; // número de divisores primos que tiene num.


    num_div = num_divisores (num);

    div =new int[num_div];

    for (int i=1;i<=num;i++)
        if(es_primo (i) && num %i == 0) // si i es primo y divide a num
```

```
            {
                    div[cont] =i;     // incrementamos el número de divisores
                    cont++;
            }

        return(div);
    }

    public static void main(String[] args) {
        int num, divisores[];
        System.out.print("Introduce numero: ");
        num=Entrada.entero();
        divisores =divisores(num);

        System.out.println("Los divisores de " + num + " son:");
        for (int i = 0; i < divisores.length; i++)
            System.out.print(divisores[i] + "   ");

        System.out.println("");

    }
}
```

11.Escribir una función que calcule el máximo común divisor de dos números.

```
package bol06ej11;

public class Main {

    // el máximo común divisor de dos números es el número más grande que
    // es capaz de dividir a ambos números
    // Para calcularlo podríamos utilizar algún algoritmo existente o hacerlo
    // un poco por la "cuenta de la vieja".
```

```
// La idea es dividir por todos los números desde 1 hasta mínimo(a, b)
// y quedarnos con el mayor.

static int max_comun_divisor (int a, int b)
{
    int mcd=1;
    int min;

    min = minimo (a,b);


    mcd=1; // existe un mcd seguro, el 1, que divide a y b.

    for (int i=2;i<=min;i++)
        if( a%i==0 && b%i==0) // si i divide a "a" y "b"
                mcd=i;        // i será el nuevo mcd.

    return(mcd);
}

  static int minimo(int a, int b){
    int min;

    if(a>b)
        min=b;
    else
        min=a;

    return(min);
}



public static void main(String[] args) {
    int a, b, mcd;
```

```
        System.out.print("Introduce numero: ");
        a=Entrada.entero();

        System.out.print("Introduce otro: ");
        b=Entrada.entero();

        System.out.println("");

        mcd = max_comun_divisor (a, b);

        System.out.println("El mcd de "+a+" y "+b+" es: "+mcd);
    }
}
```

12.Ídem con tres números.

```
package bol06ej12;

public class Main {

    // el máximo común divisor de tres números es el número más grande que
    // es capaz de dividir a todos números
    // Para calcularlo podríamos utilizar algún algoritmo existente o hacerlo
    // un poco por la "cuenta de la vieja".
    // La idea es dividir por todos los números desde 1 hasta mínimo(a, b, c)
    // y quedarnos con el mayor.

    static int max_comun_divisor (int a, int b, int c)
    {
        int mcd=1;
        int min;

        // para no implementar la función mínimo para tres números
```

```
        // utilizaremos la función con solo dos parámetros;
        min = minimo (a,minimo(b,c));


        mcd=1; // existe un mcd seguro, el 1, que divide a y b.

        for (int i=2;i<=min;i++)
            if( a%i==0 && b%i==0 && c%i==0) // si i divide a 'a', a 'b' y a 'c'
                    mcd=i;        // i será el nuevo mcd.

        return(mcd);
    }

      static int minimo(int a, int b){
        int min;

        if(a>b)
            min=b;
        else
            min=a;

        return(min);
    }



    public static void main(String[] args) {
        int a, b,c , mcd;

        System.out.print("Introduce a: ");
        a=Entrada.entero();

        System.out.print("Introduce b: ");
        b=Entrada.entero();

        System.out.print("Introduce c: ");
```

```
        c=Entrada.entero();


        System.out.println("");

        mcd = max_comun_divisor (a, b, c);

        System.out.println("El mcd de ("+a+", "+b+", "+c+") es: "+mcd);

    }
}
```

13.Ídem con una tabla.

```
package bol06ej13;

public class Main {

    /* el máximo común divisor es el número más grande que
     es capaz de dividir a todos los números
     Para calcularlo podríamos utilizar algún algoritmo existente o hacerlo
     un poco por la "cuenta de la vieja".
     La idea es dividir por todos los números desde 1 hasta mínimo(a, b, c)
     y quedarnos con el mayor.
    */

    static int max_comun_divisor (int t[])
    {
        int mcd=1;
        int min;
        boolean divide_a_todos; // una bandera para saber si un número divide
                                // a todos los elementos de la tabla
```

```
        // esto funciona para tabla con al menos un valor
        min = t[0];

        // utilizaremos la función con solo dos parámetros;
        for (int i = 0; i < t.length; i++)
            min = minimo (min,t[i]);

        // al terminar el for, min debe tener el valor mínimo de toda la tabla.


        mcd=1; // existe un mcd seguro, el 1.

        for (int i=2;i<=min;i++)
        {
            divide_a_todos =true;

            for (int k=0;k<t.length;k++)
                if( t[k]%i!=0) // si i divide a t[i]
                    divide_a_todos=false;  // entonces la i no divide a todos los elementos de t.

            if (divide_a_todos == true) // i es capaz de dividir a todos los elementos de t
                mcd =i;     // entonces i es el nuevo mcd
        }

        return(mcd);
    }

      static int minimo(int a, int b){
        int min;

        if(a>b)
            min=b;
        else
            min=a;

        return(min);
```

```
    }
    public static void main(String[] args) {
        int t[], mcd;


        t = new int [4];

        for (int i = 0; i < t.length; i++)
            t[i] = (int)(Math.random()*1000+1); // llenamos t con números aleatorios entre 1 y 1000


        System.out.println("Los números son: ");
        for (int i = 0; i < t.length; i++)
            System.out.print(t[i] +"   ");



        System.out.println("");

        mcd = max_comun_divisor (t);

        System.out.println("El mcd es: " +mcd);

    }
}
```

14.Escribir una función que calcule el mínimo común múltiplo de dos números.

```
package bol06ej14;

public class Main {


```

```
/* el mínimo común múltiplo de a y b, es el número más pequeño que
   es divisible por a y b.
   Para calcularlo podríamos utilizar algún algoritmo existente o hacerlo
   un poco por la "cuenta de la vieja".
   La idea es elegir el mayor de a y b, y multiplicarlo por i, hasta que
   el número resultante sea divisible por el menor de a y b */

static int min_comun_multiplo (int a, int b)
{
    int mcm;
    int max;
    int i;


    max = maximo (a, b);


    mcm =max;   // en principio el mcm es el mayor de los dos números
    i=1;

    while (mcm%a!=0 || mcm%b!=0)   // si el mcm no es divisible por a y b
    {
        i++;
        mcm=max*i;      // el nuevo mcm sera el mayor por i

    }


    return(mcm);
}

  static int maximo (int a, int b){
    int max;

    if(a>b)
        max=a;
```

```
        else
            max=b;

        return(max);
    }



    public static void main(String[] args) {
        int a, b, mcm;



        System.out.print ("Introduce a: ");
        a=Entrada.entero();

        System.out.print ("Introduce b: ");
        b=Entrada.entero();


        System.out.println("");

       mcm = min_comun_multiplo (a,b);

        System.out.println("El mcm es: " +mcm);

    }
}
```

```
package bol06ej14;

public class Main {

    // aprovechando que tenemos hecha la función max_comun_divisor, calcularemos
```

```
    // el mínimo común múltiplo de dos números como la multiplicación de ambos
    // divido por el mcd

    static int min_comun_multiplo (int a, int b)
    {
        int mcm;

        mcm = a*b / max_comun_divisor (a,b);

        return (mcm);
    }


    static int max_comun_divisor (int a, int b)
    {
        int mcd=1;
        int min;

        min = minimo (a,b);


        mcd=1; // existe un mcd seguro, el 1, que divide a y b.

        for (int i=2;i<=min;i++)
            if( a%i==0 && b%i==0) // si i divide a "a" y "b"
                    mcd=i;        // i será el nuevo mcd.

        return(mcd);
    }

      static int minimo(int a, int b){
        int min;

        if(a>b)
            min=b;
        else
```

```
            min=a;

        return(min);
    }



    public static void main(String[] args) {
        int a, b, mcm;

        System.out.print("Introduce a: ");
        a=Entrada.entero();

        System.out.print("Introduce b: ");
        b=Entrada.entero();

        System.out.println("");

        mcm = min_comun_multiplo (a, b);

        System.out.println("El mcm de "+a+" y "+b+" es: "+mcm);

    }
}
```

15.Ídem con tres números.

```
package bol06ej15;

public class Main {

    // el mínimo común múltiplo de a, b y c, es el número más pequeño que
    // es divisible por a, b y c.
```

```
// Para calcularlo podríamos utilizar algún algoritmo existente o hacerlo
// un poco por la "cuenta de la vieja".
// La idea es elegir el mayor de ellos, y multiplicarlo por i (1..), hasta que
// el número resultante sea divisible por todos

static int min_comun_multiplo (int a, int b, int c)
{
    int mcm=1;
    int max;
    int i;


    max = maximo (a, maximo(b,c));


    mcm =max;   // en principio el mcm es el mayor de los dos números
    i=1;

    while (mcm%a!=0 || mcm%b!=0 || mcm%c!=0)   // mientras el mcm no sea divisible por todos
    {
        i++;
        mcm=max*i;      // el nuevo mcm sera el mayor por i

    }


    return(mcm);
}

  static int maximo (int a, int b){
    int max;

    if(a>b)
        max=a;
    else
        max=b;
```

```
        return(max);
    }



    public static void main(String[] args) {
        int a, b, c, mcm;



        System.out.print ("Introduce a: ");
        a=Entrada.entero();

        System.out.print ("Introduce b: ");
        b=Entrada.entero();

        System.out.print ("Introduce c: ");
        c=Entrada.entero();

        System.out.println("");

        mcm = min_comun_multiplo (a, b, c);

        System.out.println("El mcm es: " +mcm);

    }
}
```

```
package bol06ej15;

public class Main {

    // el mínimo común múltiplo de a y b, es el número más pequeño que
```

```
    // es divisible por a y b.
    // Para calcularlo podríamos utilizar algún algoritmo existente o hacerlo
    // un poco por la "cuenta de la vieja".
    // La idea es elegir el mayor de a y b, y multiplicarlo por i, hasta que
    // el número resultante sea divisible por el menor(a,b)

    static int min_comun_multiplo (int a, int b)
    {
        int mcm=1;
        int max;
        int i;


        max = maximo (a, b);


        mcm =max;   // en principio el mcm es el mayor de los dos números
        i=1;

        while (mcm%a!=0 || mcm%b!=0)   // si el mcm no es divisible por a y b
        {
            i++;
            mcm=max*i;      // el nuevo mcm sera el mayor por i

        }


        return(mcm);
    }

      static int maximo (int a, int b){
        int max;

        if(a>b)
            max=a;
        else
```

```
            max=b;

        return(max);
    }



    public static void main(String[] args) {
        int a, b, c, mcm;



        System.out.print ("Introduce a: ");
        a=Entrada.entero();

        System.out.print ("Introduce b: ");
        b=Entrada.entero();

        System.out.print ("Introduce c: ");
        c=Entrada.entero();

        System.out.println("");

        // reutilizamos la función mcm pensada para dos números y aprovechamos
        // la propiedad conmutativa del mcm.

       mcm = min_comun_multiplo (a, b);
       mcm = min_comun_multiplo (mcm, c);

        System.out.println("El mcm es: " +mcm);

    }
}
```

16.Ídem con una tabla.

```
package bol06ej16;

public class Main {

    // el mínimo común múltiplo de a y b, es el número más pequeño que
    // es divisible por ambos.
    // Para calcularlo podríamos utilizar algún algoritmo existente o hacerlo
    // un poco por la "cuenta de la vieja".
    // La idea es elegir el mayor de ellos, y multiplicarlo por i (1..), hasta que
    // el número resultante sea divisible por ambos

    static int min_comun_multiplo (int a, int b)
    {
        int mcm=1;
        int max;
        int i;


        max = maximo (a, b);


        mcm =max;   // en principio el mcm es el mayor de los dos números
        i=1;

        while (mcm%a!=0 || mcm%b!=0)   // mientras el mcm no sea divisible por todos
        {
            i++;
            mcm=max*i;      // el nuevo mcm sera el mayor por i

        }


        return(mcm);
```

```
    }

      static int maximo (int a, int b){
        int max;

        if(a>b)
            max=a;
        else
            max=b;

        return(max);
    }



    public static void main(String[] args) {
        int t[], mcm;

        t = new int[4];

        for (int i = 0; i < t.length; i++)
            t[i]=(int)(Math.random()*100+1);

        System.out.println("Los datos son:");

        for (int i = 0; i < t.length; i++)
            System.out.print(t[i]+"  ");;


        System.out.println("");

        mcm = t[0];

        for (int i = 0; i < t.length; i++)
            mcm = min_comun_multiplo (t[i],mcm);
```

```
        System.out.println("El mcm es: " +mcm);

    }
}
```

17.Escriba una función que decida si dos números enteros positivos son amigos. Dos números son amigos, si la suma de sus divisores (distintos de ellos mismos) son iguales.

```
package bol06ej17;

public class Main {

    // la función es_primo indica si el número pasado es o no primo
    // recordamos que un número primo es solo divisible por el mismo y 1
    static boolean es_primo(int num)
    {
        boolean primo;
        int i;

        primo=true; // suponemos que el número es primo


        // este algoritmo se puede mejorar sabiendo que si un número no es
        // divisible entre 2 y su raíz cuadrada, entonces ya no será divisible
        // por ningún otro números -> será primo
        //
        // con esta mejora podemos ahorrar muchas vueltas del while para
        // números grandes

        i=2;
        while(i<num && primo==true)
        {
```

```
            if( num %i == 0) // si es divisible
                primo=false; // si hemos entrado aquí significa que el número no es primo

            i++;
        }

        return(primo);
    }



    // esta función me devuelve la suma de los divisores propios.
    // Es decir cualquier número que divida a num en el rango 1..num-1
    //
    // un ejemplo:
    // los divisores propios de 24 son: 1, 2, 3, 4, 6, 8, 12


    static int suma_divisores_propios (int num){
        int suma;

        suma=0;

        for (int i=1;i<num;i++) // al ser hasta i<num no tenemos en cuenta el propio num
            if(num %i == 0) // si i divide a num
                    suma+=i;     // acumulamos i


        return(suma);
    }



    public static void main(String[] args) {
        int a,b;
```

```
        System.out.print("Introduce a: ");
        a=Entrada.entero();

        System.out.print("Introduce b: ");
        b=Entrada.entero();

        if (a==suma_divisores_propios (b) && b==suma_divisores_propios (a))
            System.out.println(a+ " y " +b+ " son amigos.");
        else
            System.out.println(a+" y "+b+" no son amigos...\nLa siguiente vez prueba con 220, 284.");

    }
}
```

18.Diseña una función (en adelante **DUF**) que decida si un número es primo.

```
package bol06ej18;

public class Main {

     static boolean es_primo(int num){
        boolean primo;
        int i;

        primo=true;

        i=2;

        while(i<num && primo==true)
        {
            if(num%i==0)        // si num es divisible por i
                primo=false;    // si llego aquí es que num es divisible-> no es primo
```

```
            i++;
        }

        return(primo);
    }

    public static void main(String[] args) {
         boolean res;
        int num;

        System.out.print("Introduce numero: ");
        num=Entrada.entero();

        if(es_primo(num))
            System.out.println("Es primo");
        else
            System.out.println("No es primo");
    }
}
```

19.DUF que calcule a^n.

```
package bol06ej19;

public class Main {

    static int a_elevado_n(int a, int n){
        int res;

        res=1; // el resultado se inicializa a 1, ya que iremos multiplicando

        if(n==0)  // por definición cualquier número elevado a 0 es 1
```

```
            res=1;
        else
            for (int i=1;i<=n;i++)
                res=res*a;


        return(res);
    }

    public static void main(String[] args) {
        int num,exp,res;

        System.out.print("Introduzca base: ");
        num=Entrada.entero();

        System.out.print("Introduzca su exponente: ");
        exp=Entrada.entero();

        res=a_elevado_n (num,exp);
        System.out.println(num + " elevado a " + exp +" = " +res);


    }
}
```

20.DUF que muestre en binario un número entre 0 y 255.

```
package bol06ej20;

public class Main {

    static void binario(int num)
    {
```

```
        int t[];    // en t guardaremos los dígitos binarios
        int cont;

        t = new int [8];    // un número en binario entre 0 y 255 tiene 8 bits



        if (num<0 || 255<num)
            System.out.println("Número fuera de rango (0..255)");
        else
        {
            cont =0;
            if (num==0)
            {
                t[cont]=0;
                cont++;
            }

            // iremos dividiendo y cogiendo el resto

            while(num!=0)
            {
                t[cont] =num%2;
                num =num/2;
                cont++;
            }

            System.out.println("En binario: ");

            // como t tiene los dígitos en orden inverso

            for (int i=cont-1;i>=0;i--)
                System.out.print(t[i]);

            System.out.println("");
        }
```

```
    }

    public static void main(String[] args) {
        int num;

        System.out.print("Introduzca un numero (0..255): ");
        num=Entrada.entero();

        binario(num);
    }
}
```

21. Escriba una función que sume los n primeros números impares.

```
package bol06ej21;

public class Main {

    static int suma_n_impares (int n)
    {
        int suma=0;

        for (int i =1; i <=n ; i++)
            suma += 2*i-1;       // así calculamos el i-ésimo impar


        return (suma);
    }

    public static void main(String[] args) {
        int n;

        System.out.print("Introduzca un numero: ");
```

```
        n =Entrada.entero();

        System.out.println("La suma de los " +n+ " primeros impares es: " +suma_n_impares (n));

    }
}
```

```
package bol06ej21;

public class Main {

    static int suma_n_impares (int n)
    {
        int suma=0;

        for (int i =1; i <= 2*n-1 ; i+=2)   // el for irá 1, 3, 5, ....
                                            // hasta 2*n-1 (el n-ésimo impar)
            suma += i;


        return (suma);
    }

    public static void main(String[] args) {
        int n;

        System.out.print("Introduzca un numero: ");
        n =Entrada.entero();

        System.out.println("La suma de los " +n+ " primeros impares es: " +suma_n_impares (n));

    }
}
```

22. Dado el valor de un ángulo, sería interesante saber su seno, coseno y tangente. Escribir una función que muestre en pantalla los datos anteriores.

```
package bol06ej22;

public class Main {

    static void informacion_angulo (double n)
    {

        // como las funciones que calculan el seno, coseno y tangente trabajan en
        // radianes, hemos de pasar n de grados a radianes

        n = Math.PI/180*n;      // otra forma sería n = Math.toRadians(n);

        System.out.println("seno: " + Math.sin(n));
        System.out.println("coseno: " + Math.cos(n));
        System.out.println("tangente: " + Math.tan(n));

    }

    public static void main(String[] args) {
        double angulo;

        System.out.print("Introduzca un ángulo (0..360): ");

        angulo =Entrada.real();


        informacion_angulo (angulo);

    }
}
```

23. Diseñar una función que calcule la distancia euclídea de dos puntos.

```
package bol06ej23;

public class Main {

    static double distancia_euclidea (int x1, int y1, int x2, int y2)
    {
        // aquí no hay más que tirar de la fórmula de la distancia euclídea
        // y desempolvarla de los apuntes

        return (Math.sqrt( Math.pow (x1-x2, 2) + Math.pow (y1-y2, 2)));

    }

    public static void main(String[] args) {
        int x1, y1;    // primer punto
        int x2, y2;    // el otro punto
        double l;         // distancia euclídea


        System.out.println("Punto 1");
        System.out.print("x: ");
        x1 = Entrada.entero();
        System.out.print("y: ");
        y1 = Entrada.entero();

        System.out.println("\nPunto 2");
        System.out.print("x: ");
        x2 = Entrada.entero();
        System.out.print("y: ");
        y2 = Entrada.entero();


        l =distancia_euclidea (x1,y1, x2,y2);
```

```
        System.out.println("\nDistancia euclídea: " +l);

    }
}
```

24. DUF a la que se le pasa como parámetro una tabla que debe rellenar. Se leerá por teclado una serie de números: guardaremos solo los pares e ignoraremos los impares. También hay que devolver la cantidad de impares ignorados.

```
package bol06ej24;

public class Main {

    static int rellena_tabla_pares(int t[]){
        int i,num, impares_ignorados;

        i = 0;
        impares_ignorados = 0;

        while (i<t.length)  // terminaremos de rellenar la tabla cuando el número
                            // de pares sea igual que el tamaño de la tabla
        {
            System.out.print("Introduzca número: ");
            num = Entrada.entero();

            if(num %2 == 0)     // si es par
            {
                t[i] = num;     // lo guardamos
                i++;
                }
            else
                impares_ignorados++;
        }
```

```
        return(impares_ignorados);
    }


    public static void main(String[] args) {
        int igno,t[];
        t = new int[5];

        igno = rellena_tabla_pares(t);

        System.out.println("El numero de impares ignorados es de: " +igno);

        System.out.println("La tabla queda: ");

        for(int i=0;i<5;i++)
            System.out.print(t[i]+ "  ");

        System.out.println("");

    }
}
```

25. DUF a la que se le pasa una tabla de enteros y un número. Debemos buscar el número en la tabla e indicar si se encuentra o no.

```
package bol06ej25;

public class Main {

    static boolean busca(int t[],int n)
    {
        int i;
```

```
        boolean esta;

        i=0;

        while (i<t.length && t[i]!=n) // si no es t[i], paso al siguiente
                                      // con cuidado de no salirme de la tabla
            i++;


        // cuando termina el mientras, puede ser por dos motivos:
        // - que he buscado por toda la tabla sin encontrarlo y al final me salgo de la tabla
        // - o que lo he encontrado

        if (i<t.length) // si no llego al final de la tabla es por que lo he encontrado
                        // t[i] == num no es válido, ya que si i está fuera de rango da un error
            esta =true;
        else
            esta =false;


        return(esta);
    }

    public static void main(String[] args) {
        int a[]={1, 12, 38, 5, 11}; // es un ejemplo de posibles valores
        int num;
        boolean esta;


        System.out.println("La tabla es: ");
        for (int i=0;i<5;i++)
            System.out.println(a[i]);


        System.out.print("\introduce numero a buscar: ");
        num=Entrada.entero();
```

```
        esta=busca(a,num);
        if (esta)
            System.out.println("EL número está");
        else
            System.out.println("El número no está");

    }
}
```

26. Igual que el ejercicio anterior, pero suponiendo que la tabla no está siempre llena, y el número de elementos se pasa también como parámetro.

```
package bol06ej26;

public class Main {

    static boolean busca(int t[],int n, int tam)
    {
        int i;
        boolean esta;

        i=0;

        while (i<tam && t[i]!=n) // si no es t[i], paso al siguiente
                                 // con cuidado de no salirme de la tabla
            i++;


        // cuando termina el mientras, puede ser por dos motivos:
        // - que he buscado por toda la tabla sin encontrarlo y al final me salgo de la tabla
        // - o que lo he encontrado
```

```
        if (i<tam) // si no llego al final de la tabla es por que lo he encontrado
                        // t[i] == num no es válido, ya que si i está fuera de rango
                        // t[i] es basura
            esta =true;
        else
            esta =false;


        return(esta);
    }

    public static void main(String[] args) {
        int a[]= new int [20];
        int tam;
        int num;
        boolean esta;


        a[0] = 1;
        a[1] = 12;
        a[2] = 38;
        a[3] = 5;
        a[4] = 11;

        tam = 5;

        System.out.println("La tabla es: ");
        for (int i=0;i<tam;i++)
            System.out.println(a[i]);


        System.out.print("\nIntroduce numero a buscar: ");
        num=Entrada.entero();

        esta=busca(a, num, tam);
```

```
        if (esta)
            System.out.println("El número está");
        else
            System.out.println("El número no está");

    }
}
```

27. Diseñar la función opera_tabla, a la que se le pasa dos tablas, el número de elementos útiles y que operación se desea realizar: sumar, restar, multiplicar o dividir (mediante un carácter: 's', 'r', 'm', 'd'). La función debe devolver una tabla con los resultados.

```
package bol06ej27;

public class Main {

   static int[] opera_tabla(int a[], int b[], char opc, int nelem){
       int i, result[];

       result = new int[nelem];

       switch(opc)
       {
            case 's':
               for(i=0;i<=nelem-1;i++)
                   result[i] = a[i] + b[i];
               break;

           case 'r':
               for(i=0;i<=nelem-1;i++)
                   result[i] = a[i] - b[i];
               break;
```

```
            case 'm':
                for(i=0;i<=nelem-1;i++)
                    result[i] =a[i] * b[i];
                break;

            case 'd':
                for(i=0;i<=nelem-1;i++)
                    result[i] = a[i] / b[i];
                break;
        }

        return (result);
    }
     public static void main(String[] args) {
         int num_datos_utiles;
         char operacion;
         int tabla1[], tabla2[], resultado[];

         tabla1 = new int[10];
         tabla2 = new int[15];

         tabla1[0] =4;
         tabla1[1] =7;
         tabla1[2] =2;
         tabla1[3] =7;

         tabla2[0] =-3;
         tabla2[1] =3;
         tabla2[2] =6;
         tabla2[3] =17;

         num_datos_utiles =4;

         System.out.println("tabla1    tabla2");

         for(int i=0;i<num_datos_utiles;i++)
```

```
            System.out.println (tabla1[i]+ "       " +tabla2[i]);



        System.out.println("Operación (s, r, m, d): ");
        operacion = Entrada.caracter();

        resultado =opera_tabla (tabla1, tabla2, operacion, num_datos_utiles);


        System.out.println("El resultado de la operación es:");

        for(int i=0;i<num_datos_utiles;i++)
            System.out.println (resultado[i]);
    }
}
```

28. DUF que ordene la tabla que se le pasa.

```
package bol06ej28;

public class Main {

    static void ordenar(int a[]){
        int tam=a.length;
        int aux;

        // ordenaremos utilizando la ordenación por intercambio
        for (int i=0;i<tam-1-1;i++){
            for (int j=0;j<tam-i-1;j++){
                if(a[j]>a[j+1]) // si el elemento j es mayor que el j+1
                {
                    aux=a[j];       // los intercambiamos
```

```
                    a[j]=a[j+1];
                    a[j+1]=aux;
                }
            }

        }
    }
    static void mostrar_tabla(int t[]){
        int tam=t.length;

        for (int i=0;i<tam;i++){
            System.out.print(t[i] + "   ");
        }
    }

    public static void main(String[] args) {
        int t[]=new int[8];
        for (int i=0;i<8;i++){
            t[i]=(int)(Math.random()*100)+1;
        }

        System.out.println("Tabla aleatoria");
        mostrar_tabla(t);
        System.out.println("\n\nTabla ordenada");
        ordenar(t);
        mostrar_tabla(t);

    }
}
```

29. DUF que toma como parámetros dos tablas. La primera con los 6 números de una apuesta de la primitiva, y la segunda con los 6 números ganadores. La función debe devolver el número de aciertos.

```
package bol06ej29;

public class Main {

    /* algunas mejoras propuestas para el alumno son:
     * - comprobar que no hay números repetidos en las tablas
     * - comprobar que los números están en el rango 1..49
     * - mirar el tamaño de las tablas, que debe ser 6
     */

    static int primitiva(int apuesta[], int premiado[]){
        int aciertos;

        int a;    // utilizaremos a como índice de la tabla apuesta
        int p;    // y p para recorrer premiado


        aciertos=0;

        for (a=0;a<apuesta.length;a++) // recorremos la tabla de apuesta
        {
            p=0;    // para cada número de la apuesta recorremos premiado

            // se podría hacer con un for, pero con el while evitamos vueltas innecesarias
            while(p<premiado.length && apuesta[a]!=premiado[p])
                p++;

            if(p<premiado.length)   // si p indica un elemento de la tabla
                aciertos++;         // tenemos un acierto más

        }
```

```
        return(aciertos);
    }
    public static void main(String[] args) {
        int primitiva[]=new int[6];
        int apuesta[]=new int[6];
        int aciertos=0;


        System.out.println("Su apuesta es: ");
        for (int i=0;i<primitiva.length;i++)
        {
            primitiva[i]=(int)(Math.random()*49+1);
            System.out.print(primitiva[i] + "    ");
        }


        System.out.println("\nLa combinación ganadora es: ");
        for (int i=0;i<apuesta.length;i++)
        {
            apuesta[i]=(int)(Math.random()*49+1);
            System.out.print(apuesta[i] +"    ");
        }

        aciertos=primitiva(primitiva,apuesta);

        System.out.println("\n\nTiene "+aciertos+" aciertos\n");

        if (aciertos == 0)
            System.out.println("Lo importante no es ganar... es participar.\n");
    }
}
```

30. DUF recursiva que calcule a^n.

```
package bol06ej30;

public class Main {

    /* sobrecargamos la función para que funcione tanto con bases enteras
     * como reales
     *
     */
    static int potencia (int a, int n) {
        int res;

        if (n == 0) // el caso base: cuando el exponente es 0
            res = 1;
        else
            // caso recursivo: a^n = a *a^n-1

            res = a * potencia(a, n - 1);


        return (res);
    }


    static double potencia (double a, int n) {
        double res;
        if (n == 0)
            res = 1;
        else
            res = a * potencia(a, n - 1);


        return (res);
    }
```

```
    public static void main(String[] args) {
        double num, resultado;
        int potencia;
        System.out.print("Introduzca base (real): ");
        num = Entrada.real();
        System.out.print("Introduzca la potencia: ");
        potencia = Entrada.entero();

        resultado = potencia(num, potencia);

        System.out.println("El resultado es: " + resultado);
    }
}
```

31. Calcular el factorial de n recursivamente.

```
package bol06ej31;

public class Main {

    static int factorial(int num){
        int res;

        if(num==0)  // caso base: 0! es igual a 1
            res=1;
        else
            res=num*factorial(num-1);   // n!= n*(n-1)*(n-2)... un ejemplo 3!=3*2*1
                                        // también ocurre que n!=n*(n-1)!
                                        // como ejemplo 4!=4*3!

        return(res);
    }
```

```
    public static void main(String[] args) {
        int num,resultado;

        System.out.print("Introduzca el numero: ");
        num=Entrada.entero();

        resultado=factorial(num);

        System.out.println(num+"! es igual a "+resultado);
    }
}
```

32. DUF que calcule el valor máximo de una tabla de forma recursiva.

```
package bol06ej32;

public class Main {

    /** la función máximo busca el mayor número entre los elementos de t,
     * a partir de las posición pos.
     */
    static int maximo (int t[], int pos)
    {
        int res;

        if(pos==t.length-1)  // caso base: pos indica el último elemento de t
                             // en este caso este será el máximo
            res=t[pos];
        else
        {
            int k;
```

```
            k = maximo (t,pos+1); // k será el mayor desde la posición pos+1 hasta el último elemento

            if (t[pos]>k)   // si t[pos] es mayor que k
                res = t[pos];   //t[pos] es el máximo
            else
                res = k;        // en caso contrario será k el máximo
        }

        return(res);
    }

    // el usuario utilizará esta función por comodidad

    static int maximo (int t[])
    {
        return (maximo (t,0));
    }


    public static void main(String[] args) {
        int datos[];
        int max;


        datos = new int[10];

        for (int i = 0; i < datos.length; i++)
            datos[i] = (int) (Math.random()*1000+1);

        System.out.println("Los datos son:");

        for (int i = 0; i < datos.length; i++)
            System.out.print(datos[i] + "  ");


        max =maximo(datos);
```

```
        System.out.println("\n\nEl máximo es: " + max);
    }
}
```

```
package bol06ej32;

public class Main {

    public static void main(String[] args)
    {
    int t[]={7, 2, 10, 5, 1, 9}; // un ejemplo para probar la función
    int max;

    max =maximo(t);

    System.out.println("El máximo es: " +max);

    }

    // El máximo de una tabla será el máximo entre el primer elemento de la
    // tabla y el resto de la tabla (es decir del segundo elemento hasta el
    // último).
    //
    // Vamos acortando la tabla, indicando el primer índice donde se empezará a
    // buscar el máximo.
    // El caso base será una tabla donde solo se busca en un elemento elemento.
    // Está claro que ese único elemento será el mayor de la tabla (de 1 elemento).

    static int maximo(int t[])
    {
        return (maximo(t, 0, t.length-1 ));
    }
```

```
    static int maximo(int t[], int desde, int hasta)
    {
        int mayor;

        if (desde == hasta) // caso base
                            // la tabla solo tiene un elemento
            mayor=t[desde];
        else
        {
            mayor = maximo(t, desde+1, hasta);
            if (mayor < t[desde])
                mayor = t[desde];
        }
        return (mayor);
    }
}
```

33. DUF que calcule el n-ésimo término de la serie de Fibonacci. En esta serie el n-ésimo valor se calcula sumando los dos valores anteriores. Es decir fibonacci(n) = fibonacci(n-1)+fibonacci(n-2), siendo fibonacci(0)=1 y fibonacci(1)=1.

```
package bol06ej33;

public class Main {

    static int fibo(int num)
    {
        int res;

        if(num==0)      // primer caso base
            res=0;
        else{
            if(num==1)  // segundo caso base
```

```
                res=1;
            else
                res=fibo(num-1)+fibo(num-2);    // caso general recursivo
        }

        return(res);

    }

    public static void main(String[] args) {
        int num,resultado;

        System.out.print("Vamos calcular fibonacci(n).\nIntroduzca n (se recomienda n<40):  ");
        num=Entrada.entero();


        resultado=fibo(num); // si n es muy grande esto puede tardar bastante.

        System.out.println("\nfibonacci(" + num + ") = " +resultado);

    }
}
```

34. Igual que el ejercicio anterior, pero pudiendo configurar los valores de los dos primeros término de la serie.

```
package bol06ej34;

public class Main {

    /**
     * sobrecargamos la función para que funcione de la siguiente forma:
     * - si solo se le pasa el término a calcular: utiliza los casos bases típicos
```

```
     * - se le puede pasar los valores de los casos bases: fibo(0) y fibo(1)
     */

    static int fibo(int num)
    {
        int res;

        if(num==0)      // primer caso base
            res=0;
        else{
            if(num==1)  // segundo caso base
                res=1;
            else
                res=fibo(num-1)+fibo(num-2);    // caso general recursivo
        }

        return(res);

    }

    static int fibo(int num, int fibo0, int fibo1)
    {
        int res;

        if(num==0)      // primer caso base, que tendrá el valor indicado por el usuario
            res=fibo0;
        else{
            if(num==1)  // segundo caso base, también configurable
                res=fibo1;
            else
                res=fibo(num-1,fibo0, fibo1)+fibo(num-2, fibo0, fibo1);
                            // caso general recursivo
                            // hemos de acordarnos de utilizar la función fibo que
                            // tiene 3 parámetros
        }
```

```
        return(res);

    }

    public static void main(String[] args) {
        int num,resultado;
        int fibo0,fibo1;

        System.out.println("Vamos calcular fibonacci(n)\n");

        System.out.print("Introduzca el valor de fibonacci(0): ");
        fibo0 = Entrada.entero();

        System.out.print("Introduzca el valor de fibonacci(1): ");
        fibo1 = Entrada.entero();

        System.out.print ("\nIntroduzca n (se recomienda n<40):  ");
        num=Entrada.entero();


        // si n es muy grande esto puede tardar bastante.
        resultado=fibo(num, fibo0, fibo1);

        System.out.println("\nfibonacci(" + num + ") = " +resultado);
    }
}
```

35. DUF que realice la búsqueda dicotómica en una tabla, de forma recursiva.

```
package bol06ej35;

public class Main {
```

```
    // a la función se le pasa, la tabla, el elemento a buscar, y la primera
    // y última posición donde buscar.
    static int busca (int t[], int elem, int primero, int ultimo)
    {
        int pos;

        if(primero >= ultimo)  // caso base: solo hay un elemento donde buscar

            if (t[primero]==elem)
                pos =primero;
            else
                pos =-1;
        else
        {
            int pos1, pos2;


            // llamada recursiva

            //buscamos en la primera mitad de la tabla: 0..mitad
            pos1 = busca (t, elem, primero, (primero+ultimo)/2);

            // buscamos en la segunda parte de la tabla: mitad+1..ultimo
            // se pone mitad+1, por que el elemento mitad ya pertenece a la
            // primera parte... por no repetirlo
            pos2 = busca (t, elem, (primero+ultimo)/2+1, ultimo);

            if (pos1 != -1)     // si lo encuentro en la primera parte
                pos =pos1;
            else
                pos =pos2;  // en caso contrario debo encontrarlo en la segunda parte

            // en caso de no encontrarse pos1 y pos2 serán -1, y se cogerá el valor de pos2 (-1)
        }

        return(pos);
```

```
    }

    // el usuario utilizará esta función por comodidad
    // solo es necesario pasarle la tabla y el elemento a buscar
    // devuelve el índice del elemento si lo encuentra o -1 en caso contrario
    static int busca (int t[], int elem)
    {
        return (busca (t, elem, 0, t.length-1));
    }


    public static void main(String[] args) {
        int datos[];
        int num;
        int pos;


        datos = new int[10];

        // para no teclearlos, cagamos datos aleatorios
        for (int i = 0; i < datos.length; i++)
            datos[i] = (int) (Math.random()*1000+1);

        System.out.println("Los datos son:");

        for (int i = 0; i < datos.length; i++)
            System.out.print(datos[i] + "  ");


        System.out.print("\n\nElemento a buscar: ");
        num =Entrada.entero();

        // llamamos a la función buscar
        pos =busca(datos, num);

        if (pos == -1)
```

```
            System.out.println("\n\nNo encontrado");
        else
            System.out.println("\n\nEncontrado en la posición: " +pos);
    }
}
```

36. DUF que toma una tabla bidimensional de enteros, representando un tablero de ajedrez. Disponemos de las constantes PB (peón blanco), TN (torre negra), etc. (P, T, C, A, R, D). Dicho módulo debe devolver un valor booleano, que indique si el rey negro está amenazado.

```
package bol06ej36;

public class Main {

    final static int V=-1; // escaque vacío
    final static int P=0;  // peón
    final static int T=1;  // torre
    final static int C=2;  // caballo
    final static int A=3;  // alfil
    final static int D=4;  // dama
    final static int R=5;  // rey

                    // igual para los negros
    final static int PN=10;
    final static int TN=11;
    final static int CN=12;
    final static int AN=13;
    final static int DN=14;
    final static int RN=15;


    public static void main(String[] args)
    {
```

```
        final int tablero[][] = {  {V,  V,  V,  V,  V,  V,  V,  V},
                                   {V,  V,  V,  V,  V,  V,  V,  V},
                                   {V,  V,  V,  V,  V,  V,  V, AN},
                                   {V,  V,  V,  V,  V,  V,  T,  V},
                                   {V,  V, DN,  V,  V,  V,  V,  V},
                                   {V,  V,  V, CN,  V,  V,  V,  V},
                                   {V,  V,  P,  P,  V,  V,  V,  V},
                                   {V,  V,  R,  V,  V,  V,  V,  V}};

        // hay que tener con el tablero: como lo visualizamos y sus índices.
        // En realidad el tablero no es como se ve arriba. Ya que hemos de imaginarlo
        // rotado 90 grados hacia la izquierda.

        boolean amenaza;  // una posible mejora es indicar en la variable
                          // amenaza la primera pieza contraria nos está
                          // acechando e indicar con un escaque vacío que
                          // no existe peligro

        amenaza = jaque (tablero);

        // mostramos se existe amenaza
        System.out.println("Jaque: " +amenaza);
    }

static boolean jaque(int tablero[][])
    {
        boolean amenaza=false;
        int pieza;          // pieza que no está amenazando

        int pos_rey[];  // posición del rey en el tablero
        pos_rey = new int [2];

        pos_rey = busca_rey (tablero);

        // ahora iremos viendo las posibles amenazas una a una:
```

```
        // en primer lugar veremos si nos amenaza un caballo:
        amenaza =amenaza_caballo (tablero, pos_rey);

        // AHORA
        // miraremos hacia la derecha (dx:1, dy:0)
        // aquí no puede amenazar una torre o dama
        pieza =primera_pieza(tablero, pos_rey, 1, 0);
        if (pieza ==DN || pieza ==TN)
            amenaza=true;

        // miraremos hacia la izquierda (dx:-1, dy:0)
        // aquí no puede amenazar una torre o dama
        pieza =primera_pieza(tablero, pos_rey, -1, 0);
        if (pieza ==DN || pieza ==TN)
            amenaza=true;

        // miraremos hacia arriba (dx:0, dy:1)
        // aquí no puede amenazar una torre o dama
        pieza =primera_pieza(tablero, pos_rey, 0, 1);
         if (pieza ==DN || pieza ==TN)
            amenaza=true;

        // miraremos hacia abajo (dx:0, dy:-1)
        // aquí no puede amenazar una torre o dama
        pieza =primera_pieza(tablero, pos_rey, 0, -1);
        if (pieza ==DN || pieza ==TN)
            amenaza=true;

       // miraremos en la diagonal derecha arriba (dx:1, dy:1)
        // aquí no puede amenazar un alfil o una dama
        pieza =primera_pieza(tablero, pos_rey, 1, 1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;

        // miraremos en la diagonal derecha abajo (dx:1, dy:-1)
        // aquí no puede amenazar un alfil o una dama
```

```
        pieza =primera_pieza(tablero, pos_rey, 1, -1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;

        // miraremos en la diagonal izquierda arriba (dx:-1, dy:1)
        // aquí no puede amenazar un alfil o una dama
        pieza =primera_pieza(tablero, pos_rey, -1, 1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;


        // miraremos en la diagonal izquierda abajo (dx:-1, dy:-1)
        // aquí no puede amenazar un alfil o una dama
        pieza =primera_pieza(tablero, pos_rey, -1, -1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;

        // falta que nos amenace un peón

        // el posible peón se encuentra arriba a la derecha o a la izquierda
        if ( (pos_rey[0]+1<8 && pos_rey[1]+1<8 && tablero[pos_rey[0]+1][pos_rey[1]+1]==PN) ||
             (0<=pos_rey[0]-1 && pos_rey[1]+1<8 && tablero[pos_rey[0]-1][pos_rey[1]+1]==PN) )
             amenaza=true;

        return (amenaza);
    }


    // esta función busca el rey blanco y devuelve su posición
    // en una tabla de dos elementos. Siendo posicion[0] la columna y
    // posicion[1] la fila.

    static int[] busca_rey (int tablero[][])
    {
        int i,j;
        int posicion[];
```

```
        posicion = new int [2];

        for (i=0; i <8; i++)
            for (j=0; j <8; j++)
                if (tablero[i][j]==R)
                {
                    posicion[0] =i;
                    posicion[1] =j;
                }

        return (posicion);
    }

    // esta función busca la primera pieza que existe desde la posición pos,
    // en la dirección indicada por dx, dy.
    // Los valores de dx, dy son:
    //  dx  dy  dirección
    //  1    1   diagonal derecha arriba
    //  1    0   derecha
    //  1   -1  diagonal derecha abajo
    //  0    1   hacia arriba
    //  0    0   ESTE CASO NO SE DARÁ NUNCA
    //  0   -1  hacia abajo
    //  -1   1   diagonal izquierda arriba
    //  -1   0   hacia la izquierda
    //  -1  -1   diagonal izquierda abajo

    static int primera_pieza (int tablero[][], int pos[], int dx, int dy)
    {
        int posx, posy; //posición del tablero en la que estamos mirando
        int pieza;

        posx =pos[0];
        posy =pos[1];
```

```
        pieza = V;  // en principio suponemos que no hay ninguna pieza

        // damos el primer paso: es decir pasamos a la primera casilla
        // después del rey.
        posx +=dx;
        posy +=dy;

        // mientras no nos salgamos del tablero y no encontremos una pieza
        while ( 0<=posx && posx<=7 &&
                0<=posy && posy<=7 &&
                pieza ==V)
        {
            pieza = tablero[posx][posy];

            posx += dx;
            posy += dy;
        }

        return (pieza);
    }

static boolean amenaza_caballo (int tablero[][], int pos[])
{
    boolean amenaza=false;

    // Hay que tener cuidado al comprobar los caballos de no salirse del
    // tablero


    // Desde la posición actual vemos los posibles ocho posiciones desde donde
    // puede amenazarnos un caballo.
    // Algunas de estas posiciones pueden estar "fuera" del tablero
    if ( (    pos[0]+2 <8 &&     pos[1]+1 <8 && tablero[pos[0]+2][pos[1]+1] ==CN) ||
         (    pos[0]+2 <8 && 0<= pos[1]-1    && tablero[pos[0]+2][pos[1]-1] ==CN) ||
         (0<= pos[0]-2    &&     pos[1]+1 <8 && tablero[pos[0]-2][pos[1]+1] ==CN) ||
         (0<= pos[0]-2    && 0<= pos[1]-1    && tablero[pos[0]-2][pos[1]-1] ==CN) ||
```

```
         (    pos[0]+1 <8 &&     pos[1]+2 <8 && tablero[pos[0]+1][pos[1]+2] ==CN) ||
         (    pos[0]+1 <8 && 0<= pos[1]-2    && tablero[pos[0]+1][pos[1]-2] ==CN) ||
         (0<= pos[0]-1    &&     pos[1]+2 <8 && tablero[pos[0]-1][pos[1]+2] ==CN) ||
         (0<= pos[0]-1    && 0<= pos[1]-2    && tablero[pos[0]-1][pos[1]-2] ==CN) )
            amenaza = true;

    return (amenaza);
}

}
```

37. Igual que el ejercicio anterior, pero indicando si existe jaque mate a las negras.

```
package bol06ej37;

public class Main {

    final static int V=-1; // escaque vacío
    final static int P=0;  // peón
    final static int T=1;  // torre
    final static int C=2;  // caballo
    final static int A=3;  // alfil
    final static int D=4;  // dama
    final static int R=5;  // rey

                    // igual para los negros
    final static int PN=10;
    final static int TN=11;
    final static int CN=12;
    final static int AN=13;
    final static int DN=14;
    final static int RN=15;
```

```
public static void main(String[] args)
{
          int tablero[][] = {  { V,  V,  V,  V,  V,  V,  V,  V},
                               { V,  V,  V,  V,  V,  V,  V,  V},
                               { V,  V,  V,  V,  V,  V,  V,  V},
                               { V,  V,  V,  V,  V,  V,  V, DN},
                               { V,  V,  V,  V,  V,  V,  V,  V},
                               {CN,  V,  V, DN,  V,  V,  V,  V},
                               { V,  P,  P, PN,  V,  V,  V,  V},
                               { V,  V,  R,  V,  V,  V,  V,  V}};

    // hay que tener con el tablero: como lo visualizamos y sus índices.
    // En realidad el tablero no es como se ve arriba. Ya que hemos de imaginarlo
    // rotado 90 grados hacia la izquierda.


    int num_jaque=0;
    int pos_rey[];

    boolean amenaza=false;  // una posible mejora es indicar en la variable
                            // amenaza la primera pieza contraria nos está
                            // acechando e indicar con un escaque vacío que
                            // no existe peligro

    pos_rey = busca_rey(tablero);

    if ( jaque (tablero) == true )
    {
        amenaza =true;
        num_jaque=0;
        // intentaremos mover el rey a las casillas adyacentes
        // y comprobar si en la nueva ubicación recibe jaque
        num_jaque += mover_rey (tablero,  0, -1);
        num_jaque += mover_rey (tablero,  0,  1);
        num_jaque += mover_rey (tablero,  1, -1);
```

```
            num_jaque += mover_rey (tablero,  1,  0);
            num_jaque += mover_rey (tablero,  1,  1);
            num_jaque += mover_rey (tablero, -1, -1);
            num_jaque += mover_rey (tablero, -1,  0);
            num_jaque += mover_rey (tablero, -1,  1);
        }


        // mostramos se existe amenaza
        if (amenaza)
            if (num_jaque<8)
                System.out.println("Solo es jaque.");
            else
                System.out.println("Jaque Mate");
        else
            System.out.println("El rey no está amenazado");
    }

static boolean jaque(int tablero[][])
    {
        boolean amenaza=false;
        int pieza;          // pieza que no está amenazando

        int pos_rey[];  // posición del rey en el tablero
        pos_rey = new int [2];

        pos_rey = busca_rey (tablero);

        // ahora iremos viendo las posibles amenazas una a una:

        // en primer lugar veremos si nos amenaza un caballo:
        amenaza =amenaza_caballo (tablero, pos_rey);

        // AHORA
        // miraremos hacia la derecha (dx:1, dy:0)
        // aquí no puede amenazar una torre o dama
```

```
        pieza =primera_pieza(tablero, pos_rey, 1, 0);
        if (pieza ==DN || pieza ==TN)
            amenaza=true;

        // miraremos hacia la izquierda (dx:-1, dy:0)
        // aquí no puede amenazar una torre o dama
        pieza =primera_pieza(tablero, pos_rey, -1, 0);
        if (pieza ==DN || pieza ==TN)
            amenaza=true;

        // miraremos hacia arriba (dx:0, dy:1)
        // aquí no puede amenazar una torre o dama
        pieza =primera_pieza(tablero, pos_rey, 0, 1);
         if (pieza ==DN || pieza ==TN)
            amenaza=true;

        // miraremos hacia abajo (dx:0, dy:-1)
        // aquí no puede amenazar una torre o dama
        pieza =primera_pieza(tablero, pos_rey, 0, -1);
        if (pieza ==DN || pieza ==TN)
            amenaza=true;

      // miraremos en la diagonal derecha arriba (dx:1, dy:1)
        // aquí no puede amenazar un alfil o una dama
        pieza =primera_pieza(tablero, pos_rey, 1, 1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;

        // miraremos en la diagonal derecha abajo (dx:1, dy:-1)
        // aquí no puede amenazar un alfil o una dama
        pieza =primera_pieza(tablero, pos_rey, 1, -1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;

        // miraremos en la diagonal izquierda arriba (dx:-1, dy:1)
        // aquí no puede amenazar un alfil o una dama
```

```
        pieza =primera_pieza(tablero, pos_rey, -1, 1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;


        // miraremos en la diagonal izquierda abajo (dx:-1, dy:-1)
        // aquí no puede amenazar un alfil o una dama
        pieza =primera_pieza(tablero, pos_rey, -1, -1);
        if (pieza ==DN || pieza ==AN)
            amenaza=true;

        // falta que nos amenace un peón

        // el posible peón se encuentra arriba a la derecha o a la izquierda
        if ( (pos_rey[0]+1<8 && pos_rey[1]+1<8 && tablero[pos_rey[0]+1][pos_rey[1]+1]==PN) ||
             (0<=pos_rey[0]-1 && pos_rey[1]+1<8 && tablero[pos_rey[0]-1][pos_rey[1]+1]==PN) )
             amenaza=true;

        return (amenaza);
}


    // esta función busca el rey blanco y devuelve su posición
    // en una tabla de dos elementos. Siendo posicion[0] la columna y
    // posicion[1] la fila.

    static int[] busca_rey (int tablero[][])
    {
        int i,j;
        int posicion[];

        posicion = new int [2];

        for (i=0; i <8; i++)
            for (j=0; j <8; j++)
                if (tablero[i][j]==R)
```

```
                {
                    posicion[0] =i;
                    posicion[1] =j;
                }

        return (posicion);
    }

    // esta función busca la primera pieza que existe desde la posición pos,
    // en la dirección indicada por dx, dy.
    // Los valores de dx, dy son:
    //  dx  dy  dirección
    //  1    1   diagonal derecha arriba
    //  1    0   derecha
    //  1   -1  diagonal derecha abajo
    //  0    1   hacia arriba
    //  0    0   ESTE CASO NO SE DARÁ NUNCA
    //  0   -1  hacia abajo
    //  -1   1   diagonal izquierda arriba
    //  -1   0   hacia la izquierda
    //  -1  -1   diagonal izquierda abajo

    static int primera_pieza (int tablero[][], int pos[], int dx, int dy)
    {
        int posx, posy; //posición del tablero en la que estamos mirando
        int pieza;

        posx =pos[0];
        posy =pos[1];

        pieza = V;  // en principio suponemos que no hay ninguna pieza

        // damos el primer paso: es decir pasamos a la primera casilla
        // después del rey.
        posx +=dx;
        posy +=dy;
```

```
        // mientras no nos salgamos del tablero y no encontremos una pieza
        while ( 0<=posx && posx<=7 &&
                0<=posy && posy<=7 &&
                pieza ==V)
        {
            pieza = tablero[posx][posy];

            posx += dx;
            posy += dy;
        }

        return (pieza);
    }

static boolean amenaza_caballo (int tablero[][], int pos[])
{
    boolean amenaza=false;

    // Hay que tener cuidado al comprobar los caballos de no salirse del
    // tablero


    // Desde la posición actual vemos los posibles ocho posiciones desde donde
    // puede amenazarnos un caballo.
    // Algunas de estas posiciones pueden estar "fuera" del tablero
    if ( (    pos[0]+2 <8 &&     pos[1]+1 <8 && tablero[pos[0]+2][pos[1]+1] ==CN) ||
         (    pos[0]+2 <8 && 0<= pos[1]-1    && tablero[pos[0]+2][pos[1]-1] ==CN) ||
         (0<= pos[0]-2    &&     pos[1]+1 <8 && tablero[pos[0]-2][pos[1]+1] ==CN) ||
         (0<= pos[0]-2    && 0<= pos[1]-1    && tablero[pos[0]-2][pos[1]-1] ==CN) ||
         (    pos[0]+1 <8 &&     pos[1]+2 <8 && tablero[pos[0]+1][pos[1]+2] ==CN) ||
         (    pos[0]+1 <8 && 0<= pos[1]-2    && tablero[pos[0]+1][pos[1]-2] ==CN) ||
         (0<= pos[0]-1    &&     pos[1]+2 <8 && tablero[pos[0]-1][pos[1]+2] ==CN) ||
         (0<= pos[0]-1    && 0<= pos[1]-2    && tablero[pos[0]-1][pos[1]-2] ==CN) )
           amenaza = true;
```

```
    return (amenaza);
}

static int mover_rey (int tablero[][], int dx, int dy)
{
    int existe_jaque=0; // existe_jaque vale 0 si no hay peligro y 1 si
                        // el rey está amenazado o no puede moverse a esta
                        // casilla.

    int pos[];
    pos = new int [2];

    pos = busca_rey(tablero);



    if ( 0<=pos[0]+dx && pos[0]+dx<8  &&
         0<=pos[1]+dy && pos[1]+dy<8  &&
         ( tablero[pos[0]+dx][pos[1]+dy]==V  ||
           tablero[pos[0]+dx][pos[1]+dy]==PN ||
           tablero[pos[0]+dx][pos[1]+dy]==TN ||
           tablero[pos[0]+dx][pos[1]+dy]==CN ||
           tablero[pos[0]+dx][pos[1]+dy]==AN ||
           tablero[pos[0]+dx][pos[1]+dy]==DN ) )
            {
                int pieza;

                // guardamos la pieza que ocupa la posición a ocupar por el rey
                // esta pieza puede ser V (vacío) o una pieza negra que el
                // rey capturará
                pieza=tablero[pos[0]+dx][pos[1]+dy];
                // movemos el rey
                tablero[pos[0]][pos[1]] =V;
                tablero[pos[0]+dx][pos[1]+dy] =R;
                if (jaque(tablero))
                    existe_jaque=1;
```

```
                //volvemos el rey a su posición inicial
                tablero[pos[0]][pos[1]] =R;
                tablero[pos[0]+dx][pos[1]+dy] =pieza;

            }
    else
        // no podemos mover el rey, en la practica esta casilla no es utilizable
        // por el rey para escapar... es lo mismo que una amenaza (jaque).
        existe_jaque=1;

    return (existe_jaque);
}

}
```

Apéndice I
Boletines completos

BOLETÍN 1
Variables y condicionales

1. Pedir los coeficientes de una ecuación se 2° grado, y muestre sus soluciones reales. Si no existen, debe indicarlo.

2. Pedir el radio de un círculo y calcular su área. A=PI*r^2.

3. Pedir el radio de una circunferencia y calcular su longitud.
   L=2*PI*r.

4. Pedir dos números y decir si son iguales o no.

5. Pedir un número e indicar si es positivo o negativo.

6. Pedir dos números y decir si uno es múltiplo del otro.

7. Pedir dos números y decir cual es el mayor.

8. Pedir dos números y decir cual es el mayor o si son iguales.

9. Pedir dos números y mostrarlos ordenados de mayor a menor.

10. Pedir tres números y mostrarlos ordenados de mayor a menor.

11. Pedir un número entre 0 y 9.999 y decir cuantas cifras tiene.

12. Pedir un número entre 0 y 9.999 y mostrarlo con las cifras al revés.

13. Pedir un número entre 0 y 9.999, decir si es capicúa.

14. Pedir una nota de 0 a 10 y mostrarla de la forma: Insuficiente, Suficiente, Bien...

15. Pedir el día, mes y año de una fecha e indicar si la fecha es correcta. Suponiendo todos los meses de 30 días.

16. Pedir el día, mes y año de una fecha e indicar si la fecha es correcta. Con meses de 28, 30 y 31 días. Sin años bisiestos.

17. Pedir el día, mes y año de una fecha correcta y mostrar la fecha del día siguiente. suponer que todos los meses tienen 30 días.

18. Ídem que el ej. 17, suponiendo que cada mes tiene un número distinto de días (suponer que febrero tiene siempre 28 días).

19. Pedir dos fechas y mostrar el número de días que hay de diferencia. Suponiendo todos los meses de 30 días.

20. Pedir una hora de la forma hora, minutos y segundos, y mostrar la hora en el segundo siguiente.

21. Pedir una nota numérica entera entre 0 y 10, y mostrar dicha nota de la forma: cero, uno, dos, tres...

22. Pedir un número de 0 a 99 y mostrarlo escrito. Por ejemplo, para 56 mostrar: cincuenta y seis.

BOLETÍN 2
Condicionales y bucles

1. Leer un número y mostrar su cuadrado, repetir el proceso hasta que se introduzca un número negativo.

2. Leer un número e indicar si es positivo o negativo. El proceso se repetirá hasta que se introduzca un 0.

3. Leer números hasta que se introduzca un 0. Para cada uno indicar si es par o impar.

4. Pedir números hasta que se teclee uno negativo, y mostrar cuántos números se han introducido.

5. Realizar un juego para adivinar un número. Para ello pedir un número N, y luego ir pidiendo números indicando "mayor" o "menor" según sea mayor o menor con respecto a N. El proceso termina cuando el usuario acierta.

6. Pedir números hasta que se teclee un 0, mostrar la suma de todos los números introducidos.

7. Pedir números hasta que se introduzca uno negativo, y calcular la media.

8. Pedir un número N, y mostrar todos los números del 1 al N.

9. Escribir todos los números del 100 al 0 de 7 en 7.

10. Pedir 15 números y escribir la suma total.

11. Diseñar un programa que muestre el producto de los 10 primeros números impares.

12. Pedir un número y calcular su factorial.

13. Pedir 10 números. Mostrar la media de los números positivos, la media de los números negativos y la cantidad de ceros.

14. Pedir 10 sueldos. Mostrar su suma y cuantos hay mayores de 1000€.

15. Dadas las edades y alturas de 5 alumnos, mostrar la edad y la estatura media, la cantidad de alumnos mayores de 18 años, y la cantidad de alumnos que miden más de 1.75.

16. Pide un número (que debe estar entre 0 y 10) y mostrar la tabla de multiplicar de dicho número.

17. Una empresa que se dedica a la venta de desinfectantes necesita un programa para gestionar las facturas. En cada factura figura: el código del artículo, la cantidad vendida en litros y el precio por litro.
    Se pide de 5 facturas introducidas: Facturación total, cantidad en litros vendidos del artículo 1 y cuantas facturas se emitieron de más de 600 €.

18. Igual que el anterior pero suponiendo que no se introduce el precio por litro. Solo existen tres productos con precios:
    1- 0,6 €/litro, 2- 3 €/litro y 3- 1,25 €/litro.

19. Dadas 6 notas, escribir la cantidad de alumnos aprobados, condicionados (=4) y suspensos.

20. Pedir un número N, introducir N sueldos, y mostrar el sueldo máximo.

21. Pedir 10 números, y mostrar al final si se ha introducido alguno negativo.

22. Pedir 5 calificaciones de alumnos y decir al final si hay algún suspenso.

23. Pedir 5 números e indicar si alguno es múltiplo de 3.

BOLETÍN 3
Bucles anidados

1. Realiza detenidamente una traza al siguiente programa y muestra cual seria la salida por pantalla:

```
PROGRAMA ej_1
      VARIABLES
            suma, i, j: ENTERO
      COMIENZO
            PARA i <- 1 HASTA 4
                  PARA j <- 3 HASTA 0 INC -1
                        suma <- i*10+j
                        escribir (suma)
                  FIN PARA
            FIN PARA
FIN
```

2. Realiza una traza del siguiente algoritmo y muestra la salida generada por pantalla.

```
PROGRAMA ej_1
      VARIABLES
            i, j: ENTERO
      COMIENZO
            PARA i <- 1 HASTA 3
                  j <- i+1
                  MIENTRAS j < 4
                        escribir (j-i)
                        j <- j+1
                  FIN MIENTRAS
            FIN PARA
FIN
```

3. Diseña una aplicación que muestre las tablas de multiplicar del 1 al 10.

4. Dibuja un cuadrado de n elementos de lado utilizando *.

5. Necesitamos mostrar un contador con 5 dígitos (X-X-X-X-X), que muestre los números del 0-0-0-0-0 al 9-9-9-9-9, con la particularidad que cada vez que aparezca un 3 lo sustituya por una E.

6. Realizar un programa que nos pida un número n, y nos diga cuantos números hay entre 1 y n que son primos.

BOLETÍN 4
Tablas

1. Leer 5 números y mostrarlos en el mismo orden introducido.

2. Leer 5 números y mostrarlos en orden inverso al introducido.

3. Leer 5 números por teclado y a continuación realizar la media de los números positivos, la media de los negativos y contar el número de ceros.

4. Leer 10 números enteros. Debemos mostrarlos en el siguiente orden: el primero, el último, el segundo, el penúltimo, el tercero, etc.

5. Leer por teclado dos tablas de 10 números enteros y mezclarlas en una tercera de la forma: el 1º de A, el 1º de B, el 2º de A, el 2º de B, etc.

6. Leer los datos correspondiente a dos tablas de 12 elementos numéricos, y mezclarlos en una tercera de la forma: 3 de la tabla A, 3 de la B, otros 3 de A, otros 3 de la B, etc.

7. Leer por teclado una serie de 10 números enteros. La aplicación debe indicarnos si los números están ordenados de forma creciente, decreciente, o si están desordenados.

8. Diseñar una aplicación que declare una tabla de 10 elementos enteros. Leer mediante el teclado 8 números. Después se debe pedir un número y una posición, insertarlo en la posición indicada, desplazando los que estén detrás.

9. Crear un programa que lea por teclado una tabla de 10 números enteros y la desplace una posición hacia abajo (el último pasa a ser el primero).

10. Ídem, desplazar N posiciones (N es introducido por el usuario).

11. Leer 5 elementos numéricos que se introducirán ordenados de forma creciente. Éstos los guardaremos en una tabla de tamaño 10. Leer un número N, e insertarlo en el lugar adecuado para que la tabla continúe ordenada.

12. Leer por teclado una tabla de 10 elementos numéricos enteros y leer una posición (entre 0 y 9). Eliminar el elemento situado en la posición dada sin dejar huecos.

13. Leer 10 enteros. Guardar en otra tabla los elementos pares de la primera, y a continuación los elementos impares.

Realizar dos versiones: una trabajando con los valores y otra trabajando con los índices.

14. Leer dos series de 10 enteros, que estarán ordenados crecientemente. Copiar (fusionar) las dos tablas en una tercera, de forma que sigan ordenados.

15. Leer 10 enteros ordenados crecientemente. Leer N y buscarlo en la tabla. Se debe mostrar la posición en que se encuentra. Si no está, indicarlo con un mensaje.

16. Queremos desarrollar una aplicación que nos ayude a gestionar las notas de un centro educativo. Cada grupo (o clase) está compuesto por 5 alumnos. Se pide leer las notas del primer, segundo y tercer trimestre de un grupo. Debemos mostrar al final: la nota media del grupo en cada trimestre, y la media del alumno que se encuentra en la posición N (N se lee por teclado).

BOLETÍN 5
Tablas n-dimensionales

1. Crear una tabla bidimensional de tamaño 5x5 y rellenarla de la siguiente forma: la posición T[n,m] debe contener n+m. Después se debe mostrar su contenido.

2. Crear y cargar una tabla de tamaño 4x4 y decir si es simétrica o no, es decir, si se obtiene la misma tabla al cambiar filas por columnas.

3. Crear y cargar dos matrices de tamaño 3x3, sumarlas y mostrar su suma.

4. Crear y cargar una tabla de tamaño 3x3, trasponerla y mostrarla.

5. Crear una tabla de tamaño 7x7 y rellenarla de forma que los elementos de la diagonal principal sean 1 y el resto 0.

6. Crear y cargar una tabla de tamaño 10x10, mostrar la suma de cada fila y de cada columna.

7. utilizando dos tablas de tamaño 5x9 y 9x5, cargar la primera y trasponerla en la segunda.

8. Crear una matriz "marco" de tamaño 8x6: todos sus elementos deben ser 0 salvo los de los bordes que deben ser 1. Mostrarla.

9. Hacer lo mismo que el ejercicio anterior, pero con una matriz 9x9x9. Es decir, creamos un cubo con las caras puestas a 1 y el interior a 0.

10. Los siguientes programas piden una serie de datos y tras procesarlos ofrecen unos resultados por pantalla. Mostrar el resultado:

```
PROGRAMA Ej10a
VARIABLES
    i, m, a: ENTEROS
    t: TABLA [5] ENTEROS
COMIENZO
    PARA i←0 HASTA 4
        leer (t[i])
```

```
PROGRAMA Ej10b
VARIABLES
    n, i: ENTEROS
    a, b: TABLA [100] ENTEROS
COMIENZO
    n ← 10
```

```
        FIN PARA
        m ← 0
        PARA i ← 0 HASTA 4
            SI t[i] > m
                m ← t[i]
             FIN SI
        FIN PARA
        a ← t[4-m]
        t[4-m] ← t[m]
        t[m] ← a
        PARA i ← 0 HASTA 4
            escribir (t[i])
        FIN PARA
    FIN PROGRAMA

    Datos de entrada:
        -4, 0, 1, 3 y 2.
```

```
    PARA i←0 HASTA n-1
        leer (a[i])
    FIN PARA
    PARA i ← 0 HASTA n/2
        b[i] ← a[n-1-i]
        b[n-1-i] ← a[i]
    FIN PARA
    PARA i ← 0 HASTA n-1
        SI i mod 2 = 0
            escribir (a[i])
        SINO
            escribir (b[i])
        FIN SI
    FIN PARA
FIN PROGRAMA

Datos de entrada:
    6, 2, 8, 9, 2, 5, 8, 2, 6 y 1.
```

11-Se pretende realizar un programa para gestionar la lista de participaciones en una competición de salto de longitud. El número de plazas disponible es de 10. Sus datos se irán introduciendo en el mismo orden que vayan inscribiéndose los atletas. Diseñar el programa que muestre las siguientes opciones:

1- Inscribir un participante.
2- Mostrar listado de datos.
3- Mostrar listado por marcas.
4- Finalizar el programa.

Si se selecciona 1, se introducirán los datos de uno de los participantes: Nombre, mejor marca del 2002, mejor marca del 2001 y mejor marca del 2000.
Si se elige la opción 2, se debe mostrar un listado por número de dorsal.
La opción 3 mostrará un listado ordenado por la marca del 2002, de mayor a menor.
Tras procesar cada opción, se debe mostrar de nuevo el menú inicial, hasta que se seleccione la opción 4, que terminará el programa.

BOLETÍN 6
Funciones

1. Realizar una función, a la que se le pase como parámetro un número N, y muestre por pantalla N veces, el mensaje: "Módulo ejecutándose"

2. Diseñar una función que tenga como parámetros dos números, y que calcule el máximo.

3. Ídem una versión que calcule el máximo de 3 números.

4. Ídem una versión que calcule el máximo de una tabla de n elementos.

5. Función a la que se le pasan dos enteros y muestra todos los números comprendidos entre ellos, inclusive.

6. Función que muestra en pantalla el doble del valor que se le pasa como parámetro.

7. Realizar una función que calcule (muestre en pantalla) el área o el volumen de un cilindro, según se especifique. Para distinguir un caso de otro se le pasará el carácter 'a' (para área) o 'v' (para el volumen). Además hemos de pasarle a la función el radio y la altura.

8. Ídem que devuelva una tabla con el área y el volumen.

9. Módulo al que se le pasa un número entero y devuelve el número de divisores primos que tiene.

10. Ídem diseñar una función que devuelve una tabla con los divisores.

11. Escribir una función que calcule el máximo común divisor de dos números.

12. Ídem con tres números.

13. Ídem con una tabla.

14. Escribir una función que calcule el mínimo común múltiplo de dos números.

15. Ídem con tres números.

16. Ídem con una tabla.

17. Escriba una función que decida si dos números enteros positivos son amigos. Dos números son amigos, si la suma de sus divisores (distintos de ellos mismos) son iguales.

18. Diseña una función (en adelante **DUF**) que decida si un número es primo.

19. DUF que calcule a^n.

20. DUF que muestre en binario un número entre 0 y 255.

21. Escriba una función que sume los n primeros números impares.

22. Dado el valor de un ángulo, sería interesante saber su seno, coseno y tangente. Escribir una función que muestre en pantalla los datos anteriores.

23. Diseñar una función que calcule la distancia euclídea de dos puntos.

24. DUF a la que se le pasa como parámetro una tabla que debe rellenar. Se leerá por teclado una serie de números: guardaremos solo los pares e ignoraremos los impares. También hay que devolver la cantidad de impares ignorados.

25. DUF a la que se le pasa una tabla de enteros y un número. Debemos buscar el número en la tabla e indicar si se encuentra o no.

26. Igual que el ejercicio anterior, pero suponiendo que la tabla no está siempre llena, y el número de elementos se pasa también como parámetro.

27. Diseñar la función opera_tabla, a la que se le pasa dos tablas, el número de elementos útiles y que operación se desea realizar: sumar, restar, multiplicar o dividir (mediante un carácter: 's', 'r', 'm', 'd'). La función debe devolver una tabla con los resultados.

28. DUF que ordene la tabla que se le pasa.

29. DUF que toma como parámetros dos tablas. La primera con los 6 números de una apuesta de la primitiva, y la segunda con los 6 números ganadores. La función debe devolver el número de aciertos.

30. DUF recursiva que calcule a^n.

31. Calcular el factorial de n recursivamente.

32. DUF que calcule el valor máximo de una tabla de forma recursiva.

33. DUF que calcule el n-ésimo término de la serie de Fibonacci. En esta serie el n-ésimo valor se calcula sumando los dos valores anteriores. Es decir fibonacci(n) = fibonacci(n-1)+fibonacci(n-2), siendo fibonacci(0)=1 y fibonacci(1)=1.

34. Igual que el ejercicio anterior, pero pudiendo configurar los valores de los dos primeros término de la serie.

35. DUF que realice la búsqueda dicotómica en una tabla, de forma recursiva.

36. DUF que toma una tabla bidimensional de enteros, representando un tablero de ajedrez. Disponemos de las constantes PB (peón blanco), TN (torre negra), etc. (P, T, C, A, R, D).

Dicho módulo debe devolver un valor booleano, que indique si el rey negro está amenazado.

37. Igual que el ejercicio anterior, pero indicando si existe jaque mate a las negras.

Apéndice II
Clase Entrada

```
import java.io.*;

public class Entrada {
    static String inicializar(){
        String buzon="";
        InputStreamReader flujo=new InputStreamReader(System.in);
        BufferedReader teclado=new BufferedReader(flujo);
        try{
        buzon=teclado.readLine();
        }
        catch(Exception e){
            System.out.append("Entrada incorrecta)");
        }
        return buzon;
    }

    static int entero(){
        int valor=Integer.parseInt(inicializar());
        return valor;
    }

    static double real(){
        double valor=Double.parseDouble(inicializar());
```

```
        return valor;
    }

    static String cadena(){
        String valor=inicializar();
        return valor;
    }

    static char caracter(){
        String valor=inicializar();
        return valor.charAt(0);
    }
}
```