

# Tarea para PROG02

## Contenido

1. Ejercicio 1 .....	2
Programa .....	2
Ejemplo de ejecución.....	2
2. Ejercicio 2 .....	3
Programa .....	3
Ejemplo de ejecución.....	4
3. Ejercicio 3 .....	4
Programa .....	4
Ejemplo de ejecución.....	5
4. Ejercicio 4 .....	6
Programa .....	6
Ejemplo de ejecución.....	7
5. Ejercicio 5 .....	7
Programa .....	8
Ejemplo de ejecución.....	9
6. Ejercicio 6 .....	10
Programa .....	10
Ejemplo de ejecución.....	12
7. Ejercicio 7 .....	12
Programa .....	13
Ejemplo de ejecución.....	14
8. Ejercicio 8 .....	15
Programa .....	15
Ejemplo de ejecución.....	17
9. Ejercicio 9 .....	18
Programa .....	18
Ejemplo de ejecución.....	19
10. Ejercicio 10.....	20
Programa .....	20
Ejemplo de ejecución.....	21

## 1. Ejercicio 1

Diseña un programa Java denominado PROG01\_Ejerc1 que solicite al usuario ingresar la cantidad de kilómetros recorridos por una motocicleta y la cantidad de litros de combustible que consumió durante ese recorrido. Mostrar el consumo de combustible por kilómetro.

### Programa

```
package PROG02_Ejerc1;

import java.util.Scanner;

public class PROG02_Ejerc1 {

    public static void main(String[] args) {

        /* declaramos las variables donde guardaremos los kilometros recorridos
        y la cantidad de litros gastados*/

        double kilometros, litros;

        System.out.println("Escriba la cantidad de kilometros recorridos: "); // solicitamos al usuario el número de km

        Scanner sc = new Scanner (System.in); //creamos un nuevo scanner

        kilometros = sc.nextInt(); //asignamos a kilometros el valor introducido

        System.out.println("Escriba la cantidad de litros consumidos: "); // solicitamos al usuario los litros gastados

        litros = sc.nextInt(); //asignamos a litros el valor introducido

        double consumo = litros / kilometros; //realizamos el calculo, dividiendo litros entre kilometros

        System.out.println("El consumo de combustible por kilometro es: " + consumo); // mostramos el resultado

    }

}
```

### Ejemplo de ejecución

1. El usuario escribe los kilómetros recorridos: 100
2. El usuario escribe los litros consumidos: 40
3. Se imprime en pantalla: "El consumo de combustible por kilometro es: 0.4"

```
--- maven-compiler-plugin:3.1:compile (default-compile) @ PROG02_Ejerc1 ---
Changes detected - recompiling the module!
Compiling 1 source file to C:\Users\Usuario\Documents\NetBeansProjects\PROG02_Ejerc1\target\classes

--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc1 ---
Escriba la cantidad de kilometros recorridos:
100
Escriba la cantidad de litros consumidos:
40
El consumo de combustible por kilometro es: 0.4
-----
BUILD SUCCESS
-----
Total time: 20.042 s
Finished at: 2022-11-18T12:34:04+01:00
-----
```

## 2. Ejercicio 2

Diseña un programa Java denominado PROG02\_Ejerc2 que dada la edad de una persona, muestre un mensaje indicando si es mayor de edad, adolescente (entre 13 y 18), niño (5 y 12) o bebe (entre 0 y 4) .

### Programa

```
package PROG02_Ejerc2;
```

```
import java.util.Scanner;
```

```
public class PROG02_Ejerc2 {
    public static void main(String[] args) {
        int edad; //declaramos la variable donde almacenaremos la edad
        Scanner sc = new Scanner (System.in);
        System.out.println("Escribe tu edad: "); // pedimos al usuario que indique su
edad
        edad = sc.nextInt(); // almacenamos el dato en la variable
        //utilizaremos una anidación de condicionales para comprobar la edad de
manera excluyente
        if (edad > 18) { // comprobamos si el valor introducido es mayor a 18
            System.out.println("Eres un adulto."); //dentro de cada una de las llaves
escribimos el respectivo mensaje a imprimir
        }
    }
}
```

```
    } else if (edad >= 13 && edad <= 18){ // comprobamos si el valor es igual o
mayor a 13 e igual o menor a 18

        System.out.println("Eres un adolescente.");

    } else if (edad >= 5 && edad <= 12) { // comprobamos si el valor es igual o
mayor a 5 e igual o menor a 12

        System.out.println("Eres un niño.");

    } else if (edad >= 0 && edad <= 4) { // comprobamos si el valor es igual o
mayor a 0 e igual o menor a 4

        System.out.println("Eres un bebe, ¿qué hace un bebe usando un
ordenador?");

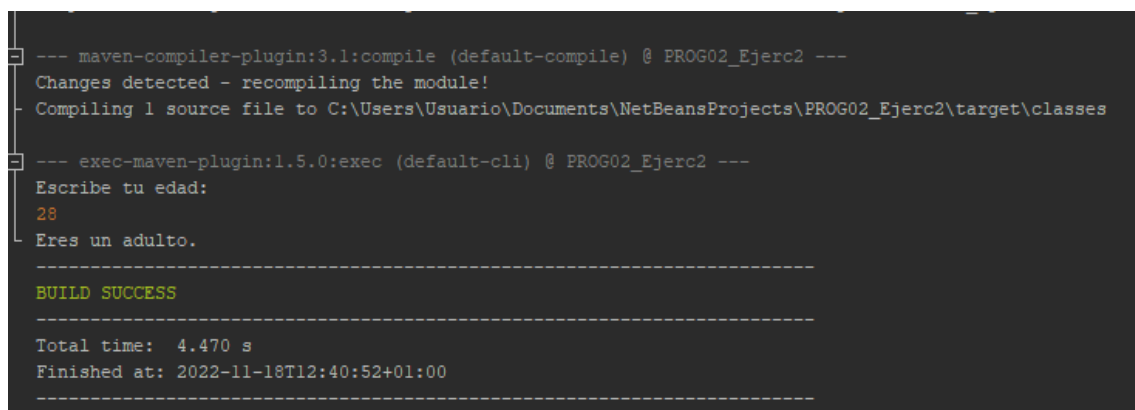
    }

}

}
```

### Ejemplo de ejecución

1. El usuario introduce su edad: 28
2. El programa devuelve un mensaje indicando la etapa de la vida en la que se encuentre un usuario. En este caso: “Eres un adulto”.



```
--- maven-compiler-plugin:3.1:compile (default-compile) @ PROG02_Ejerc2 ---
Changes detected - recompiling the module!
Compiling 1 source file to C:\Users\Usuario\Documents\NetBeansProjects\PROG02_Ejerc2\target\classes

--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc2 ---
Escribe tu edad:
28
Eres un adulto.

BUILD SUCCESS

Total time: 4.470 s
Finished at: 2022-11-18T12:40:52+01:00
```

## 3. Ejercicio 3

Diseña un programa Java denominado PROG02\_Ejerc3 que, dado un número de milisegundos, muestre en pantalla cuántos minutos, horas y días contiene.

### Programa

```
package PROG02_Ejerc3;

import java.util.Scanner;

public class PROG02_Ejerc3 {
    public static void main(String[] args) {
        int milisegundos, minutos, horas, dias; //declaramos las variables donde
        almacenaremos los datos

        Scanner sc = new Scanner (System.in);

        System.out.println("Escriba una cantidad de milisegundos"); //pedimos al
        usuario que escriba la cantidad

        milisegundos = sc.nextInt();

        /*realizamos las conversiones de tiempo*/
        minutos = milisegundos/60000;
        horas = minutos/60;
        dias = horas/24;

        /*imprimimos los resultados*/

        System.out.println("La cantidad de milisegundos contiene " + minutos + "
        minutos.");

        System.out.println(horas + " horas.");
        System.out.println("Y son " + dias + " días");
    }
}
```

### Ejemplo de ejecución

1. El programa pide al usuario una cantidad de milisegundos: Escriba una cantidad de milisegundos: 1000000000
2. El programa devuelve el resultado en minutos, horas y días:  
"La cantidad de milisegundos contiene 16666 minutos.  
277 horas.  
Y son 11 días."

```
--- maven-compiler-plugin:3.1:compile (default-compile) @ PROG02_Ejerc3 ---  
- Nothing to compile - all classes are up to date  
  
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc3 ---  
Escriba una cantidad de milisegundos  
1000000000  
La cantidad de milisegundos contiene 16666 minutos.  
277 horas.  
Y son 11 días  
  
-----  
BUILD SUCCESS  
-----  
  
Total time: 3.953 s  
Finished at: 2022-11-18T12:46:12+01:00  
-----
```

## 4. Ejercicio 4

Diseña un programa Java denominado PROG02\_Ejerc4 que cree un tipo enumerado para las siguientes razas de perro: Mastín, Terrier, Bulldog, Pekines, Caniche y Galgo. El programa debe realizar las siguientes operaciones:

- Crea una variable denominada var1 del tipo enumerador. Asígnale un valor.
- Crea una variable denominada var2 del tipo enumerador. Asígnale un valor.
- Muestra por pantalla el valor obtenido de comparar ambas variables.

Investiga sobre la posibilidad averiguar la posición que ocupa un determinado valor en el enumerado, así como mostrar la cantidad de valores que contiene. Si lo consigues, muestra la posición de las dos variables en el tipo enumerad

### Programa

```
package PROG02_Ejerc4;
```

```
public class PROG02_Ejerc4 {
```

```
    enum RazasPerro{ //creamos un tipo enumerado y escribimos las razas como  
    constantes
```

```
        MASTIN, TERRIER, BULLDOG, PEKINES, CANICHE, GALGO;
```

```
    }
```

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

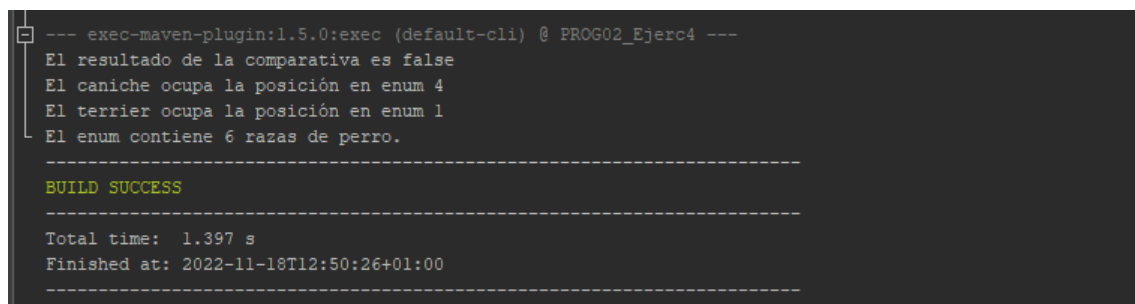
```
        //creamos una variable de tiempo enum y les asignamos un valor
```

```
RazasPerro var1 = RazasPerro.CANICHE;
RazasPerro var2 = RazasPerro.TERRIER;
//comparamos ambas variables
boolean comparativa = var1 == var2;
//imprimimos el resultado de la comparativa
System.out.println("El resultado de la comparativa es " + comparativa);
//comprobamos que posicion ocupa un valor en el enumerado
System.out.println("El caniche ocupa la posición en enum " + var1.ordinal());
System.out.println("El terrier ocupa la posición en enum " + var2.ordinal());
//mostramos la cantidad de valores que contiene enum
System.out.println("El enum contiene " + (RazasPerro.values().length) + "
razas de perro.");

}
}
```

## Ejemplo de ejecución

1. El programa muestra el resultado de la comparativa
2. El programa muestra la posición de var1
3. El programa muestra la posición de var2
4. El programa cuenta la cantidad de valores que contiene enum y lo muestra.



```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc4 ---
El resultado de la comparativa es false
El caniche ocupa la posición en enum 4
El terrier ocupa la posición en enum 1
El enum contiene 6 razas de perro.
-----
BUILD SUCCESS
-----
Total time: 1.397 s
Finished at: 2022-11-18T12:50:26+01:00
-----
```

## 5. Ejercicio 5

Escribe un programa que calcule la raíz cuadrada recogiendo los valores a,b y c por pantalla y calculando la variable x. Podeis usar la libreria Math para calcular la raíz cuadrada. Usar la función Math.sqrt()

### Programa

```
package PROG02_Ejerc5;
```

```
import java.util.Scanner;
```

```
public class PROG02_Ejerc5 {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner (System.in);  
        /*declaramos los coeficientes para a, b, c, las soluciones y el discriminante*/  
        double a, b, c, x1, x2, dis;  
        //Pedimos los coeficientes  
        System.out.println("Introduce (a)");  
        a = sc.nextDouble();  
        System.out.println("Introduce (b)");  
        b = sc.nextDouble();  
        System.out.println("Introduce (c)");  
        c = sc.nextDouble();  
        //calculamos el discriminante mediante  $b^2 - 4ac$   
        dis = (Math.pow(b, 2) - 4 * a * c);  
        //en función del resultado podemos adivinar el número de soluciones y resolver  
        if (dis < 0) { //si el discriminante es menor a 0 no tiene soluciones reales  
            System.out.println("No hay soluciones reales");  
        } else if (dis == 0){ //el discriminante es igual a cero y tiene una única solución  
            x1 = (-b + Math.sqrt(dis)) / (2*a);  
            System.out.println("La solución es: " + x1);  
        } else if (dis > 0){ // el discriminante es positivo, tenemos dos soluciones
```



```
x1 = (-b + Math.sqrt(dis)) / (2*a);  
x2 = (-b - Math.sqrt(dis)) / (2*a);  
System.out.println("Solución 1: " + x1);  
System.out.println("Solución 2: " + x2);  
  
    }  
}  
}
```

### Ejemplo de ejecución

1. Solicitamos al usuario el valor de a
2. Solicitamos al usuario el valor de b
3. Solicitamos al usuario el valor de c
4. El programa calcula el discriminante
5. Si no hay soluciones, se imprime en pantalla el mensaje: "No hay soluciones reales");
6. En caso de que haya una o dos soluciones reales, el programa muestra el resultado por pantalla.

No hay soluciones reales:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc5 ---  
Introduce (a)  
1  
Introduce (b)  
-2  
Introduce (c)  
5  
No hay soluciones reales  
-----  
BUILD SUCCESS  
-----  
Total time: 27.851 s  
Finished at: 2022-11-18T12:57:52+01:00  
-----
```

Hay una solución real:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc5 ---
Introduce (a)
1
Introduce (b)
-2
Introduce (c)
1
La solución es: 1.0
-----
BUILD SUCCESS
-----
Total time: 5.919 s
Finished at: 2022-11-18T13:04:22+01:00
-----
```

Hay dos soluciones reales:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc5 ---
Introduce (a)
2
Introduce (b)
-7
Introduce (c)
3
Solución 1: 3.0
Solución 2: 0.5
-----
BUILD SUCCESS
-----
Total time: 6.072 s
Finished at: 2022-11-18T13:11:39+01:00
-----
```

## 6. Ejercicio 6

Diseña un programa Java denominado denominado PROG02\_Ejer6 que dados el número de alumnos matriculados en Programación, número de alumnos matriculados en Entornos de Desarrollo y número de alumnos matriculados en Base de datos (recogerá el número de alumnos por pantalla). El programa deberá mostrar el % de alumnos matriculado en cada uno de los tres módulos. Se supone que un alumno sólo puede estar matriculado en un módulo. Trata de mostrar un solo decimal en los porcentajes

### Programa

```
package PROG02_Ejerc6;
```

```
import java.text.DecimalFormat;
```

```
import java.util.Scanner;
```

```
public class PROG02_Ejerc6 {  
    public static void main(String[] args) {  
        /*declaramos las variables para almacenar los alumnos matriculados*/  
        int alumProg, alumEnt, alumBases;  
  
        /*recogemos el numero de alumnos por pantalla*/  
        Scanner sc = new Scanner (System.in);  
        System.out.println("Introduce el número de alumnos matriculados en  
Programación: ");  
        alumProg = sc.nextInt();  
        System.out.println("Introduce el número de alumnos matriculados en Entornos  
de Desarrollo: ");  
        alumEnt = sc.nextInt();  
        System.out.println("Introduce el número de alumnos matriculados en Bases de  
Datos: ");  
        alumBases = sc.nextInt();  
        /*sumamos el total de los alumnos*/  
        double total = alumProg + alumEnt + alumBases;  
  
        /*calculamos el porcentaje de alumnos matriculados en cada uno de los  
modulos*/  
        double porcenProg = alumProg*100/total;  
        double porcenEnt = alumEnt*100/total;  
        double porcenBases = alumBases*100/total;  
  
        /*mostramos el porcentaje de alumnos matriculados*/  
        DecimalFormat decimales = new DecimalFormat("0.0"); //especificamos que  
queremos usar un decimal  
        System.out.println("Numero de alumnos de Entornos: " + alumEnt + "  
porcentaje: " + decimales.format(porcenEnt)+"%");  
        System.out.println("Numero de alumnos de Programación: " + alumProg + "  
porcentaje: " + decimales.format(porcenProg) + "%");  
        System.out.println("Numero de alumnos de Bases de Datos: " + alumBases + "  
porcentaje: " + decimales.format(porcenBases) + "%");  
    }  
}
```

```
}  
}
```

### Ejemplo de ejecución

1. El usuario introduce el número de alumnos matriculados en Programación: 5
2. El usuario introduce el número de alumnos matriculados en Entornos de Desarrollo: 10
3. El usuario introduce el número de alumnos matriculados en Bases de Datos: 25
4. El programa muestra el número de alumnos y los porcentajes con un decimal:

“Número de alumnos de Entornos: 10 porcentaje: 25,0%

Número de alumnos de Programación: 5 porcentaje: 12,5%

Número de alumnos de Bases de Datos: 25 porcentaje: 62,5%”

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc6 ---  
Introduce el número de alumnos matriculados en Programación:  
5  
Introduce el número de alumnos matriculados en Entornos de Desarrollo:  
10  
Introduce el número de alumnos matriculados en Bases de Datos:  
25  
Numero de alumnos de Entornos: 10 porcentaje: 25,0%  
Numero de alumnos de Programación: 5 porcentaje: 12,5%  
Numero de alumnos de Bases de Datos: 25 porcentaje: 62,5%  
-----  
BUILD SUCCESS  
-----  
Total time: 31.347 s  
Finished at: 2022-11-18T13:16:49+01:00  
-----
```

## 7. Ejercicio 7

Aunque aparentemente los años bisiestos son aquellos que son múltiplos de 4, no es del todo preciso, ya que años como 1900 y como 2100 no fueron, ni serán, bisiestos respectivamente.

Existe una explicación física que tiene que ver con que el año del calendario no coincide exactamente, en duración, con el año solar. Así, un cálculo más exacto indica que la duración real de un año es de 365,2425 días.

Para corregir este desfase, se utiliza el criterio de que se considerará año bisiesto aquel que sea divisible por 4 pero no por 100 salvo que sea divisible por 400. Por esto 1900 no es bisiesto, 1904 sí y 2000 también. Diseña un programa Java, denominado PROG02\_Ejer7, que dado un año indique si es bisiesto o no.

## Programa

```
package PROG02_Ejerc7;

import java.util.Scanner;

public class PROG02_Ejerc7 {
    public static void main(String[] args) {
        /*declaramos la variable para almacenar el año*/
        int bisiesto;

        /*pedimos un año al usuario*/
        Scanner sc = new Scanner(System.in);
        System.out.println("Introduce un año para comprobar si es bisiesto: ");
        bisiesto = sc.nextInt();

        if (bisiesto % 4 == 0 && bisiesto % 100 != 0) { //comprobamos si el año es
bisiesto
            System.out.println("El año " + bisiesto + " es bisiesto.");
        } else if (bisiesto % 4 == 0 && bisiesto % 100 == 0 && bisiesto % 400 != 0)
{ //comprobamos si el año es divisible entre 4 y 100
            System.out.println("El año " + bisiesto + " no es bisiesto.");
        } else if (bisiesto % 4 == 0 && bisiesto % 100 == 0 && bisiesto % 400 == 0)
{ //comprobamos las excepciones a la regla anterior: si un número es divisible
entre 100 y 400 es bisiesto
            System.out.println("El año " + bisiesto + " es bisiesto.");
        } else if (bisiesto % 4 != 0){ //comprobamos si el año no es bisiesto
            System.out.println("El año " + bisiesto + " no es bisiesto.");
        }
    }
}
```

```
}
```

```
}
```

## Ejemplo de ejecución

1. El programa pide al usuario un año para comprobar si es bisiesto.
2. Tras evaluar las condiciones, el programa devuelve en pantalla si el año es o no bisiesto.

El número es divisible no es divisible entre 4:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc7 ---
Introduce un año para comprobar si es bisiesto:
1994
El año 1994 no es bisiesto.
-----
BUILD SUCCESS
-----
Total time: 4.614 s
Finished at: 2022-11-18T13:25:20+01:00
-----
```

El número es divisible entre 4 y no es divisible entre 100:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc7 ---
Introduce un año para comprobar si es bisiesto:
2004
El año 2004 es bisiesto.
-----
BUILD SUCCESS
-----
Total time: 4.658 s
Finished at: 2022-11-18T13:27:28+01:00
-----
```

El número es divisible entre 4 y 100:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc7 ---
Introduce un año para comprobar si es bisiesto:
1900
El año 1900 no es bisiesto.
-----
BUILD SUCCESS
-----
Total time: 4.225 s
Finished at: 2022-11-18T13:28:05+01:00
-----
```

El número es divisible entre 4, 100 y 400:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc7 ---
Introduce un año para comprobar si es bisiesto:
2000
El año 2000 es bisiesto.
-----
BUILD SUCCESS
-----
Total time: 3.704 s
Finished at: 2022-11-18T13:28:46+01:00
-----
```

## 8. Ejercicio 8

Diseña un programa Java, denominado PROG02\_Ejerc8, que realice las siguientes operaciones, en el orden que se muestran

### Programa

```
package PROG02_Ejerc8;
```

```
public class PROG02_Ejerc8 {
    public static void main(String[] args) {
        float x = (float) 4.5;
        float y = (float) 3.0;
        int i = 2;
        System.out.println("----- Conversiones entre enteros y coma flotante -----");
        int j = (int) (i*x); // convertimos de manera explicita a int con cast
        System.out.println("Producto de int por float: j= i*x = " + j);
        double dx = 2.0;
        double dz = dx*y;
        System.out.println("Producto de float por double: dz=dx * y = " + dz);
        System.out.println("----- Operaciones con byte -----");
        byte bx = 5;
        byte by = 2;
        byte bz = (byte) (bx - by);
        System.out.println("byte: " + bx + " - " + by + " = " + bz);
        bx = -128; //al hacer este calculo sobrepasamos el rango de byte
```

```
by = 1;
bz = (byte) (bx - by); // convertimos de manera explicita a byte con cast
System.out.println("byte" + bx + " - " + by + " = " + bz);
int conversion = (bx - by); //tenemos que usar una variable intermedia int
para poder convertir desde byte
System.out.println("(int)(" + bx + " - " + by + ") = " + conversion);
System.out.println("----- Operaciones con short -----");
short sx = 5;
short sy = 2;
short sz = (short) (sx - sy);
System.out.println("short: " + sx + " - " + sy + " = " + sz);
sx = 32767;
sy = 1;
sz = (short) (sx + sy);
System.out.println("short: " + sx + " - " + sy + " = " + sz);
System.out.println("----- Operaciones con char -----");
char cx = '\u000F';
char cy = '\u0001';
int z = (char) (cx - cy);
System.out.println("char: " + cx + " - " + cy + " = " + z);
z = ((int) cx) - 1;
System.out.println("char (0x000f) -1 = " + z);
cx = '\uFFFF';
z = cx;
System.out.println("(int)( ) = " + z);
sx = (short) cx;
System.out.println("(short)( ) = " + sx);
sx = -32768;
cx = (char) sx;
z = (int) sx;
System.out.println(z + " short-char-int = " + -z);
sx = -1;
```



```
        cx = (char) sx;
        z = cx;
        System.out.println(sx + " short-char-int = " + z);

    }
}
```

### Ejemplo de ejecución

1. El programa realiza las conversiones de tipo y los cálculos y muestra los siguientes resultados:

----- Conversiones entre enteros y coma flotante -----

Producto de int por float:  $j = i * x = 9$

Producto de float por double:  $dz = dx * y = 6.0$

----- Operaciones con byte -----

byte:  $5 - 2 = 3$

byte-128 - 1 = 127

(int)(-128 - 1) = -129

----- Operaciones con short -----

short:  $5 - 2 = 3$

short:  $32767 - 1 = -32768$

----- Operaciones con char -----

char:  $- = 14$

char (0x000f) -1 = 14

(int)( ) = 65535

(short)( ) = -1

-32768 short-char-int = 32768

-1 short-char-int = 65535

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc8 ---
----- Conversiones entre enteros y coma flotante -----
Producto de int por float: j= i*x = 9
Producto de float por double: dz=dx * y = 6.0
----- Operaciones con byte -----
byte: 5 - 2 = 3
byte-128 - 1 = 127
(int)(-128 - 1) = -129
----- Operaciones con short -----
short: 5 - 2 = 3
short: 32767 - 1 = -32768
----- Operaciones con char -----
char: 14 - 14 = 14
char (0x000f) -1 = 14
(int)( ) = 65535
(short)( ) = -1
-32768 short-char-int = 32768
-1 short-char-int = 65535
-----
BUILD SUCCESS
-----
Total time: 1.394 s
Finished at: 2022-11-18T13:32:45+01:00
-----
```

## 9. Ejercicio 9

Escribir un programa que recoja un número  $n$  y escriba mientras  $n$  sea mayor o igual que cero. Llamarle PROG02\_Ejerc9

$n = 7$

vuelta 7

vuelta 5

vuelta 3

...

vuelta 1

### Programa

```
package PROG02_Ejerc9;
```

```
import java.util.Scanner;
```

```
public class PROG02_Ejerc9 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

```
System.out.println("Introduce un número entero positivo: "); //solicitamos un
numero entero positivo
```

```
int n = sc.nextInt(); //almacenamos el numero en n
```

```
if (n <= 0) { //comprobamos que el numero es positivo
```

```
System.out.println("El número entero tiene que ser positivo");
```

```
}
```

```
/*utilizamos un bucle for en el que inicializamos en base al numero
introducido,
```

```
el bucle debe continuar mientras el número sea mayor o igual a cero
```

```
y realizamos un decremento de -2*/
```

```
for (int i = n; i >= 0; i = i - 2) {
```

```
System.out.println("Vuelta: " + i);
```

```
}
```

```
}
```

```
}
```

### Ejemplo de ejecución

1. El usuario introduce un número entero positivo.
2. El programa comprueba si el número entero es negativo, si lo es devuelve un mensaje por pantalla.
3. En caso de que sea positivo, el bucle for se ejecuta, y va restando -2 al número, mientras que n es mayor o igual a 0.

El número entero es positivo:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc9 ---
Introduce un número entero positivo:
24
Vuelta: 24
Vuelta: 22
Vuelta: 20
Vuelta: 18
Vuelta: 16
Vuelta: 14
Vuelta: 12
Vuelta: 10
Vuelta: 8
Vuelta: 6
Vuelta: 4
Vuelta: 2
Vuelta: 0

-----
BUILD SUCCESS
-----

Total time:  6.088 s
Finished at: 2022-11-18T13:39:15+01:00
-----
```

El número entero es negativo:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc9 ---
Introduce un número entero positivo:
-4
El número entero tiene que ser positivo

-----
BUILD SUCCESS
-----

Total time:  2.452 s
Finished at: 2022-11-18T13:42:13+01:00
-----
```

## 10. Ejercicio 10

Crear un programa que calcule las tablas de multiplicar PROG02\_Ejerc10

Uso de un anidado para intentar mostrar la tabla de productos para los primeros diez números

### Programa

```
package PROG02_Ejerc10;
```

```
public class PROG02_Ejerc10 {  
    public static void main(String[] args) {  
        for (int tablaMul = 1; tablaMul <= 10; tablaMul++){ //utilizamos el loop externo  
            para pasar por cada una de las tablas  
                System.out.println("Tabla de multiplicar para el " + tablaMul); //imprimimos  
                la tabla en la que estamos actualmente  
                for (int i = 1; i <= 10; i++) { //utilizamos el loop interno para realizar todas  
                    las multiplicaciones de cada tabla  
                        System.out.println(tablaMul + " X " + i + " = " + tablaMul * i); //calculamos  
                        e imprimimos cada una de las multiplicaciones  
                    }  
                }  
        }  
    }  
}
```

### Ejemplo de ejecución

1. Al ejecutar el programa el loop externo inicia la primera de las tablas y muestra "Tabla de multiplicar para el: <tablaMul>" .
2. El loop interno realiza todos los cálculos y va mostrando los resultados con "1 X 1 = 1," , etc.
3. Al terminar la primera tabla se pasa a la segunda en el loop externo y se repite el proceso en el loop interno. El programa continúa hasta que llega al final de la tabla del 10.

Principio y final de la ejecución:

```
--- exec-maven-plugin:1.5.0:exec (default-cli) @ PROG02_Ejerc10 ---
Tabla de multiplicar para el 1
1 X 1 = 1
1 X 2 = 2
1 X 3 = 3
1 X 4 = 4
1 X 5 = 5
1 X 6 = 6
1 X 7 = 7
1 X 8 = 8
1 X 9 = 9
1 X 10 = 10
Tabla de multiplicar para el 2
2 X 1 = 2
2 X 2 = 4
2 X 3 = 6
2 X 4 = 8
2 X 5 = 10
2 X 6 = 12
2 X 7 = 14
2 X 8 = 16
2 X 9 = 18
2 X 10 = 20
Tabla de multiplicar para el 3
3 X 1 = 3
3 X 2 = 6
3 X 3 = 9
3 X 4 = 12
3 X 5 = 15
3 X 6 = 18
3 X 7 = 21
3 X 8 = 24
3 X 9 = 27
3 X 10 = 30
Tabla de multiplicar para el 4
4 X 1 = 4
4 X 2 = 8
4 X 3 = 12
4 X 4 = 16
4 X 5 = 20
4 X 6 = 24
4 X 7 = 28
4 X 8 = 32
```

```
9 X 10 = 90
Tabla de multiplicar para el 10
10 X 1 = 10
10 X 2 = 20
10 X 3 = 30
10 X 4 = 40
10 X 5 = 50
10 X 6 = 60
10 X 7 = 70
10 X 8 = 80
10 X 9 = 90
10 X 10 = 100
```

-----  
**BUILD SUCCESS**  
-----

Total time: 1.220 s  
Finished at: 2022-11-18T13:45:34+01:00  
-----