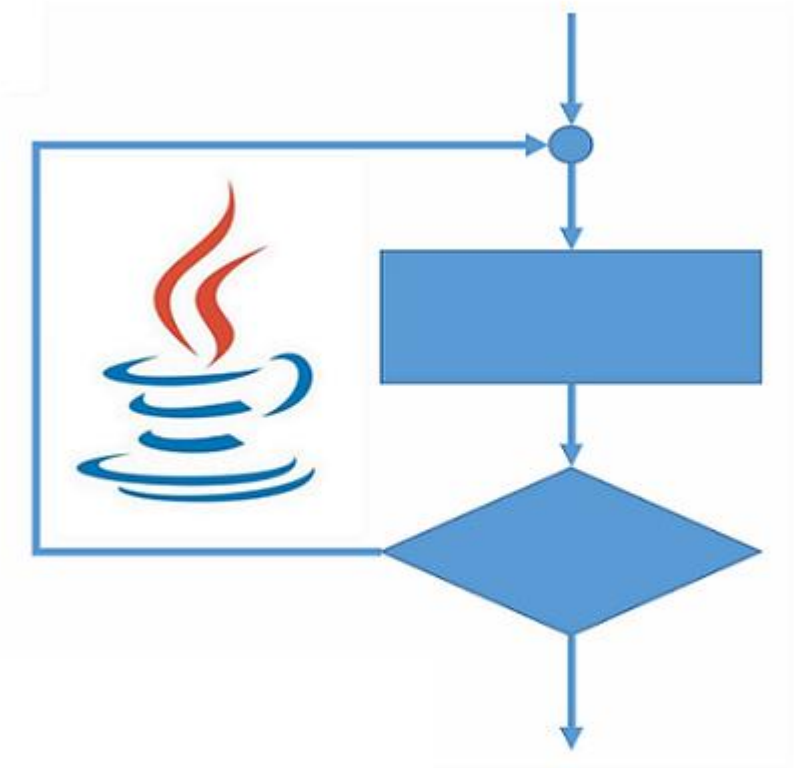
CEPSUNI
www.ceps.uni.edu.pe

FUNDAMENTOS DE PROGRAMACIÓN CON JAVA



UNIDAD 03
ENTRADA Y SALIDA DE DATOS

Eric Gustavo Coronel Castillo
youtube.com/DesarrollaSoftware
gcoronel@uni.edu.pe

INDICE

LECTURA DE DATOS

Este es probablemente el método más preferido para tomar datos de entrada. El objetivo principal de la clase **Scanner** es analizar los tipos primitivos y las cadenas con expresiones regulares, sin embargo, también se puede usar para leer las entradas del usuario en la línea de comandos.

Ejemplo 1

```
// Demostración del funcionamiento de la clase Scanner
import java.util.Scanner;

public class Ejemplo01
{
    public static void main(String args[])
    {
        // Usando Scanner para obtener información del usuario
        Scanner in = new Scanner(System.in);

        String s = in.nextLine();
        System.out.println("Usted ingresó la cadena: "+s);

        int a = in.nextInt();
        System.out.println("Usted ingresó un entero: "+a);

        float b = in.nextFloat();
        System.out.println("Usted ingresó un float: "+b);
    }
}
```

SALIDA DE DATOS

Para poder imprimir en la consola debemos utilizar System.out que pertenece a la clase java.io.PrintStream, aquí tenemos implementados los siguientes métodos:

- print()
- println()
- printf()

Método: print()

Sintaxis

```
System.out.print( dato );
```

Se trata de un método sobrecargado, esto quiere decir que existe una implementación para cada tipo de dato, pero de eso no nos debemos preocupar, simplemente como dato ponemos una expresión de cualquier tipo y listo, el lenguaje se encarga de utilizar el método correcto, veamos algunos ejemplos.

```
int nota = 15;
double igv = 0.19;
System.out.print(nota);
System.out.print("\t");
System.out.print(igv);
```

En este ejemplo estamos usando print para imprimir tres datos de tipos diferentes, primero un int, luego un String, y finalmente un double, el resultado sería el siguiente:

```
15   0.19
```

La característica de este método es que no realiza un cambio de línea.

Método: println()

Sintaxis

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

Al igual que el método print, este también se encuentra sobrecargado, existe una implementación para cada tipo de dato, donde dato puede ser una expresión de cualquier tipo, el lenguaje se encarga de utilizar el método correcto, veamos algunos ejemplos.

```
int nota = 15;
double igv = 0.19;
String msg = "El verdadero discípulo es el que supera al maestro";
System.out.println(nota);
System.out.println(msg);
System.out.println(igv);
```

En este ejemplo estamos usando println para imprimir tres datos de tipos diferentes, primero un int, luego un String, y finalmente un double, el resultado sería el siguiente:

```
15
El verdadero discípulo es el que supera al maestro
0.19
```

La característica de este método es que realiza un cambio de línea.

Método: printf()

Sintaxis

```
System.out.printf( cadena_formato, Lista_parametros );
```

Este método puede tener varios parámetros, pero el primero de ellos es una cadena con formato, y los demás parámetros dependen de los parámetros que tiene la cadena con formato.

La sintaxis general para indicar los formatos es el siguiente:

```
%[nro_parametro$][flags][ancho][.precisión]sinbolo_conversión
```

Las opciones que están entre corchetes son opcionales y se utilizan según el tipo de dato y necesidades que impresión que se tengan.

La siguiente tabla muestra los diferentes caracteres que pueden ser usados como flag, que deben ser utilizados dependiendo tipo del dato que se quiere imprimir.

FLAG	DESCRIPCIÓN
'-'	El resultado se ajustará a la izquierda.
'#'	El resultado debe usar una forma de conversión alterna.
'+'	El resultado incluirá el signo.
' '	El resultado incluirá un espacio para números positivos.
'0'	El resultado será rellenado con ceros.
','	El resultado incluirá separadores.
'('	Los números negativos serán encerrados entre paréntesis.

Veamos el siguiente ejemplo:

```
float precio = 18.567f;
System.out.printf("Precio: %1$f   %1$010.2f", precio);
```

Se está imprimiendo el precio dos veces, en la segunda se está indicando que se rellene con ceros, un ancho de 10, y decimales, el resultado es el siguiente:

```
Precio: 18,566999        0000018,57
```

En la siguiente tabla se muestra los diferentes símbolos que podemos utilizar para dar formatos.

SÍMBOLO	CATEGORÍA	SIGNIFICADO
b B	General	Si el argumento es null imprime false. Si el argumento es boolean imprime false ó true. para cualquier otro caso imprime true.
h H	General	Si el argumento es null imprime null. Cualquier otro caso el valor hexadecimal del argumento.
s S	General	Si el argumento es null imprime null. Cualquier otro caso el el argumento convertido a String.
c C	Caracter	El argumento se trata como número entero y se muestra el caracter cuyo código ASCII se corresponde con el valor.
d	Entero	El argumento se trata como número entero y se imprime su valor decimal.
o	Entero	El argumento se trata como un número entero y se imprime su valor en octal.
x X	Entero	El argumento se trata como un número entero y se imprime su valor en hexadecimal.
e E	Punto flotante	El argumento se trata como un número de tipo double y se imprime en notación científica.
f	Punto flotante	El argumento se trata como un número de tipo double y se imprime como un decimal.
g G	Punto flotante	El argumento se imprime como un decimal ó notación científica, dependiendo de la precisión y el valor después del redondeo.
a A	Punto flotante	El argumento se ajusta a un formato de punto flotante hexadecimal.
t T	Fecha/Hora	Este prefijo se utiliza para conversiones de fecha y hora.

SÍMBOLO	CATEGORÍA	SIGNIFICADO
%	Porcentaje	Representa el símbolo del porcentaje
n	letra n	Representa un salto de línea.

Veamos el siguiente ejemplo:

```
float precio = 18.567f;
System.out.printf("Precio: %1$e   %1$.2f", precio);
```

Se está imprimiendo el mismo número dos veces, la primera vez se imprime en notación científica y la segunda vez se especifica con dos decimales, el resultado es el siguiente:

```
Precio: 1.856700e+01   18,57
```

El siguiente ejemplo ilustra el uso de formatos de fecha:

```
import java.util.Date;
public class test
{
  public static void main(String[] args)
  {
    Date fecha = new Date();
    System.out.printf("Fecha de hoy: %1$tY-%1$tm-%1$td", fecha);
  }
}
```

El resultado es el siguiente:

```
Fecha de hoy: 2005-12-18
```

Note usted que se está imprimiendo el mismo argumento tres veces, la primera vez se imprime el año, la segunda vez el mes, y finalmente el día.

CURSOS RELACIONADOS

https://www.ceps.uni.edu.pe/


SQL SERVER
INTELIGENCIA DE
NEGOCIOS
NIVEL IV


Fundamentos
de Programación


PROGRAMACIÓN
ORIENTADA A OBJETOS
CON JAVA


CLIENTE - SERVIDOR


DESARROLLO WEB


FACEBOOK
PARA NEGOCIO