

**Aprender y practicar**  
lenguaje de  
programación  
**Java**

José Rafael Baca Pumarejo  
Vicente Villanueva Hernández



**Aprender y practicar**  
lenguaje de  
programación  
**Java**

---

Aprender y practicar lenguaje de programación Java/José Rafael Baca Pumarejo, Vicente Villanueva Hernández .—Cd. Victoria, Tamaulipas : Universidad Autónoma de Tamaulipas ; 2023. 171 págs. ; 17 x 23 cm.

l. 005 - Programación, programas, datos de computadores

LC: LC QA76.73.J38 B3.3 2023

DEWEY: UMX Lenguajes de programación y extensión / “scripting”: generalidades

---

Universidad Autónoma de Tamaulipas  
Matamoros SN, Zona Centro  
Ciudad Victoria, Tamaulipas C.P. 87000  
D. R. © 2023

Consejo de Publicaciones UAT  
Centro Universitario Victoria  
Centro de Gestión del Conocimiento. Segundo Piso  
Ciudad Victoria, Tamaulipas, México. C.P. 87149  
Tel. (52) 834 3181-800 • extensión: 2905 • [www.uat.edu.mx](http://www.uat.edu.mx)  
[consejopublicacionesuat@outlook.com](mailto:consejopublicacionesuat@outlook.com)

Libro aprobado por el Consejo de Publicaciones UAT  
ISBN UAT: 978-607-8888-22-1

Se prohíbe la reproducción total o parcial de esta obra incluido el diseño tipográfico y de portada, sea cual fuera el medio, electrónico o mecánico, sin el consentimiento del Consejo de Publicaciones UAT.  
Libro digital

**Este manual fue evaluado y aprobado por el Consejo de Publicaciones de la UAT para el uso del sello editorial de la Universidad Autónoma de Tamaulipas. Su contenido fue sometido al análisis de un software antiplagio para garantizar su originalidad e integridad.**

**Aprender y practicar**  
lenguaje de  
programación  
**Java**

José Rafael Baca Pumarejo  
Vicente Villanueva Hernández



C.P. Guillermo Mendoza Cavazos  
PRESIDENTE

Dra. Mariana Zerón Félix  
VICEPRESIDENTE

Dr. Leonardo Uriel Arellano Méndez  
SECRETARIO TÉCNICO

Mtro. Franklin Huerta Castro  
VOCAL

Dra. Rosa Issel Acosta González  
VOCAL

Mtro. Rafael Pichardo Torres  
VOCAL

Mtro. Mauricio Pimentel Torres  
VOCAL

**Consejo Editorial del Consejo de Publicaciones de la Universidad Autónoma de Tamaulipas**

**Dra. Lourdes Arizpe Slogher** • Universidad Nacional Autónoma de México | **Dr. Amalio Blanco** • Universidad Autónoma de Madrid, España | **Dra. Rosalba Casas Guerrero** • Universidad Nacional Autónoma de México | **Dr. Francisco Díaz Bretones** • Universidad de Granada, España | **Dr. Rolando Díaz Lowing** • Universidad Nacional Autónoma de México | **Dr. Manuel Fernández Ríos** • Universidad Autónoma de Madrid, España | **Dr. Manuel Fernández Navarro** • Universidad Autónoma Metropolitana, México | **Dra. Juana Juárez Romero** • Universidad Autónoma Metropolitana, México | **Dr. Manuel Marín Sánchez** • Universidad de Sevilla, España | **Dr. Cervando Martínez** • University of Texas at San Antonio, E.U.A. | **Dr. Darío Páez** • Universidad del País Vasco, España | **Dra. María Cristina Puga Espinosa** • Universidad Nacional Autónoma de México | **Dr. Luis Arturo Rivas Tovar** • Instituto Politécnico Nacional, México | **Dr. Aroldo Rodrigues** • University of California at Fresno, E.U.A. | **Dr. José Manuel Valenzuela Arce** • Colegio de la Frontera Norte, México | **Dra. Margarita Velázquez Gutiérrez** • Universidad Nacional Autónoma de México | **Dr. José Manuel Sabucedo Cameselle** • Universidad de Santiago de Compostela, España | **Dr. Alessandro Soares da Silva** • Universidad de São Paulo, Brasil | **Dr. Akexandre Dorna** • Universidad de CAEN, Francia | **Dr. Ismael Vidales Delgado** • Universidad Regiomontana, México | **Dr. José Francisco Zúñiga García** • Universidad de Granada, España | **Dr. Bernardo Jiménez** • Universidad de Guadalajara, México | **Dr. Juan Enrique Marciano Medina** • Universidad de Puerto Rico-Humacao | **Dra. Ursula Oswald** • Universidad Nacional Autónoma de México | **Arq. Carlos Mario Yori** • Universidad Nacional de Colombia | **Arq. Walter Debenedetti** • Universidad de Patrimonio, Colonia, Uruguay | **Dr. Andrés Piqueras** • Universitat Jaume I, Valencia, España | **Dra. Yolanda Troyano Rodríguez** • Universidad de Sevilla, España | **Dra. María Lucero Guzmán Jiménez** • Universidad Nacional Autónoma de México | **Dra. Patricia González Aldea** • Universidad Carlos III de Madrid, España | **Dr. Marcelo Urra** • Revista Latinoamericana de Psicología Social | **Dr. Rubén Ardila** • Universidad Nacional de Colombia | **Dr. Jorge Gissi** • Pontificia Universidad Católica de Chile | **Dr. Julio F. Villegas †** • Universidad Diego Portales, Chile | **Ángel Bonifaz Ezeta †** • Universidad Nacional Autónoma de México

# Índice

<b>Introducción</b>	11
<b>Capítulo 1</b>	
<b>Aplicaciones Java</b>	17
1.1. Objetivo del capítulo	17
1.2. Introducción	17
1.3. Pasos para descargar	17
1.4. Comandos <i>print</i> , <i>println</i> o <i>printf</i>	18
1.5. Ejemplos prácticos realizados utilizando los comandos anteriores	18
1.6. Ejercicios para resolver. Casos prácticos	38
1.7. Referencias de consulta	39
1.8. Videos de apoyo	39
<b>Capítulo 2</b>	
<b>Estructuras de control <i>if</i></b>	41
2.1. Objetivo del capítulo	41
2.2. Introducción	41
2.3. Ejemplos prácticos realizados con el comando <i>if</i>	41
2.4. Ejercicios para resolver. Casos prácticos	69
2.5. Referencias de consulta	71
2.6. Videos de consulta	71
<b>Capítulo 3</b>	
<b>Ciclos <i>for</i></b>	73
3.1. Objetivo del capítulo	73
3.2. Introducción	73
3.3. Ejemplos prácticos realizados con el comando <i>for</i>	73
3.4. Ejercicios para resolver. Casos prácticos	88
3.5. Referencias de consulta	98
3.6. Videos de consulta del ciclo <i>for</i> en Java	98

## Capítulo 4

Ciclos <i>do-while</i>	91
4.1. Objetivo del capítulo	91
4.2. Introducción	91
4.3. Ejemplos prácticos realizados, utilizando el comando de repetición <i>do-while</i>	91
4.4. Ejercicios para resolver. Casos prácticos	97
4.5. Referencias de consulta	98
4.6. Videos de apoyo	98

## Capítulo 5

Ciclos <i>while</i>	99
5.1. Objetivo del capítulo	99
5.2. Introducción	99
5.3. Ejemplos prácticos realizados utilizando ciclo <i>while</i>	99
5.4. Ejercicios para resolver utilizando ciclo <i>while</i> . Casos prácticos	106
5.5. Referencias de consulta	107
5.6. Video de apoyo	107

## Capítulo 6

### Instrucción Selección Múltiple *Switch*

6.1. Objetivo del capítulo	109
6.2. Introducción	109
6.3. Ejemplos prácticos realizados utilizando ciclo <i>switch</i>	109
6.4. Ejercicios para resolver. Casos prácticos	118
6.5. Referencias de consulta	120
6.6. Videos de apoyo	120

## Capítulo 7

Estructuras de control anidadas	121
7.1. Objetivo del capítulo	121
7.2. Introducción	121
7.3. Ejemplos prácticos realizados utilizando comandos anteriores	121
7.4. Ejercicios para resolver. Casos prácticos	138
7.5. Referencias de consulta	138
7.6. Videos de apoyo	139



## **Capítulo 8**

Arreglos unidimensionales	141
8.1. Objetivo del capítulo	141
8.2. Introducción	141
8.3. Ejemplos prácticos utilizando arreglos de una sola dimensión	141
8.4. Resolución de casos prácticos	151
8.5. Referencias de consulta	152
8.6. Video de apoyo	152

## **Capítulo 9**

Arreglos multidimensionales	153
9.1. Objetivos del capítulo	153
9.2. Introducción	153
9.3. Ejemplos prácticos realizados utilizando arreglos multidimensionales	153
9.4. Ejercicios para resolver. Casos prácticos	167
9.5. Referencias de consulta	168
9.6. Video de apoyo	168

Referencias	169
-------------	-----

<b>De los autores</b>	171
-----------------------	-----



# Introducción

---

La presente obra tiene por objetivo que el estudiante de informática de la Universidad Autónoma de Tamaulipas adquiera conocimientos actualizados de programación orientada a objetos, para reforzar las habilidades de análisis y síntesis en la resolución de problemas relacionados con su profesión, mediante una actitud innovadora, creativa y ética que le permita desarrollar un liderazgo positivo.

Este libro de enfoque práctico para uso académico recopila, describe, utiliza y desarrolla algoritmos a partir de los códigos pertenecientes al lenguaje Java, abona de manera directa al perfil de egreso de un estudiante de informática, porque aporta elementos que contribuyen a que el futuro profesionista de esta área adquiera competencias y conocimientos especializados en las tecnologías de información y comunicación, para analizar, desarrollar e integrar soluciones innovadoras para la transformación de los procesos de cualquier organización.

En este contexto se busca fortalecer al capital humano en formación de los programas de licenciatura o ingeniería en las tecnologías de la información y comunicación ofrecidos por diversas escuelas y facultades de la Universidad Autónoma de Tamaulipas. El tratamiento y análisis de los estatutos para la programación son explicados con detalle. Se busca apoyar el entendimiento y asimilación de los algoritmos descritos y probados para ofrecer a los alumnos ejemplos de los diversos tipos de instrucciones en Java. Así estos elementos didácticos fortalecen la labor de programación de los estudiantes para la resolución de problemas de la vida real. Inducir mediante la práctica la programación de aplicaciones usando el lenguaje Java, entrena al alumno en el análisis y la programación necesarias para la solución de los problemas en organizaciones del sector productivo. Este libro teórico-práctico incide en el Programa Nacional Estratégico de Educación (Pronaces-Educación) y representa un factor innovador porque elevará la calidad de la educación y entrenamiento en esta área de la informática tan importante para la competitividad de nuestros estudiantes. La obra enfatiza la reflexión sobre la práctica al abordar la actividad de programación de los alumnos encaminada al abordaje de la tecnología representada por este superlenguaje computacional. Esto le permitirá diversificar sus aplicaciones en el ámbito profesional con un criterio analítico reforzado.

[...] el programa de investigación e incidencia a mediano y largo plazo tiene como objetivo fomentar la inclusión social y la autonomía de las personas

mediante procesos de fortalecimiento de la lectoescritura, la educación cívica para la paz, los lenguajes formales, las vocaciones científicas, la educación indígena y la evaluación educativa horizontal (Conahcyt, 2023).

El libro contribuye a la autonomía de los futuros profesionistas apropiándose de un conocimiento efectivo para desarrollar programas y aplicaciones en Java para su desempeño profesional; contribuye también a desarrollar vocaciones científicas y a asegurar una evaluación positiva de la educación horizontal del educando con equidad y calidad.

## Estructuración de los conceptos ilustrados

Los capítulos de este libro cubren este propósito mediante el desarrollo de los principales conceptos asociados a los estatutos más usados del lenguaje Java, explicado mediante ejemplos prácticos que constituyen los laboratorios a experimentar por el alumno para comprobar en su computadora el funcionamiento de los estatutos revisados, de esta manera se coadyuva a que los alumnos asimilen los principales conceptos acerca de este lenguaje y desarrollen aplicaciones en las organizaciones con el conocimiento adecuado. Los capítulos abordan el estudio de los estatutos mencionados de manera independiente, sin embargo, su ordenamiento va de las instrucciones más sencillas hasta las más sofisticadas. Esta distribución permite a docentes y estudiantes examinar los capítulos o detalles de estatutos según su interés y propósito de tal forma que el manual pueda ser incluido en cursos de manera básica o complementaria, para fortalecer el aprendizaje de los alumnos o interesados:

## Capítulo 1. Aplicaciones Java

En este primer capítulo del libro se revisan los conceptos acerca de la programación orientada a objetos utilizando la metodología de solución de problemas mediante computadora, para ello se aborda el aprendizaje básico de lenguaje de programación Java, se estudian los estatutos de Entrada/Salida para imprimir o desplegar en pantalla, así como aquellos correspondientes a las operaciones de matemáticas básicas como suma, resta, división, multiplicación, los operadores de comparación y los estatutos *set* y *get*.

El conocimiento de Java como lenguaje se presenta como la solución de problemas del mundo real, mediante el diseño de una solución luego codificada en este lenguaje de programación orientado a objetos. En este objetivo se fomenta la construcción de programas a través de la implementación de los algoritmos y que el alumno desarrolle soluciones para problemas reales con esta herramienta.

## Capítulo 2. Estructuras de control *if*

En este capítulo se analiza el estatuto de control *if* utilizado para bifurcar el flujo de seguimiento de instrucciones de un programa codificado en Java, la sintaxis funciona con base en una condición después de la palabra *if*. La condición es una pregunta y debe estar rodeada por paréntesis. Para formar una pregunta, puede usar símbolos relacionales u otros operadores (Dean, 2019). Esta sección tiene como objetivo que el alumno empiece a manejar las estructuras condicionales de control a partir de una condición simple o con varias alternativas de solución (anidadas).

## Capítulo 3. Ciclos *for*

En esta sección se analiza el estatuto de control *for* que permite el ciclo repetitivo de un rango de estatutos del lenguaje que se repetirán hasta un límite predefinido por una condición, la sintaxis funciona con base en una condición que define el propio estatuto *for*; esta condición se define con base en una variable contador *i* que generalmente se inicializa en 0 y aumenta cada vez que recorre el ciclo del rango de instrucciones (Dean, 2019). Con un bucle *for*, el mecanismo de contador es una parte fundamental del bucle, dicha variable de contador recibe un nombre especial: variable de índice. Las partes esenciales del estatuto *for* son tres componentes:

- Componente de inicialización
- Componente de condición
- Actualizar componente

El objetivo de esta sección dedicada al estatuto *for* es que el alumno aprenda a ejecutar e iterar con el comando las instrucciones contenidas en un rango y practique el control sobre el ciclo, mediante una variable de control (Dean, 2019). Por lo que se espera que el alumno aprenda a identificar y utilizar una técnica para la solución de un problema empleando esta estructura de control repetitiva. El estudiante deberá de aprender que esta instrucción sirve para implementar un ciclo en un conjunto de instrucciones o rango de código definido, mediante una variable de control a través de un valor de inicio, un tamaño de incremento y un valor límite para el ciclo (González, 2018).

## Capítulo 4. Ciclos *do-while*

En este capítulo se analiza el estatuto de control denominado *do-while*. El objetivo es que el alumno aprenda formas alternativas de aplicar estructuras repetitivas básicas con el ciclo *do-while*. El estatuto es analizado en su sintaxis y sus componentes para usarlo en el control de la ejecución secuencial de un conjunto de instrucciones contenidas en un rango mediante un rango de instrucciones controladas por una

variable índice a través de un incremento o decremento en su valor inicial (Dean, 2019). *Do-while*, es una estructura repetitiva, para ejecutar al menos una vez su bloque de instrucciones, en contraste con el *while* o del *for* que pueden no ejecutar el bloque de instrucciones porque la condición está satisfecha al principio. Esta estructura repetitiva se utiliza cuando conocemos de antemano que por lo menos una vez entra al cuerpo del ciclo a ejecutar los estatutos que están dentro del ciclo (Dean, 2019). De la definición anterior se deduce:

[...] se crea un ciclo que ejecuta una sentencia especificada, hasta que la condición de comprobación se evalúa como falsa. La condición se evalúa después de ejecutar el estatuto, dando como resultado que la instrucción especificada se ejecute al menos una vez.

## Capítulo 5. Ciclos *while*

En este capítulo se analiza el estatuto de control *while* para que el alumno comprenda otra alternativa de establecer rutinas repetitivas, se describe la sintaxis del estatuto y sus componentes para que lleve a cabo sus funciones. Se resalta y se hacen prácticas para comparar la ventaja que representa en su implementación no tener la necesidad de establecer un valor inicial numérico para su variable índice además de que puede manejar diferentes tipos de variables tales como booleanas o de otro tipo según se requiera para implementar este bucle. En este sentido se analiza que *while* crea un ciclo que ejecuta una instrucción o varias mientras se ejecuta la condición, si la condición es verdadera, se ejecuta, pero si la condición es evaluada como falsa se sale del ciclo (Dean, 2019).

## Capítulo 6. Instrucción de Selección Múltiple *Switch*

Este apartado de la obra tiene como propósito que el alumno aprenda a manejar y aplicar esta instrucción de Java que se usa para ejecutar un control múltiple a través de su capacidad selectora de diversas rutas en el código del programa según los valores contenidos en la variable de control de tal forma que mediante el poder de esta estructura de control condicional un grupo particular de sentencias entre varias alternativas posibles son ejecutadas para proseguir con el resto de los códigos y lograr los resultados esperados (Dean, 2019). Para ello los ejemplos expuestos al estudiante lo ilustran sobre la instrucción *switch* mediante la variable recibida como argumento y la obtención del valor de dicha variable.

Después lo compara con los valores asociados con las etiquetas *case*, comenzando a ejecutar el código que hay debajo de la etiqueta cuando el valor coincidió. Al encontrar la instrucción *break* finalizará la ejecución del *switch*. En caso de no encontrar una etiqueta *case* coincidente, ejecutará el código que existe dentro de

la etiqueta *default*. La etiqueta *default* es opcional, por tanto, en caso de no tenerla simplemente no hará nada.

## Capítulo 7. Estructuras de control anidadas

Este capítulo tiene como propósito ilustrar al estudiante sobre estas técnicas cuando las necesidades de resolución de problemas planteados implican que en los códigos de los programas se aniden diferentes estructuras de control de *loops*, para ello el alumno conocerá cómo se logra anidar estas rutinas cíclicas. Además, conocerá que tanto la estructura de control interior y exterior pueden manejar diferentes tipos de datos. En este propósito se analizan programas con *if-else* anidados, por ejemplo, de igual manera un ciclo dentro de otro ciclo o diferentes casos de estructuras anidadas. Adicionalmente, se abordan las estructuras *while*, *for*, *do while*. Estas estructuras pueden ir una dentro de la otra con distinta variable de control (Dean, 2019). El alumno practicará con estructuras anidadas utilizando diferentes combinaciones de estructuras cíclicas, conteniendo otra estructura.

## Capítulo 8. Arreglos unidimensionales

En este apartado de la obra se estudian las estructuras de datos esenciales en la programación de códigos en el lenguaje Java. El propósito de esta sección es lograr que el alumno maneje las estructuras de datos llamadas arreglos unidimensionales para introducir datos en una variable de arreglo de una dimensión y refrendará la importancia de las variables índices, porque ellas tendrán la función de apuntadores o indicadores respecto a los datos almacenados en la estructura de estos arreglos unidimensionales.

## Capítulo 9. Arreglos multidimensionales

En este capítulo se aprenderá a manejar estructuras que a diferencia de los arreglos unidimensionales permiten almacenar datos caracterizados en arreglos más allá de la dimensión lineal, para avanzar a los arreglos bidimensionales o matrices, así como los arreglos de tres dimensiones o cúbicos. Para ello se tiene como objetivo que el estudiante aprenda a desarrollar diferentes aplicaciones que aborden problemas relacionados con inventarios, ventas generales, detectar qué día se vende más, que producto se vendió más y en qué día mediante códigos de programas con el uso de matrices y subíndices para acceder a los datos almacenados y generar información útil a la organización.





# Capítulo 1

## Aplicaciones Java

### 1.1. Objetivo del capítulo

Reconoce los conceptos básicos sobre la programación orientada a objetos utilizando la metodología de solución de problemas a través de una computadora

### 1.2. Introducción

En este capítulo se aprende lo básico del lenguaje de programación Java, desde cómo se imprime un valor hasta las operaciones de matemáticas básicas de cómo se realiza una suma, resta, división, multiplicación y otras operaciones. Así, como también como se utilizan los operadores de igualdad y métodos *set* y *get*.

Se pretende abordar la solución de un problema del mundo real mediante el diseño de una solución codificada en un lenguaje de programación orientado a objetos. Para ello se usarán bloques básicos que permitan la construcción de programas a través de la implementación de los algoritmos.

### 1.3. Pasos para descargar

Cuando un alumno va a capturar un programa, antes debe descargar el programa mediador (NETBEANS 8.0, ECLIPSE, JBUILDER, JCREATOR, BLUE, etcétera), que permiten editar o correr un programa. Cuando ya se tenga instalado el *software* elegido, por ejemplo el NETBEANS, automáticamente va a mostrar el editor de textos donde podrá editar sus programas, en la parte superior se muestra un menú, donde vienen *file*, *edit*, *compile*, etcétera. Al presionar *file* va a preguntar el nombre del proyecto y despliega una segunda pantalla con el editor de texto donde se captura el programa. Si el nombre del proyecto fuera *programa1* y se presiona el botón *finish*, automáticamente se generará el siguiente código:

```
Public class programa1
{
    Public void main (string[args])
    {
    }
}
```

Todo programa se empieza a programar a partir del segundo *public* entre las dos llaves, ese va a ser el programa principal.

#### 1.4. Comandos *print*, *println* o *printf*

Este comando despliega el valor que contiene una variable o una constante en la pantalla. El comando *print* maneja tres formatos:

*Print*, *println* y *printf* pero para que esto funcione necesitamos anteponer *System.out.print* o *println* o *printf* con `()` y con comillas para los mensajes de texto y sin comillas para los valores de cualquier tipo de datos, ya sean numéricos o alfanuméricos.

#### 1.5. Ejemplos prácticos realizados utilizando los comandos anteriores

##### Laboratorio 1. Desarrollo de programa para imprimir en una línea

Diseñar y desarrollar un programa que imprima en una misma línea “Bienvenido a la Programación de Java” y que luego imprima cada una de las palabras de esa frase en líneas separadas.

**Tabla 1**

*Variables utilizadas en el Laboratorio 1*

Nombre	Cadena de caracteres	Propósito
<i>System.out</i>	Permite a las aplicaciones de java mostrar un <i>string</i> de caracteres en la ventana de comandos	
<i>System.out, print</i>	Esta frase puede mostrar varios mensajes de cadena	Visualiza o imprime un mensaje de texto en la pantalla
<i>Secuencia de escape</i>	<code>\n</code>	Mueve el cursor a la siguiente línea
<i>Diagonales</i>	<code>//</code>	Se utiliza para comentarios de fin de línea

Fuente: elaboración propia de acuerdo con Deitel (2008).

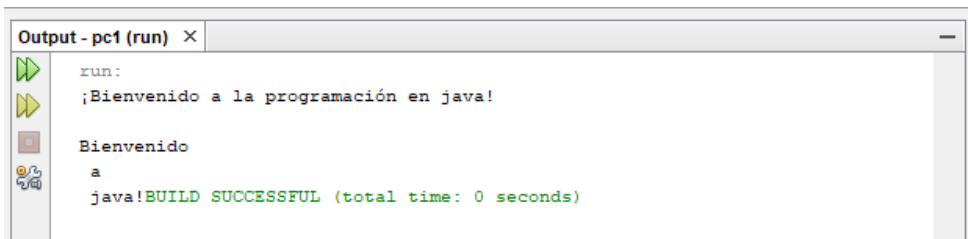
## Codificación

```
public class Laboratorio1 {  
    /**  
     * @param args the command line arguments  
     */  
    //la rutina main inicia la ejecución de la aplicación en java  
    public static void main (String [] args) {  
        // TODO code application logic here  
        System.out.print(¡" Bienvenido a la programación en java! \n\n");  
        System.out.print("Bienvenido\n a\n java!");  
    } //fin de la rutina main  
} //fin de la clase práctica#1
```

## Ejecución

### Figura 1

*Ejecución del Laboratorio 1*



## Laboratorio 2. Desarrollo de programa para suma e impresión

Diseñar y desarrollar un programa que sume dos números e imprimir dicha suma.

### Tabla 2

*Variables utilizadas en el Laboratorio 2*

Nombre de las variables	Tipo de variable	Propósito
<i>import</i>	Clase predefinida	Auxilia al compilador a detectar una clase predefinida que usa el lenguaje
<i>num1</i>	<i>Int</i> (entera)	Obtiene un número desde el teclado
<i>num2</i>	<i>Int</i> (entera)	Recibe un número desde el teclado
<i>Sum2</i>	<i>Int</i> (entera)	Almacena el resultado de la sumatoria

Fuente: elaboración propia basada en Deitel (2008).

## Codificación

```
import java. util. Scanner;
public class Laboratorio2
{
    public static void main (String [] args)
    {
        \\ Es una instrucción de declaración de una variable. Y una dirección de memoria
        donde se guarda el dato. La variable es de tipo scanner que puede recibirse
        mediante el teclado (Deitel, 2008)

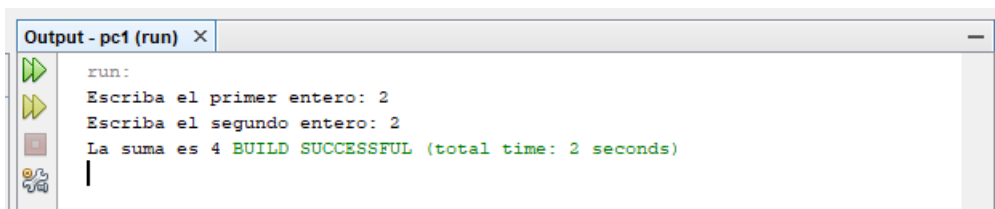
        Scanner entrada = new Scanner (System.in);
        int num1;
        int num2;
        int Sum2;
        System.out.print("Escriba el primer entero:");
        \\ La siguiente rutina se ejecuta mediante el método nextInt del objeto de entrada
        de la clase scanner y recibe un entero mediante el teclado

        num1 = entrada2. nextInt ();
        System.out.print("Escriba el segundo entero:");
        num2 = entrada2. nextInt ();
        Sum2 = num1 + num2;
        System.out.printf("La suma es %d\n", Sum2);
    } \FIN DEL MÉTODO
} \FIN DE LA CLASE
```

## Ejecución

### Figura 2

*Ejecución del Laboratorio 2*



```
Output - pc1 (run) x
run:
Escriba el primer entero: 2
Escriba el segundo entero: 2
La suma es 4 BUILD SUCCESSFUL (total time: 2 seconds)
|
```

### Laboratorio 3. Desarrollo de programa para comparación de enteros

Diseñar y desarrollar un programa que compara dos enteros utilizando instrucciones *if*, operadores relacionales.

**Tabla 3**

*Variables utilizadas en el Laboratorio 3*

Nombre de la variable	Tipo de variable	Propósito
<i>num1</i>	<i>Int</i>	Recibe un número desde el teclado
<i>num2</i>	<i>Int</i>	Almacena un número desde el teclado
<i>operador de igualdad</i>	<code>==</code>	Compara una variable con otra. Significa igual a
<i>operador relacional</i>	<code>&lt;</code>	Compara una variable con otra. Significa menor que
<i>operador relacional</i>	<code>&lt;=</code>	Compara una variable con otra. Significa, menor o igual que
<i>operador relacional</i>	<code>&gt;</code>	Compara una variable con otra. Significa mayor que
<i>operador relacional</i>	<code>&gt;=</code>	Compara una variable con otra. Significa mayor o igual que

Fuente: elaboración propia.

### Codificación

```
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Comparación {
/**
 * @param args the command line arguments
 */
    public static void main (String [] args) {
        // TODO code application logic here
        Scanner entrada3 = new Scanner (System.in);
        int num1;
        int num2;
        System.out.print("Escriba el primer entero:");
        num1= entrada3. nextInt ();
        System.out.print("Escriba el segundo entero:");
        num2= entrada3. nextInt ();
```

```

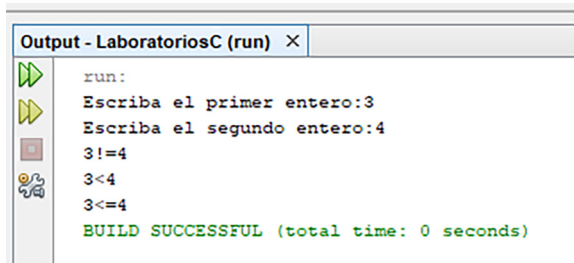
if(num1==num2)
    System.out.printf("%d==%d\n", num1, num2);
if (!num1! =num2)
    System.out.printf("%d! =%d\n", num1, num2);
if(num1<num2)
    System.out.printf("%d<%d\n", num1, num2);
if(num1>num2)
    System.out.printf("%d>%d\n", num1, num2);
if(num1<=num2)
    System.out.printf("%d<=%d\n", num1, num2);
if(num1>=num2)
    System.out.printf("%d>=%d\n", num1, num2);
}}

```

## Ejecución

**Figura 3**

*Ejecución del Laboratorio 3*



## Laboratorio 4. Desarrollo de programa de figuras compuestas por caracteres

Diseñar y desarrollar un programa que muestre las siguientes figuras:



**Tabla 4***Variables utilizadas en el Laboratorio 4*

Nombre	Propósito
<i>System.out.println</i>	Sirve para imprimir un argumento o texto introducido. Representa la salida estándar
<i>System</i>	Es una clase final definida en el paquete <i>java.lang</i>
<i>out</i>	Es una instrucción del tipo <i>PrintStream</i> , que es un campo miembro público y estático de la clase <i>System</i>
<i>println</i>	Imprime cualquier argumento que se le pase y salta una nueva línea a la salida

Fuente: elaboración propia.

### Codificación

```

package ejercicio8;
/**
 *
 * @author jxs_b
 */
public class Ejercicio8 {
    /**
     * @param args the command line arguments
     */
    public static void main (String [] args) {
        // Se aplica el código lógico aquí:
        System.out.println("*****   ***   *   *");
        System.out.println(" *   *   *   *   ***   *");
        System.out.println(" *   *   *   *   *****   *   *");
        System.out.println(" *   *   *   *   *   *   *");
        System.out.println(" *   *   *   *   *   *   *");
        System.out.println(" *   *   *   *   *   *   *");
        System.out.println(" *   *   *   *   *   *   *");
        System.out.println(" *   *   *   *   *   *   *");
        System.out.println("*****   ***   *   *");
    }
}

```

## Ejecución

**Figura 4**

*Ejecución del Laboratorio 4*

```
run:
*****  ***  *  *
*  *  *  *  *  *  *
*  *  *  *  *  *  *
*  *  *  *  *  *  *
*  *  *  *  *  *  *
*  *  *  *  *  *  *
*  *  *  *  *  *  *
*  *  *  *  *  *  *
*****  ***  *  *
BUILD SUCCESSFUL (total time: 0 seconds)
```

## Laboratorio 5. Desarrollo de programa de entrada de datos e impresión de resultados

Diseñar y desarrollar una aplicación que solicite al usuario dos datos numéricos, que los adicione e imprima suma, producto, diferencia, y división de los números.

**Tabla 5**

*Variables utilizadas en el Laboratorio 5*

Nombre de la variable	Tipo de variable	Propósito
<i>Num25</i>	<i>Int</i>	Almacena un número desde el teclado
<i>Num15</i>	<i>Int</i>	Guarda un número desde el teclado
<i>operador de igualdad</i>	<code>==</code>	Compara una variable con otra. Significa igual a
<i>operador relacional</i>	<code>&lt;</code>	Compara una variable con otra. Significa menor que
<i>operador relacional</i>	<code>&lt;=</code>	Compara una variable con otra. Significa, menor o igual que

Fuente: elaboración propia.



## Codificación

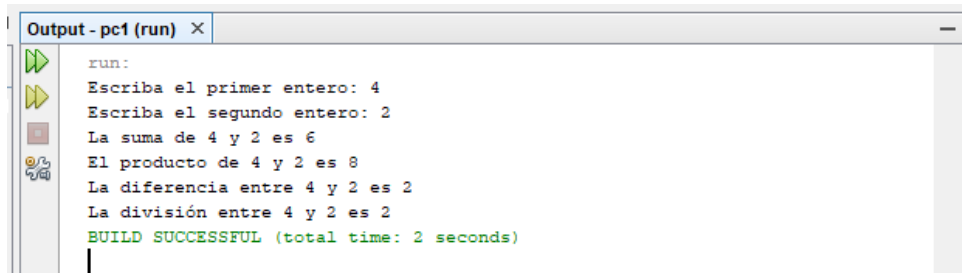
```
package ejercicio7;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio7 {
    /**
     * @param args instrucción para el traspaso de argumentos

    */
    public static void main (String [] args) {
        // Se aplica el código lógico aquí:
        Scanner entrada5 = new Scanner (System.in);
        int num15;
        int num25;
        int Suma5;
        int Producto5;
        int Diferen5;
        int Divis5;
        System.out.print("Escriba el primer entero:");
        num15 = entrada5. nextInt ();
        System.out.print("Escriba el segundo entero:");
        num25 = entrada5. nextInt ();
        Suma5 = num15 + num25;
        System.out.printf("La suma5 de %d y %d es %d\n", num15, num25, Suma5);
        Producto5= num15*num25;
        System.out.printf("El producto5 dé %d y %d es %d\n", num15, num25, Producto5);
        Diferen5= num15-num25;
        System.out.printf("La Diferen5 entre %d y %d es %d\n", num15, num25, Diferen5);
        Divis5= (num15/num25);
        System.out.printf("La Divis5 entre %d y %d es %d\n", num15, num25, Divis5);
    }
}
```

## Ejecución

**Figura 5**

*Ejecución del Laboratorio 5*



```
Output - pc1 (run) ×
run:
Escriba el primer entero: 4
Escriba el segundo entero: 2
La suma de 4 y 2 es 6
El producto de 4 y 2 es 8
La diferencia entre 4 y 2 es 2
La división entre 4 y 2 es 2
BUILD SUCCESSFUL (total time: 2 seconds)
```

## Laboratorio 6. Desarrollo de programa de impresión para diversos strings de caracteres

Diseña y desarrolla un programa que calcule e imprima la siguiente instrucción:

```
System.out.println("*\n **\n ***\n ****\n *****\n"); (Deitel, 2008).
```

**Tabla 6**

*Variables utilizadas en el Laboratorio 6*

Nombre	Propósito
<code>System.out.println</code>	Estatuto para imprimir un argumento o texto que se le pasa. Representa el flujo de salida estándar
<code>System</code>	Instrucción tipo clase final definida en el paquete <code>java.lang</code>
<code>out</code>	Estatuto del tipo <code>PrintStream</code> , que es un campo miembro público y estático de la clase <code>System</code>
<code>println</code>	Instrucción que imprime un argumento que se le pase y agrega una nueva línea en la salida
<code>\n</code>	Instrucción para salto de línea

Fuente: elaboración propia, basada en Geeksforgeeks (s.f.).

### Codificación:

```
package ejercicio10;
/**
 *
 * @author jxs_b
 */
public class Ejercicio10 {

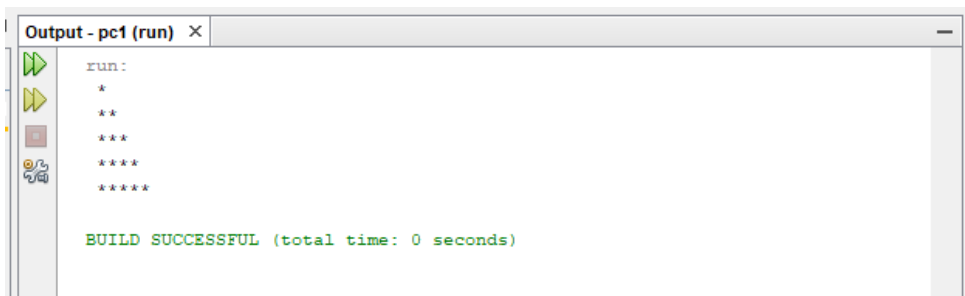
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        // Se aplica el código lógico aquí:
        System.out.println(" *\n **\n ***\n ****\n *****\n");
    }

}
```

### Ejecución

#### Figura 6

*Ejecución del Laboratorio 6*



```
Output - pc1 (run) ×
run:
 *
 **
 ***
 ****
 *****

BUILD SUCCESSFUL (total time: 0 seconds)
```

### Laboratorio 7. Desarrollo de programa de impresión de caracteres en diversas longitudes

Diseñar una aplicación que imprima la siguiente instrucción:

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

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

```
System.out.println("***"); (Deitel, 2008).
```

**Tabla 7**

Variables utilizadas en el Laboratorio 7

Nombre	Propósito
<i>System.out.println</i>	Instrucción para imprimir un argumento o texto que se le pasa. Representa el flujo de salida estándar
<i>System</i>	Estatuto tipo clase final definida en el paquete <i>java.lang</i>
<i>out</i>	Instrucción del tipo <i>PrintStream</i> que es un campo público y estático de la clase <i>System</i>
<i>println</i>	Estatuto que imprime cualquier argumento que se le transfiera y agrega una nueva línea a la salida

Fuente: elaboración propia, basada en Geeksforgeeks (s.f.).

## Codificación

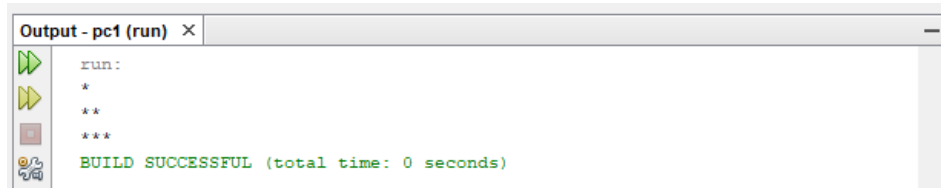
```
package ejercicio11;
/**
 *
 * @author jxs_b
 */
public class Ejercicio11 {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        // Se aplica el código lógico aquí:
        System.out.println("*");
        System.out.println("***");
        System.out.println("***");
    }
}
```

## Ejecución

**Figura 7**

*Ejecución del Laboratorio 7*



## Laboratorio 8. Desarrollo de programa que arroje datos a partir de un parámetro

Diseñar y desarrollar un programa que capture el radio de un círculo desde el teclado y que imprima el diámetro, la circunferencia y el área del círculo.

**Tabla 8**

*Variables utilizadas en el Laboratorio 8*

Nombre de la variable	Tipo de variable	Propósito
<i>System.out.println</i>	<i>string</i>	Estatuto para imprimir un argumento o texto que se le defina. Es la instrucción de salida estándar
<i>System</i>	<i>string</i>	Instrucción tipo clase final contenida en el paquete <i>java.lang</i>
<i>out</i>	<i>string</i>	Estatuto del tipo <i>PrintStream</i> , que es un campo público y estático de la clase <i>System</i>
<i>println</i>	<i>string</i>	Instrucción para imprimir algún argumento introducido y agrega una línea en blanco a la salida
<i>Diáme8</i>	<i>double</i>	Almacena el radio de un círculo
<i>Circunfe8</i>	<i>double</i>	Guarda el valor de la circunferencia de un círculo
<i>Área8</i>	<i>double</i>	Almacena el valor del área de un círculo

Fuente: elaboración propia, basada en Geeksforgeeks (s.f.).

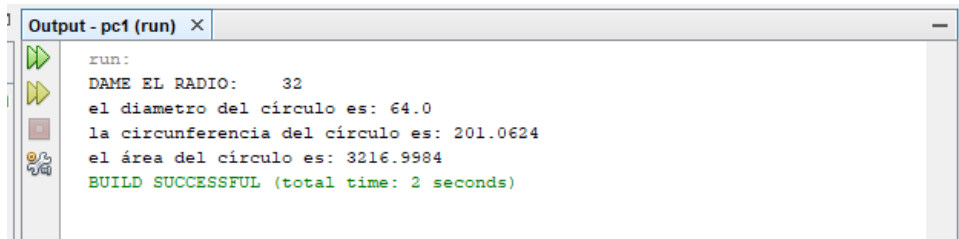
## Codificación

```
package ejercicio20;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio20 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada = new Scanner (System.in);
        double r;
        double diame8;
        double circunfe8;
        double area8;
        System.out.printf("DAME EL RADIO: ");
        r = entrada. nextInt ();
        diame8= 2 * r;
        System.out.println("el radio del círculo es;" +diame8);
        circunfe8 = (2*3.1416*r);
        System.out.println("la circunferencia del círculo es;" +circunfe8);
        area8 = (3.1416*r*r);
        System.out.println("el área del círculo es:" +area8);
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

**Figura 8**

*Ejecución del Laboratorio 8*



## Laboratorio 9. Desarrollo de programa de simulación de tablero de damas

Diseñar y desarrollar un programa para imprimir el tablero de damas (Deitel, 2008).

**Tabla 9**

*Variables utilizadas en el Laboratorio 9*

Nombre de la variable	Tipo de variable	Propósito
<i>System.out.println</i>	<i>string</i>	Estatuto para imprimir un argumento o texto que se le pasa. Representa el flujo de salida estándar
<i>System</i>	<i>string</i>	Instrucción tipo clase final definida en el paquete <i>java.lang</i>
<i>out</i>	<i>string</i>	Estatuto tipo <i>PrintStream</i> que es un campo miembro público y estático de la clase <i>System</i>
<i>println</i>	<i>string</i>	Instrucción para imprimir cualquier argumento que se le pase y agrega una nueva línea a la salida

Fuente: elaboración propia, basada en Geeksforgeeks (s.f.).

## Codificación

```
package ejercicio19;
/**
 *
 * @author jxs_b
 */
public class Ejercicio19 {
    /**
     * @param args the command line arguments
     */
    public static void main (String [] args) {
```

```

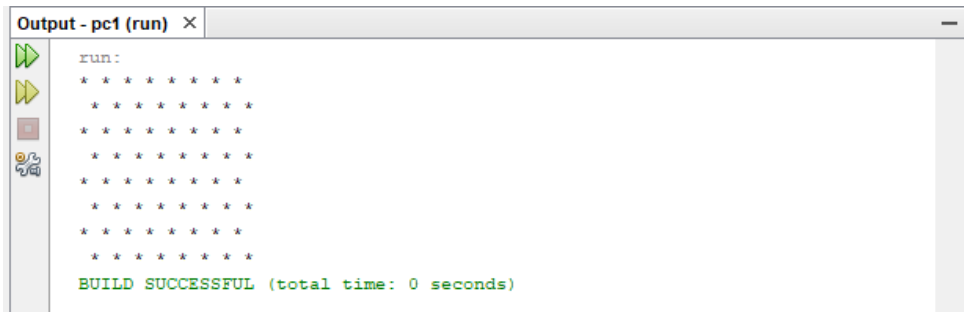
System.out.println("*****");
System.out.println("*****");
System.out.println("*****");
System.out.println("*****");
System.out.println("*****");
System.out.println("*****");
System.out.println("*****");
System.out.println("*****");
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 9

*Ejecución del Laboratorio 9*





## Laboratorio 10. Desarrollo de programa de aplicación de funciones matemáticas de cálculo

Desarrollar un programa que aplique todos de métodos matemáticos pidiendo los valores desde el teclado (*abs, ceil, cos, exp, floor, log, max, min, pow, sin, tan*).

**Tabla 10**

*Variables utilizadas en el Laboratorio 10*

Nombre de la variable	Tipo de variable	Propósito
<i>System.out.println</i>	<i>string</i>	Estatuto para imprimir un argumento o texto que se le pasa. Representa el flujo de salida estándar
<i>System</i>	<i>string</i>	Instrucción tipo clase final definida en el paquete <i>java.lang</i>
<i>out</i>	<i>string</i>	Estatuto tipo <i>PrintStream</i> , que es público y estático de la clase <i>System</i>
<i>println</i>	<i>string</i>	Instrucción para imprimir un argumento que se le defina y salta una línea en la salida
<i>A10</i>	<i>double</i>	Guarda el valor absoluto
<i>B10</i>	<i>double</i>	Almacena el valor de la circunferencia de un círculo
<i>C10</i>	<i>double</i>	Almacena el valor del área de un círculo
<i>D10</i>	<i>double</i>	Guarda el valor de la exponenciación de la variable
<i>E10</i>	<i>double</i>	Almacena el redondeo de una variable
<i>F10</i>	<i>double</i>	Guarda el logaritmo natural de la variable
<i>G10</i>	<i>double</i>	Almacena el mínimo de una variable
<i>H10</i>	<i>double</i>	Guarda el máximo de una variable
<i>I10</i>	<i>double</i>	almacena el valor de la variable elevada a la potencia especificada por otra variable
<i>J10</i>	<i>double</i>	Guarda el seno de la variable
<i>K10</i>	<i>double</i>	Almacena la tangente el valor de una variable

Fuente: elaboración propia, basada en Geeksforgeeks (s.f.).

### Codificación

```
package metodos.pkg1;
import java.util.Scanner;
/**
 *
 * @author jxs_b
 */
public class Metodos1 {
```

```

/**
 * @param args instrucción para el traspaso de argumentos
 */
public static void main (String [] args) {
    // Se aplica el código lógico aquí:
    Scanner entrada = new Scanner (System.in);
    double a10, b10, c10, d10, e10, f10, g10, h10, i10, j10, k10, x10, y10, z10, w0;

    System.out.println("RAÍZ CUADRADA");
    System.out.print("Dame valor de w10: ");
    w10=entrada. nextDouble ();
    System.out.print("la raíz cuadrada w10 es:");
    System.out.println(Math.sqrt(w10));
    System.out.println();
    //////////////////////////////////////
    System.out.println("VALOR ABSOLUTO");
    System.out.print("Dame valor de a10: ");
    a10=entrada. nextDouble ();
    System.out.print("el valor absoluto de a10 es:");
    System.out.println(Math.abs(a10));
    System.out.println();
    //////////////////////////////////////
    System.out.println("REDONDEA X AL ENTERO");
    System.out.print("Dame valor de b10: ");
    b10=entrada. nextDouble ();
    System.out.print("redondea b10 al entero más cercano es:");
    System.out.println(Math.ceil(b10));
    System.out.println();
    //////////////////////////////////////
    System.out.println("COSENO");
    System.out.print("Dame valor de c10: ");
    C10=entrada. nextDouble ();
    System.out.print("coseno de c10 es:");
    System.out.println(Math.cos(c10));
    System.out.println();
    //////////////////////////////////////
    System.out.println("EXPONENCIAL");

```

```

System.out.print("Dame valor de d10: ");
d10=entrada. nextDouble ();
System.out.print("el exponencial de d10 es:");
System.out.println(Math.exp(d10));
System.out.println();
////////////////////////////////////
System.out.println("REDONDEA X AL ENTERO MÁS GRANDE QUE NO SEA MAYOR
QUE X");
System.out.print("Dame valor de e10: ");
e10=entrada. nextDouble ();
System.out.print("redondea e10 al entero más grande:");
System.out.println(Math.floor(e10));
System.out.println();
////////////////////////////////////
System.out.println("LOGARITMO NATURAL DE X");
System.out.print("Dame valor de f10: ");
f10=entrada. nextDouble ();
System.out.print("logaritmo de f10 es:");
System.out.println(Math.log(f10));
System.out.println();
////////////////////////////////////
System.out.println("VALOR MÁXIMO DE G Y X");
System.out.print("Dame valor de g10: ");
g10=entrada. nextDouble ();
System.out.print("Dame valor de x10: ");
X10=entrada. nextDouble ();
System.out.print("el máximo de g10 y x10 es:");
System.out.println(Math.max (g10, x10));
System.out.println();
////////////////////////////////////
System.out.println("VALOR MÍNIMO DE H Y Y");
System.out.print("Dame valor de h10: ");
h10=entrada. nextDouble ();
System.out.print("Dame valor de y10: ");
Y10=entrada. nextDouble ();
System.out.print("el valor mínimo de h10 y y10 es:");
System.out.println(Math.min (h10, y10));
System.out.println();

```

```

////////////////////////////////////
System.out.println("X ELEVADO A LA POTENCIA Y");
System.out.print("Dame valor de i10: ");
i10=entrada.nextDouble ();
System.out.print("Dame valor a elevar z10: ");
Z10=entrada.nextDouble ();
System.out.print("i elevado a la z10 es:");
System.out.println(Math.pow (i10, z10));
System.out.println();
////////////////////////////////////
System.out.println("SENO DE X");
System.out.print("Dame valor de j10: ");
j10=entrada.nextDouble ();
System.out.print("seno de j10 es:");
System.out.println(Math.sin(j10));
System.out.println();
////////////////////////////////////
System.out.println("TANGENTE DE X");
System.out.print("Dame valor de k10: ");
k10=entrada.nextDouble ();
System.out.print("tangente de k10 es:");
System.out.println(Math.tan(k10));
System.out.println();
}
}

```

## Ejecución

### Figura 10

*Ejecución del Laboratorio 10*

```
Output - pc1 (run) X
run:
RAIZ CUADRADA
Dame valor de w: 4
la raiz cuadrada w es:2.0

VALOR ABSOLUTO
Dame valor de a: 5
el valor absoluto de a es:5.0

REDONDEA X AL ENTERO
Dame valor de b: 6
redondea b al entero más cercano es:6.0

COSENO
Dame valor de c: 7
coseno de c es:0.7539022543433046

EXPONENCIAL
Dame valor de d: 5
el exponencial de d es:148.4131591025766

REDONDEA X AL ENTERO MAS GRANDE QUE NO SEA MAYOR QUE X
Dame valor de e: 5.2
redondea e al entero más grande:5.0

LOGARITMO NATURAL DE X
Dame valor de f: 2
logaritmo de f es:0.6931471805599453

VALOR MAXIMO DE G Y X
Dame valor de g: 4
Dame valor de x: 5
el maximo de g y x es:5.0

VALOR MINIMO DE H Y Y
Dame valor de h: 2
Dame valor de y: 3
el valor minimo de h y y es:2.0

X ELEVADO A LA POTENCIA Y
Dame valor de i: 2
Dame valor a elevar z: 3
i elevado a la z es:8.0

SENO DE X
Dame valor de j: 12
seno de j es:-0.5365729180004349

TANGENTE DE X
Dame valor de k: 3
tangente de k es:-0.1425465430742778

BUILD SUCCESSFUL (total time: 56 seconds)
|
```

## 1.6. Ejercicios para resolver. Casos prácticos

Diseñe y desarrolle un programa que lea 10 edades de alumnos de la UAT y que imprima cuál de esas edades es la mayor.

Diseñe y desarrolle un programa que obtenga el promedio de edades de 10 alumnos de la UAT e imprimirlos.

Diseñe y desarrolle un programa que genere las numeraciones del 1 al 20 y que imprima el mensaje "numeraciones del 1-20" y por separado los nones y los pares.

Desarrolle un programa para desplegar en pantalla por separado los números del 1 al 20, donde se obtenga el cubo de los números nones y el cuadrado de los pares, para imprimirlos en el siguiente orden:

NONES CUBO PARES CUADRADO

Diseñe y desarrolle un programa que capture mediante el teclado los siguientes datos: RFC, Nombre, Turno, Categoría, Sueldo diario, Días trabajados, Comisión, Horas extras se pagan al doble, Faltas se le descuenta el 20% de su salario diario, Cuota del ISPT, Cuota sindical y Cuota del IMSS que muestra lo siguiente en la pantalla:

Universidad Autónoma de Tamaulipas  
Cálculo de sueldos

R.FC	FEMJ6923923
Nombre	Vicente Villanueva
Turno	Matutino
Categoría	Coordinador
Sueldo quincenal	150
Horas extras	3
Faltas	1
Comisiones	300
Cuota del ISPT	200
Cuota del IMSS	200
Cuota sindical	200

El sueldo de Vicente Villanueva es...

Los descuentos son de...

Diseñe y desarrolle un programa que obtenga el siguiente resultado con N empleados

Florería El Romance  
Reportes de ventas

Número de empleado	233
Nombre	Blanquita
Turno	Matutino
Venta1	34000
Venta2	56000
Venta3	15000
Venta4	70000
Venta5	80000
Total de ventas diarias	?
Comisión	?

DESEAS CONTINUAR S/N

Si es N entonces deberá de imprimir lo siguiente:

Total de empleados procesados	?
Total de ventas generales	?
Total de comisiones generales	?

Nota: La comisión se obtendrá del 15 % sobre el total de ventas diarias.

## 1.7. Referencias de consulta

<http://www.manualweb.net/java/introduccion-java/>

[https://es.wikipedia.org/wiki/Java\\_\(lenguaje\\_de\\_programaci%C3%B3n\)](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n))

<http://www.manualweb.net/java/>

## 1.8. Videos de apoyo

<https://www.youtube.com/watch?v=L1oML...>

<https://www.youtube.com/watch?v=Cs5ymoNkrX8>





# Capítulo 2

## Estructuras de control *if*

### 2.1. Objetivo del capítulo

Que el alumno empiece a manejar las estructuras *control* condicionales a partir de una condición simple o con varias alternativas de solución (anidadas).

### 2.2. Introducción

El estatuto *if* es crucial para cualquier superlenguaje computacional y Java no es la excepción, porque las necesidades de bifurcaciones a través de un programa necesitan de una instrucción que brinde la oportunidad de evaluar condiciones de una variable y dependiendo de sus valores el estatuto *if* deja ejecutar el curso normal de las instrucciones siguientes o altera este curso normal mediante un salto a otro punto en el programa para ejecutar otro conjunto de instrucciones que así convengan. Al respecto Ceballos (2008) lo expresa de manera concreta:

[...] es una instrucción que evalúa la expresión condición, y si es verdadera, se ejecutan las instrucciones que están después de la condición del *if* desde la primera llave hasta la segunda llave. Si la condición no se cumple las instrucciones a ejecutar serían las que están después de la palabra reservada *else*. En contraste, si la condición del *if* se cumple, se ejecuta lo que está dentro de las llaves.

### 2.3. Ejemplos prácticos realizados con el comando *if*

Laboratorio 11. Desarrollo de programa que dé resultados a partir de números enteros alimentados por teclado

Diseñar y desarrollar un programa que a partir de dos datos solicitados al usuario imprima el número mayor y en caso de que sean iguales despliegue “estos números son iguales”.

## Codificación

```
package ejercicio4;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio4 {
/**
 * @param args instrucción para el traspaso de argumentos

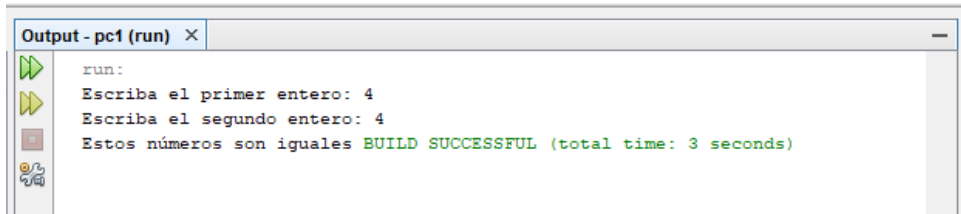
 */
public static void main (String [] args) {
    // Se aplica el código lógico aquí:
    Scanner entrada = new Scanner (System.in);
    int num111;
    int num211;

    System.out.print("Escriba el primer entero:");
    num111 = entrada. nextInt ();
    System.out.print("Escriba el segundo entero:");
    num211 = entrada. nextInt ();
    if (num111 > num211)
    {
        System.out.printf("El número %d es mayor que %d", num111, num211);
    }
    if (num111< num211)
    {
        System.out.printf("El número %d es menor que %d", num111, num211);
    }
    if (num111 == num211)
    {
        System.out.print("Estos números son iguales");
    }
}
}
```

## Ejecución

### Figura 11

*Ejecución del Laboratorio 11*



## Laboratorio 12. Desarrollo de programa que genere resultados a partir de datos interactivos

Diseñar y desarrollar un programa que a partir de tres enteros alimentados por teclado imprima la suma, el producto, el mayor y el promedio de los datos alimentados.

## Codificación

```
package ejercicio5;
import java.util. Scanner;

/**
 *
 * @author jxs_b
 */
public class Ejercicio5 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        // Se aplica el código lógico aquí:
        Scanner entrada = new Scanner (System.in);
        int num112;
        int num212;
        int num312;
        int Producto;
        int Suma;
        System.out.print("Escriba el primer entero:");
```

```

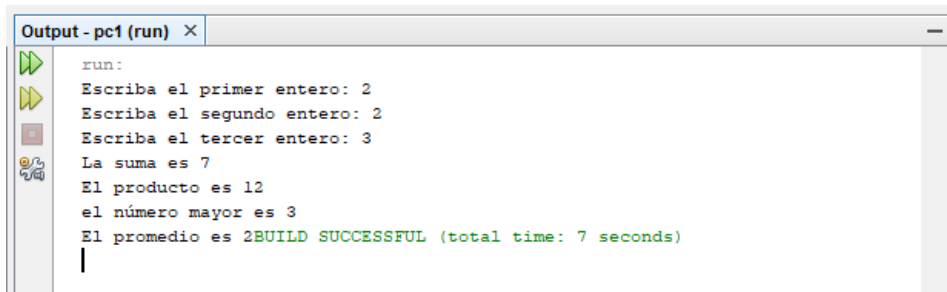
num112 = entrada.nextInt ();
System.out.print("Escriba el segundo entero:");
num212 = entrada.nextInt ();
System.out.print("Escriba el tercer entero:");
num312 = entrada.nextInt ();
Suma = num112 + num212 + num312;
System.out.printf("La suma es %d\n", Suma);
Producto = num112 * num212 * num312;
System.out.printf("El producto es %d\n", Producto);
if ((num112> num212) &&(num112> num312))
    System.out.printf("el número mayor es %d\n ", num112);
if ((num212> num112) &&(num212> num312))
{
    System.out.printf("el número mayor es %d\n ", num212);
}
if((num312> num112) &&(num312> num212))
{
    System.out.printf("el número mayor es %d\n ", num312);
}
if ((num112< num212) && (num112< num312))
    System.out.printf("el número menor es %d \n ", num112);
if ((num212< num112) && (num212< num312))
{
    System.out.printf("el número menor es %d \n ", num212);
}
if ((num312< num112) &&(num312<num212))
{
    System.out.printf("el número menor es %d \n ", num312);
}
}
if (Suma == 7)
{
    System.out.print("El promedio es 2");
}
}
}

```

## Ejecución

Figura 12

*Ejecución del Laboratorio 12*



```
Output - pc1 (run) X
run:
Escriba el primer entero: 2
Escriba el segundo entero: 2
Escriba el tercer entero: 3
La suma es 7
El producto es 12
el número mayor es 3
El promedio es 2BUILD SUCCESSFUL (total time: 7 seconds)
|
```

## Laboratorio 13. Desarrollo de programa que arroje dictamen de nivel de aprovechamiento

Diseñar y desarrollar un programa para reportar el rendimiento académico de un alumno que solicite al usuario una calificación del 1 al 10 e imprimir “deficiente” si obtiene una calificación de 5, si saca 6 imprimir “suficiente”, si obtiene 7 imprimir “bien”, si saca 8 imprimir “notable”, si obtiene 9 imprimir “muy bien” y 10 imprimir “excelente”.

## Codificación

```
package tarea1;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Tarea1 {
/**
 * @param args instrucción para el traspaso de argumentos
 */
public static void main (String [] args) {
int calif13;
Scanner entrada = new Scanner (System.in);
System.out.print("Dame UNA Calif13: ");
calif13=entrada. nextInt ();
if (calif13 == 1)
```

```
{
    System.out.printf("%d es Deficiente", calif13);
}
if (calif13 == 2)
{
    System.out.printf("%d es Deficiente", calif13);
}
if (calif13 == 3)
{
    System.out.printf("%d es Deficiente", calif13);
}
if (calif13 == 4)
{
    System.out.printf("%d es Deficiente", calif13);
}
if (calif13 == 5)
{
    System.out.printf("%d es Deficiente", calif13);
}
if (calif13 == 6)
{
    System.out.printf("%d es Suficiente", calif13);
}
if (calif13 == 7)
{
    System.out.printf("%d está Bien", calif13);
}
if (calif13 == 8)
{
    System.out.printf("%d es Notable", calif13);
}
if (calif13 == 9)
{
    System.out.printf("%d está Muy Bien", calif13);
}
if (calif13 == 10)
{
    System.out.printf("%d es Excelente", calif13);
}
```

```

    }
    // Se aplica el código lógico aquí:
}

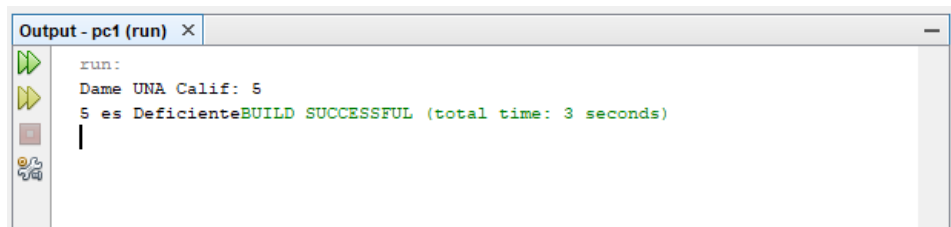
}

```

## Ejecución

**Figura 13**

*Ejecución del Laboratorio 13*



## Laboratorio 14. Desarrollo de programa que ordene datos de mayor a menor

Diseñar y desarrollar un programa que lea mediante el teclado tres números y los imprima en forma ordenada de mayor a menor.

## Codificación

```

package tarea2;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Tarea2 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada = new Scanner (System.in);
        int num114;
        int num214;
        int num314;
    }
}

```

```

System.out.printf("Teclea el primer número:");
num114 = entrada.nextInt ();
System.out.printf("Teclea el segundo número:");
num214 = entrada.nextInt ();
System.out.printf("Teclea el tercer número:");
num314 = entrada.nextInt ();
if ((num114 > num214) && (num114 > num314) &&(num214>num314))
{
System.out.printf("%d. %d, %d\n", num114, num214, num314);
}
if ((num114 > num214) && (num114 > num314) &&(num314>num214))
{
System.out.printf("%d. %d, %d\n", num114, num314, num214);
}
if ((num214 > num114) && (num214 > num314) &&(num114>num314))
{
System.out.printf("%d. %d, %d\n", num214, num114, num314);
}
if ((num214>num114) &&(num314>num114) &&(num314>num114))
{
System.out.printf("%d. %d, %d\n", num214, num314, num114);
}
if ((num314 > num214) && (num314 > num114) &&(num214>num114))
{
System.out.printf("%d. %d, %d\n", num314, num214, num114);
}
if ((num314 > num214) && (num314 > num114) &&(num114>num214))
{
System.out.printf("%d. %d, %d\n", num314, num114, num214);
}
// Se aplica el código lógico aquí:
// System.out.printf("%d, %d, %d\n", num114, num314, num214);
}
}

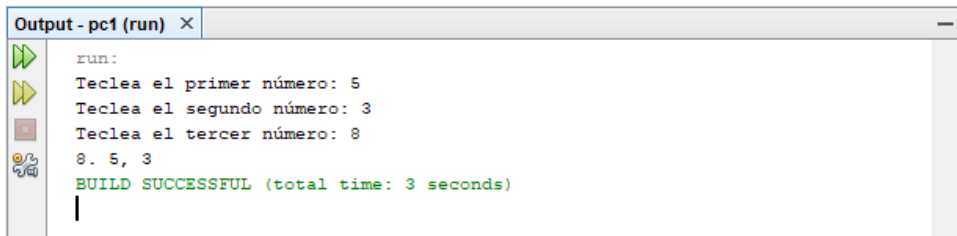
```



## Ejecución

### Figura 14

*Ejecución del Laboratorio 14*



```
Output - pc1 (run) X
run:
Teclea el primer número: 5
Teclea el segundo número: 3
Teclea el tercer número: 8
8. 5, 3
BUILD SUCCESSFUL (total time: 3 seconds)
```

## Laboratorio 15. Desarrollo de programa que arroje resultados de números positivos y negativos

Diseñar y desarrollar un programa que reciba cinco números por el teclado y que imprima cuántos son números negativos y cuántos son ceros además de la cantidad de números positivos de esos números capturados.

## Codificación

```
package ejercicio18;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio18 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada = new Scanner (System.in);
        int num115;
        int num215;
        int num315;
        int num415;
        int num515;
        System.out.printf("Teclea el primer número:");
        num115 = entrada. nextInt ();
        System.out.printf("Teclea el segundo número:");
```

```

num215 = entrada.nextInt ();
System.out.printf("Teclea el tercer número:");
num315 = entrada.nextInt ();
System.out.printf("Teclea el cuarto número:");
num415 = entrada.nextInt ();
System.out.printf("Teclea el quinto número:");
num515 = entrada.nextInt ();
int n=0;
int numeropositivo = 0;
int numeronegativo = 0;
if (num115 == 0)
{
    n=n+1;
}
if(num115<0)
{
    numeronegativo=numeronegativo+1;
}
if(num115>0)
{
    numeropositivo = numeropositivo + 1;
}
if (num215 == 0)
{
    n=n+1;
}
if(num215<0)
{
    numeronegativo=numeronegativo+1;
}
if(num215>0)
{
    numeropositivo = numeropositivo + 1;
}
if (num315 == 0)
{
    n=n+1;
}

```

```

if(num315<0)
{
numeronegativo=numeronegativo+1;
}
if(num315>0)
{
numeropositivo = numeropositivo + 1;
}
if (num415 == 0)
{
n=n+1;
}
if(num415<0)
{
numeronegativo=numeronegativo+1;
}
if(num415>0)
{
numeropositivo = numeropositivo + 1;
}
if (num515 == 0)
{
n=n+1;
}
if(num515<0)
{
numeronegativo=numeronegativo+1;
}
if(num515>0)
{
numeropositivo = numeropositivo + 1;
}

```

```

System.out.printf("Hay %d números negativos y %d valor cero y %d números
positivos", numeronegativo, n, numeropositivo);

```

```

// Se aplica el código lógico aquí:

```

```

}

```

```

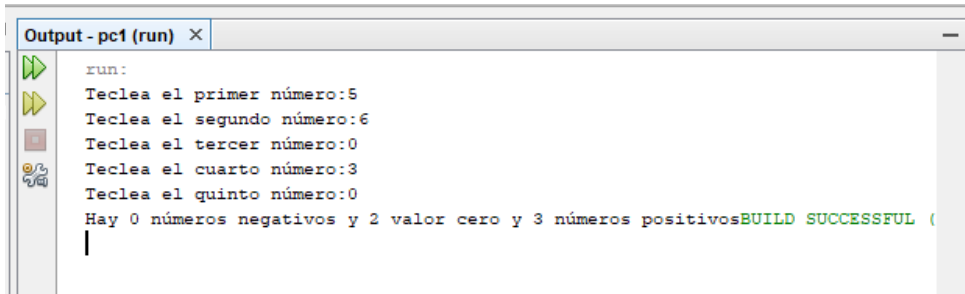
}

```

## Ejecución

### Figura 15

*Ejecución del Laboratorio 15*



Laboratorio 16. Desarrollo de programa que genere el resultado de los dígitos que forman los números capturados

Diseñar y desarrollar un programa que capture un número mediante el teclado entre el 0 y 10 000 e imprimir cuántas cifras tiene ese número.

## Codificación

```
package tarea3;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Tarea3 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {

        Scanner entrada = new Scanner (System.in);
        int num16;
        System.out.print("Dame un num16: ");
        num16=entrada. nextInt ();
        if((num16>=1) &&(num16<10))
        {
            System.out.printf("Este número %d tiene 1 cifra ", num16);
        }
    }
}
```

```

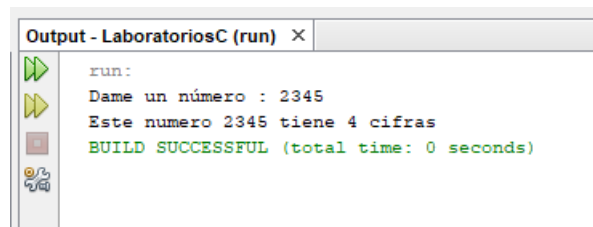
if((num16>=10) &&(num16<100))
{
System.out.printf("Este número %d tiene 2 cifras ", num16);
}
if((num16>=100) &&(num16<1000))
{
System.out.printf("Este número %d tiene 3 cifras ", num16);
}
if((num16>=1000) &&(num16<10000))
{
System.out.printf("Este número %d tiene 4 cifras ", num16);
}
if((num16>=10000) &&(num16<100000))
{
System.out.printf("Este número %d tiene 5 cifras ", num16);
}
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 16

*Ejecución del Laboratorio 16*



```

Output - LaboratoriosC (run) ×
run:
Dame un número : 2345
Este numero 2345 tiene 4 cifras
BUILD SUCCESSFUL (total time: 0 seconds)

```

## Laboratorio 17. Desarrollo de programa que imprima números a la inversa

Diseñar y desarrollar un programa que pida al usuario mediante el teclado un número entre el 0 y 10 000 y que lo imprima al revés.

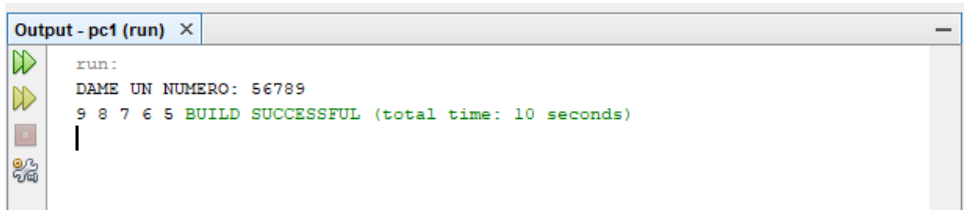
### Codificación

```
package tarea4;
import java. util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Tarea4 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada = new Scanner (System.in);
        int num17, v17, w17, x17, y17, z17;
        System.out.print("DAME UN NÚMERO: ");
        num17=entrada. nextInt ();
        if((num17>=1) &&(num17<=100000))
        {
            V17 = num17/10000;
            W17 = (num17 %10000) /1000;
            X17 = (num17%10000) %1000/100;
            Y17 = ((num17%10000) %1000) %100/10;
            Z17 = (((num17%10000) %1000) %100) %10;
            System.out.printf("%d %d %d %d %d", z17, y17, x17, w17, v17);
        }
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

**Figura 17**

*Ejecución del Laboratorio 17*



Laboratorio 18. Desarrollo de programa que arroje resultados de números positivos, negativos y 0

Diseñar y desarrollar un programa que reciba 10 números por el teclado e imprima la media de los que sean positivos, negativos y la cantidad de 0.

## Codificación

```
package mediadenumeros;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Mediadenumeros {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
Scanner hola = new Scanner (System.in);
        int a18; int b18; int c18; int d18; int e18; int f18; int g18; int h18;
        int i18;
        int j18;
        int cero18=0;
        int numpo18=0;
        int numneg18=0;
        int pos18=0;
        int neg18=0;
        double mediane18;
```

```

double mediapo18;
System.out.print("Teclea el primer número: ");
a18=hola.nextInt ();
System.out.print("Teclea el segundo número: ");
b18=hola.nextInt ();
System.out.print("Teclea el tercer número: ");
c18=hola.nextInt ();
System.out.print("Teclea el cuarto número: ");
d18=hola.nextInt ();
System.out.print("Teclea el quinto número: ");
e18=hola.nextInt ();
System.out.print("Teclea el sexto número: ");
f18=hola.nextInt ();
System.out.print("Teclea el séptimo número: ");
g18=hola.nextInt ();
System.out.print("Teclea el octavo número: ");
h18=hola.nextInt ();
System.out.print("Teclea el noveno número: ");
i18=hola.nextInt ();
System.out.print("Teclea el décimo número: ");
j18=hola.nextInt ()
if(a18>0)
{
    pos18=pos18+a18;
}
if(b18>0)
{
    pos18=pos18+b18;
}
if(c18>0)
{
    pos18=pos18+c18;
}
if(d18>0)
{
    pos18=pos18+d18;
}
if(e18>0)

```



```
{
  pos18=pos18+e18;
}
  if(f18>0)
  {
    pos18=pos18+f18;
  }
    if(g18>0)
    {
      pos18=pos18+g18;
    }
      if(h18>0)
      {
        pos18=pos18+h18;
      }
        if(i18>0)
        {
          pos18=pos18+i18;
        }
          if(j18>0)
          {
            pos18=pos18+j18;
          }
            if(a18>0)
            {
              numpo18=numpo18+1;
            }
              if(b18>0)
              {
                numpo18=numpo18+1;
              }
                if(c18>0)
                {
                  numpo18=numpo18+1;
                }
                  if(d18>0)
                  {
                    numpo18=numpo18+1;
                  }

```

```

}
    if(e18>0)
    {
        numpo18=numpo18+1;
    }
    if(f18>0)
    {
        numpo18=numpo18+1;
    }
    if(g18>0)
    {
        numpo18=numpo18+1;
    }
    if(h18>0)
    {
        numpo18=numpo18+1;
    }
    if(i18>0)
    {
        numpo18=numpo18+1;
    }
    if(j18>0)
    {
        numpo18=numpo18+1;
    }
    if(a18<0)
    {
        neg18=neg18+a18;
    }
    if(b18<0)
    {
        neg18 =neg18+b18;
    }
    if(c18<0)
    {
        neg18=neg18+c18;
    }
    if(d18<0)

```

```

{
  neg18=neg18+d18;
}
  if(e18<0)
{
  neg18=neg18+e18;
}
  if(f18<0)
{
  neg18=neg18+f18;
}
  if(g18<0)
{
  neg18=neg18+g18;
}
  if(h18<0)
{
  neg18=neg18+h18;
}
  if(i18<0)
{
  neg18=neg18+i18;
}
  if(j18<0)
{
  neg18=neg18+j18;
}
  if(a18<0)
{
  numne18=numne18+1;
}
  if(b18<0)
{
  numne18 =numne18+1;
}
  if(c18<0)
{
  numne18=numne18+1;
}

```

```

}
  if(d18<0)
  {
    numne18=numne18+1;
  }
  if(e18<0)
  {
    numne18=numne18+1;
  }
  if(f18<0)
  {
    numne18=numne18+1;
  }
  if(g18<0)
  {
    numne18=numne18+1;
  }
  if(h18<0)
  {
    numne18=numne18+1;
  }
  if(i18<0)
  {
    numne18=numne18+1;
  }
  if(j18<0)
  {
    numne18=numne18+1;
  }
  if(a18==0)
  {
    cero18=cero18+1;
  }
  if(b18==0)
  {
    cero18 =cero18+1;
  }
  if(c18==0)

```

```

    {
        cero18=cero18+1;
    }
    if(d18==0)
    {
        cero18=cero18+1;
    }
    if(e18==0)
    {
        cero18=cero18+1;
    }
    if(f18==0)
    {
        cero18=cero18+1;
    }
    if(g18==0)
    {
        cero18=cero18+1;
    }
    if(h18==0)
    {
        cero18=cero18+1;
    }
    if(i18==0)
    {
        cero18=cero18+1;
    }
    if(j18==0)
    {
        cero18=cero18+1;
    }
    mediapo18=pos18 /numpo18;
    mediane18=neg18 /numne18;

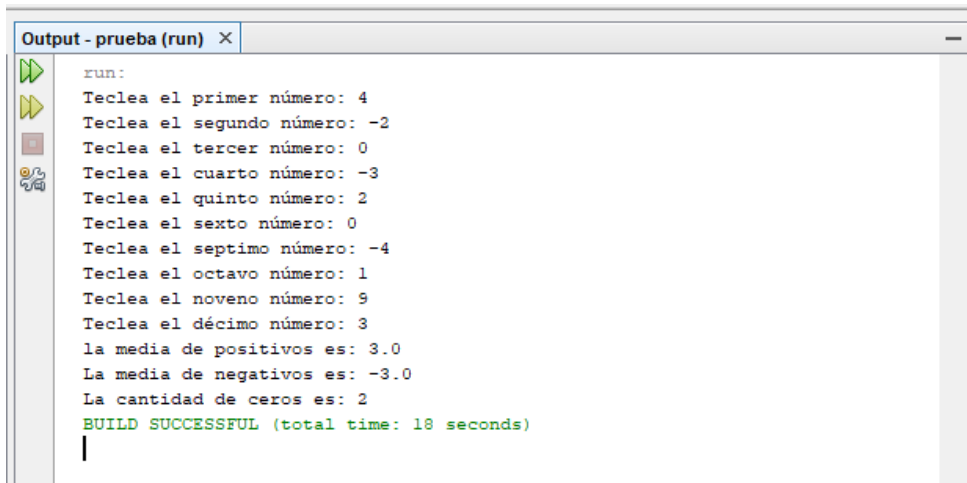
    System.out.println("la media de positivos es: "+mediapo18);
    System.out.println("La media de negativos es: "+mediane18);
    System.out.println("La cantidad de ceros es: "+cero18);
}
}

```

## Ejecución

**Figura 18**

*Ejecución del Laboratorio 18*



```
run:
Teclea el primer número: 4
Teclea el segundo número: -2
Teclea el tercer número: 0
Teclea el cuarto número: -3
Teclea el quinto número: 2
Teclea el sexto número: 0
Teclea el séptimo número: -4
Teclea el octavo número: 1
Teclea el noveno número: 9
Teclea el décimo número: 3
la media de positivos es: 3.0
La media de negativos es: -3.0
La cantidad de ceros es: 2
BUILD SUCCESSFUL (total time: 18 seconds)
|
```

## Laboratorio 19. Desarrollo de programa que a partir de datos cuantitativos genere como resultado la media

Diseñar y desarrollar un programa que capture las edades y la altura de 5 alumnos del CBTis 7 de la ciudad de Reynosa, Tamaulipas. Obtener lo siguiente:

- La media de la edad
- La media de las estaturas de los alumnos
- ¿Cuántos alumnos rebasaron o tuvieron 18 años?
- El número de alumnos con estatura igual a 1.75

## Codificación

```
package estaturas;
import java.util.Scanner;
/**
 *
 * @author jxs_b
 */
public class Estaturas {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
```

```

public static void main (String [] args) {
Scanner hola19 = new Scanner (System.in);
int a19; int b19; int c19; int d19; int e19;
double a119; double b119; double c119; double d119; double e119;
double suma19=0; double sum19=0; double meeda19=0; double meesta19=0;
int esta19=0; int eda19=0;
System.out.print("Teclea la primera edad: ");
a19=hola19.nextInt ();
System.out.print("Teclea la primera estatura: ");
a119=hola19.nextDouble();
System.out.print("Teclea la segunda edad: ");
b19=hola19.nextInt ();
System.out.print("Teclea la segunda estatura: ");
b119=hola19.nextDouble();
System.out.print("Teclea la tercera edad: ");
c19=hola19.nextInt ();
System.out.print("Teclea la tercera estatura: ");
c119=hola19.nextDouble();
System.out.print("Teclea la cuarta edad: ");
d19=hola19.nextInt ();
System.out.print("Teclea la cuarta estatura: ");
d119=hola19.nextDouble();
System.out.print("Teclea la quinta edad: ");
e19=hola19.nextInt ();
System.out.print("Teclea la quinta estatura: ");
e119=hola19.nextDouble();
if(a119==1.75)
{
esta19=esta+1;
}
if(b119==1.75)
{
esta19=esta19+1;
}
if(c119==1.75)
{
esta19=esta19+1;
}
}

```

```

    if(d119==1.75)
    {
        esta19=esta19+1;
    }
    if(e119==1.75)
    {
        esta19=esta19+1;
    }
    if(a19>18)
    {
        eda19=eda19+1;
    }
        if(b19>18)
    {
        eda19=eda19+1;
    }
        if(c19>18)
    {
        eda19=eda19+1;
    }
        if(d19>18)
    {
        eda19=eda19+1;
    }
        if(e19>18)
    {
        eda19=eda19+1;
    }
    sum19=a19+b19+c19+d19+e19;
    meeda19=sum19/5;
    System.out.println("La media de edad de alumnos es: "+meeda19);
    suma19=a119+b119+c119+d119+e119;
    meesta19=suma19/5;
    System.out.println("La media de estaturas de alumnos es: "+meesta19);
    System.out.println("La cantidad de alumnos mayores de 18 es: "+eda19);
    System.out.println("La cantidad de alumnos que mide 1.75 es: "+esta19);
    // Se aplica el código lógico aquí:
}
}

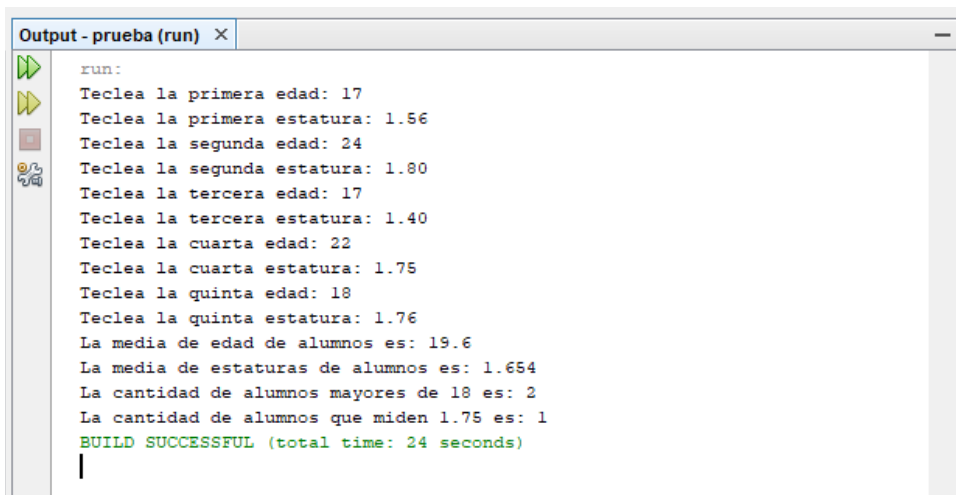
```



## Ejecución

Figura 19

*Ejecución del Laboratorio 19*



```
run:
Teclea la primera edad: 17
Teclea la primera estatura: 1.56
Teclea la segunda edad: 24
Teclea la segunda estatura: 1.80
Teclea la tercera edad: 17
Teclea la tercera estatura: 1.40
Teclea la cuarta edad: 22
Teclea la cuarta estatura: 1.75
Teclea la quinta edad: 18
Teclea la quinta estatura: 1.76
La media de edad de alumnos es: 19.6
La media de estaturas de alumnos es: 1.654
La cantidad de alumnos mayores de 18 es: 2
La cantidad de alumnos que miden 1.75 es: 1
BUILD SUCCESSFUL (total time: 24 seconds)
```

Laboratorio 20. Desarrollo de programa para una mipyme que a partir de datos interactivos, arroje resultados sobre las ventas realizadas

Diseñar y desarrollar un programa de una mipyme llamada Todo con todo que se dedica a vender veneno para cucarachas, esta empresa pequeña necesita dar facturas al cliente.

En cada factura, debe de ir la cantidad vendida de veneno y el precio. Se pide al usuario que capture 5 facturas mediante el teclado. Obtener la cantidad de facturas que se emitieron, la cantidad de litros vendidos y la cantidad de facturas que se emitieron arriba de 6000 pesos.

## Codificación

```
package factura;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Factura {

/**
```

```

* @param args instrucción para el traspaso de argumentos
*/
public static void main (String [] args) {
    Scanner hola20 = new Scanner (System.in);
double a; double b; double c; double d; double e; double a1; double b1;
double c1; double d1; double e1; double suma20=0; double sum20=0; double
mas20=0;
    System.out.print("Teclea la primera factura: ");
    a20=hola20.nextDouble ();
    System.out.print("Teclea cuántos litros vendidos: ");
    a120=hola20.nextDouble();
    System.out.print("Teclea la segunda factura: ");
    b20=hola20. nextDouble ();
    System.out.print("Teclea cuántos litros vendidos: ");
    b120=hola20.nextDouble();
    System.out.print("Teclea la tercera factura: ");
    c20=hola20. nextDouble ();
    System.out.print("Teclea cuántos litros vendidos: ");
    c120=hola20.nextDouble();
    System.out.print("Teclea la cuarta factura: ");
    d20=hola20. nextDouble ();
    System.out.print("Teclea cuántos litros vendidos: ");
    d120=hola20.nextDouble();
    System.out.print("Teclea la quinta factura: ");
    e20=hola20. nextDouble ();
    System.out.print("Teclea cuántos litros vendidos: ");
    e120=hola20.nextDouble();
    if(a20>6000)
    {
        mas20=mas20 +1;
    }
    if(b20>6000)
    {
        mas20=mas20 +1;
    }
    if(c20>6000)
    {
        mas20=mas20 +1;
    }
}

```

```

    }
    if(d20>6000)
    {
        mas20=mas20 +1;
    }
    if(e20>6000)
    {
        mas20=mas20 +1;
    }
    Suma20=a20+b20+c20+d20+e20;
    Sum20=a120+b120+c120+d120+e120;
    System.out.println("La suma de las facturas es: "+suma20);
    System.out.println(" La cantidad de litros vendidos es: "+sum20);
    System.out.println(" La cantidad de facturas mayores a $6000 es: "+mas20);
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

**Figura 20**

*Ejecución del Laboratorio 20*

```

Output - prueba (run) x
run:
Teclea la primera factura: 30000
Teclea cuántos litros vendidos: 4
Teclea la segunda factura: 23000
Teclea cuántos litros vendidos: 13
Teclea la tercera factura: 12345
Teclea cuántos litros vendidos: 6
Teclea la cuarta factura: 15678
Teclea cuántos litros vendidos: 8
Teclea la quinta factura: 13453
Teclea cuántos litros vendidos: 12
La suma de las facturas es: 94476.0
La cantidad de litros vendidos es: 43.0
La cantidad de facturas mayores a $6000 es: 5.0
BUILD SUCCESSFUL (total time: 25 seconds)

```

## Laboratorio 21. Desarrollo de programa que arroje el valor máximo, a partir de valores capturados

Diseñar y desarrollar un programa que busque el valor máximo de tres valores capturados por medio del teclado.

### Codificación

```
package buscador. máximo;
import java. util. Scanner;
/**
 *
 * @author obrya
 */
public class BuscadorMaximo {
    /**
     * @param args instrucción para el traspaso de argumentos */
    // obtiene tres valores de punto flotante y determina el valor máximo
    public void determinarMaximo ()
    { // crea objeto Scanner para introducir datos desde la ventana de comandos
        Scanner entrada21 = new Scanner (System.in) ; // pide y recibe como entrada21 tres
valores de punto flotante
        System.out.print("Escriba tres valores de punto flotante separados por espacios: ");
        double numero121 = entrada21. nextDouble (); // lee el primer valor double
        double numero221 = entrada21. nextDouble (); // lee el segundo valor double
        double numero321 = entrada21. nextDouble (); // lee el tercer valor double
        // determina el valor máximo
        double resultado = máximo (numero121, numero221, numero321);
        // muestra el valor máximo
        System.out.println("El máximo es: " + resultado);
    } // fin del método determinar Máximo

    // devuelve el máximo de sus tres parámetros double
    public double máximo (double x21, double y21, double z21)
    {
        double valorMaximo21 = x21;
        // asume que x21 es el mayor para empezar
        // determina si y21 es mayor que valorMaximo21
        if (y21 > valorMaximo21)
            valorMaximo21 = y21;
```

```

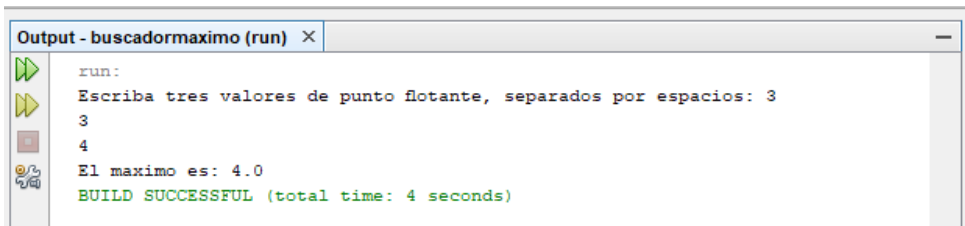
// determina si z21 es mayor que valorMaximo21
if (z21 > valorMaximo21)
    valorMaximo21 = z21;
return valorMaximo21;
} // fin del método máximo
} // fin de la clase Bu
package buscador. máximo;
/**

```

## Ejecución

### Figura 21

*Ejecución del Laboratorio 21*



```

Output - buscadormaximo (run) x
run:
Escriba tres valores de punto flotante, separados por espacios: 3
3
4
El maximo es: 4.0
BUILD SUCCESSFUL (total time: 4 seconds)

```

## 2.4. Ejercicios para resolver. Casos prácticos

Diseñe y desarrolle un programa para la empresa Helados Reynosa, que el usuario escoja un tipo de agua de frutas, cuántos litros de agua requiere e imprima el costo total a pagar. Los datos se deben de capturar desde el teclado.

Nota: Si escogió más de 5 litros de agua se le descontará 20% del costo total.

1. Agua de sandía 20 pesos litro.
2. Agua de horchata 25 pesos cada litro.
3. Agua de limón 10 pesos el medio litro.
4. Agua de Jamaica 15 pesos litro.
5. Agua de melón 30 pesos el litro.

Diseñe y desarrolle un programa cuya entrada de teclado sean dos datos y que imprima cuál es el mayor de esos dos números.

Diseñe y desarrolle un programa que lea tres números capturados desde el teclado y que imprima cuál es el mayor de esos tres números, hágalo con menos líneas posibles.

Diseñe y desarrolle un programa que lea un número capturado desde el teclado e imprima si es múltiplo de 25.

Diseñe y desarrolle un programa cuyo dato leído del teclado indague si es par o es impar.

Diseñe y desarrolle un programa que obtenga si el número que se va a capturar es par o es impar.

Diseñe y desarrolle un programa para la Farmacia Villanueva que capture los siguientes artículos de inventario mediante el teclado:

30 jeringas a 5 pesos cada una, 20 frascos de jarabe a 58 pesos cada una, 4 vendas a 15 cada una, 12 cajas de pastillas para el dolor de cabeza a 90 pesos cada una. Obtener la utilidad total de cada artículo si al vender más de 10 jeringas se descuenta 10% del costo total y si el comprador adquiere más de 2 jarabes se le descontará el 15% del costo total de jarabes.

Diseñe y desarrolle un programa para la tienda Tigres de playeras deportivas

Esta tienda otorga un descuento de acuerdo con su tarjeta de crédito por cada compra. Si es tarjeta BANAMEX y compra menos de 100 pesos se le otorgará un descuento del 5%. Si es tarjeta INBURSA y compra más de 100 pero menos de 200 se le dará un descuento del 10% de su compra total, pero si compra más de 300 y su tarjeta es Santander se le otorgará un 30% de descuento. Los datos serán capturados mediante el teclado.

Diseñe y desarrolle un programa para la liga independiente de *softball* llamada "50 y más". En la final se encuentran dos equipos con los siguientes resultados.

Juegos	Equipo Los Románticos	Equipo Los Filósofos
Juego 1	5	8
Juego 2	3	2
Juego 3	5	5
Juego 4	3	6

Obtener lo siguiente:

- a. Qué equipo ganó la final de *softball*.
- b. Qué equipo ganó en cada juego y en qué juego ganó el campeonato.
- c. En cuántos juegos obtuvo el campeonato el equipo ganador.
- d. Cuántos empates hubo.
- e.Cuál fue el promedio de carreras de cada equipo.

Diseñe y desarrolle un programa para la escuela preparatoria "José de Escandón" que tiene N aspirantes y los siguientes requisitos:

Si son hombres que tengan entre 21-24 años inclusive y que hayan obtenido el promedio mayor a 8.

Si son mujeres que tengan entre 18 y 21 años y que hayan obtenido un promedio de 8.

Para ser aceptado se necesita saber lo siguiente:

- a. Cuántos son hombres y tienen entre 21 y 24 además que su promedio sea mayor o igual a 8.
- b. Cuántas son mujeres y tienen edades entre 18 y 21.
- c. Obtener el promedio de edad de las mujeres y de los hombres.
- d.Cuál es el promedio de los aceptados con respecto a los no aceptados.

## 2.5. Referencias de consulta

<http://www.edu4java.com/es/progbasica/progbasica5.html>

<https://programandointentandolo.com/2017/07/estructuras-condicionales-java.html>

<https://javautodidacta.es/if-else-en-java/>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/while>

## 2.6. Videos de consulta

<https://www.youtube.com/watch?v=QgndTPSjcRo>

<https://www.youtube.com/watch?v=jGfB6UzG-6A><https://www.youtube.com/watch?v=gL4ZQj6MEic>





# Capítulo 3

## Ciclos *for*

### 3.1. Objetivo del capítulo

El objetivo es que el alumno aprenda a ejecutar e iterar con el comando *for* las instrucciones contenidas en un rango y practique el control sobre el ciclo, mediante una variable de control.

### 3.2. Introducción

#### Aprendizaje esperado

Tiene como propósito que el lector identifique y utilice una técnica para la solución de un problema empleando distintas estructuras de control repetitivas como *for*, *while*, *do while*.

#### Definición del ciclo *for*

*For*: es un estatuto que sirve para implementar un ciclo en un conjunto de instrucciones o rango de código definido mediante una variable de control a través de un valor de inicio, un tamaño de paso y un valor límite para el ciclo (González, 2018).

Esta estructura facilita que la acción se repita un número de veces (Gottfried, 1994).

Según la definición de González (2018) un ciclo *for* ejecuta una cierta cantidad de veces la instrucción colocada en el ciclo y una vez cumplida sale de éste.

Los siguientes ejemplos ilustran el manejo del ciclo *for* con el lenguaje de programación Java.

### 3.3. Ejemplos prácticos realizados con el comando *for*

#### Laboratorio 22. Desarrollo de programa que calcule números impares

Diseñar y desarrollar un programa que obtenga el producto de los números impares entre 1 y 15.

## Codificación

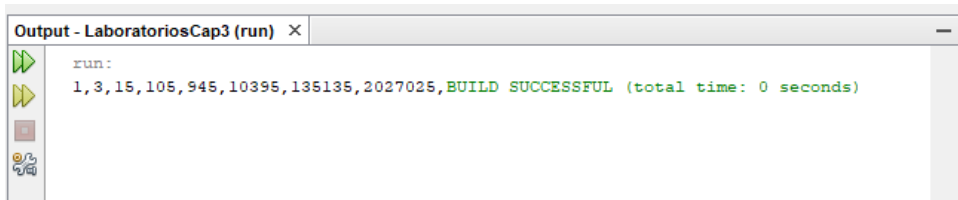
```
package práctica14;
/**
 *
 * @author jxs_b
 */
public class Práctica14 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int producto22=1;
        int a22;
        for (a22=1; a22<=16; a22++)
        {
            if (a22%2 ==1)
            {

                producto22=producto22*a22;
                System.out.printf("%d,", producto22);
            }
        }
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

### Figura 22

*Ejecución del Laboratorio 22*



```
Output - LaboratoriosCap3 (run) ×
run:
1, 3, 15, 105, 945, 10395, 135135, 2027025, BUILD SUCCESSFUL (total time: 0 seconds)
```

## Laboratorio 23. Desarrollo de programa para imprimir el número de caracteres recibido mediante un número teclado

Diseñar y desarrollar un programa que lea 3 números cada uno entre 1 y 30. Para un dato recibido, se deberán desplegar los asteriscos que correspondan al valor del dato (Deitel, 2008). Una aplicación interesante de la computadora es dibujar gráficos convencionales y de barra.

### Codificación

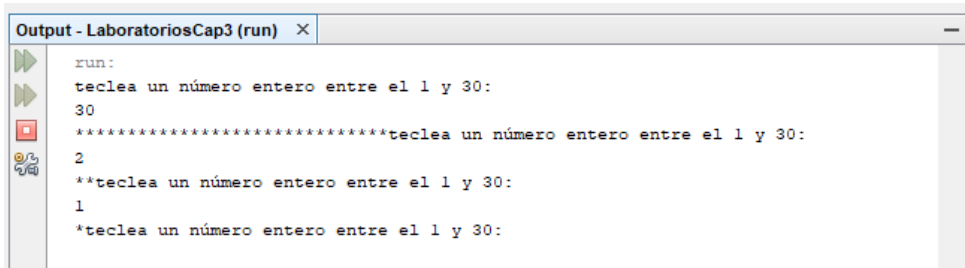
```
package ejercicio63;}
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio63 {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada23 = new Scanner (System.in);
        int x23;
        for (int a23=1; a23<=5; a23++) {
            System.out.println("teclea un número entero entre el 1 y 30:");
            x23= entrada23. nextInt ();
            for (int e23 =1; e23<=x; e23++) {
                System.out.print("*");
            }
        }
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

Figura 23

Ejecución del Laboratorio 23



```
run:
tecllea un número entero entre el 1 y 30:
30
*****tecllea un número entero entre el 1 y 30:
2
**tecllea un número entero entre el 1 y 30:
1
*tecllea un número entero entre el 1 y 30:
```

Laboratorio 24. Desarrollo de programa que multiplique números impares  
Diseñar y desarrollar un programa que multiplique todos los números impares del 1 al 16.

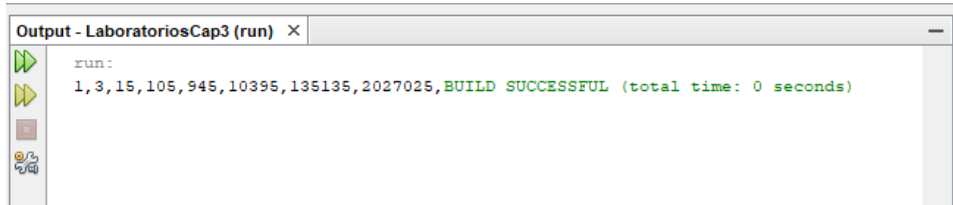
## Codificación

```
package práctica14;
/**
 *
 * @author jxs_b
 */
public class Práctica14 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int producto24=1;
        int a24;
        for (a24=1; a24<=16; a24++)
        {
            if (a24%2 ==1)
            {
                producto24=producto24*a24;
                System.out.printf("%d,", producto24);
            }
        }
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

### Figura 24

Ejecución del Laboratorio 24



## Laboratorio 25. Desarrollo de programa que utilice un contador de *for*

Diseñar y desarrollar un programa que utilice un contador de *for* del 1 al 10 con incrementos en 1 utilizando la secuencia de escape de tabulación.

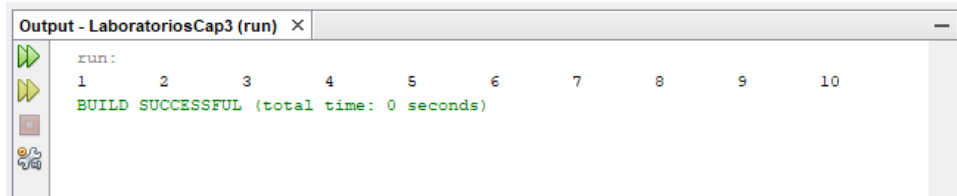
## Codificación

```
package contador25for;
/**
 *
 * @author jxs_b
 */
public class Contador25For {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        for (int contador25 =1; contador25 <=10; contador25++)
        {
            System.out.printf("%d\t", contador25);
        }
        System.out.println();
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

**Figura 25**

*Ejecución del Laboratorio 25*



Laboratorio 26. Diseñar un programa para desplegar las tablas de multiplicar

Desarrollar un programa que despliegue las tablas de multiplicar del 1 al 10 utilizando un mensaje antes de imprimir “la tabla del 1” y así sucesivamente.

## Codificación

```
package ejercicio57;
/**
 *
 * @author jxs_b
 */
public class Ejercicio57 {

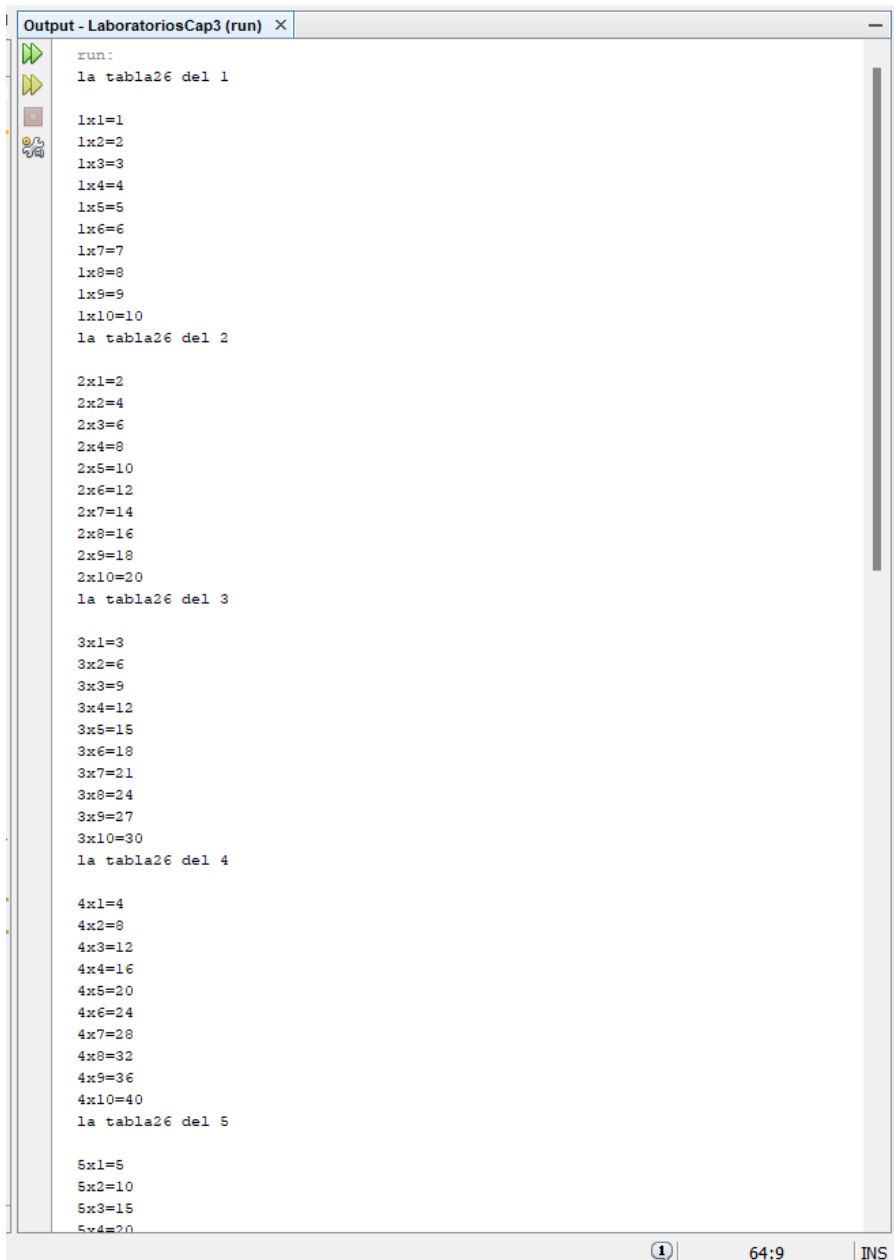
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        for (int tabla26=1; tabla26<=10; tabla26++)
        {
            System.out.printf("la tabla26 del %d\n\n", tabla26);

            for (int numero26=1; numero26<=10; numero26++)
            {
                System.out.println(tabla26+"x"+numero26+"="+tabla26*numero26);
                // Se aplica el código lógico aquí:
            }
        }
    }
}
```

## Ejecución

Figura 26

*Ejecución del Laboratorio 26*



```
run:
la tabla26 del 1

1x1=1
1x2=2
1x3=3
1x4=4
1x5=5
1x6=6
1x7=7
1x8=8
1x9=9
1x10=10
la tabla26 del 2

2x1=2
2x2=4
2x3=6
2x4=8
2x5=10
2x6=12
2x7=14
2x8=16
2x9=18
2x10=20
la tabla26 del 3

3x1=3
3x2=6
3x3=9
3x4=12
3x5=15
3x6=18
3x7=21
3x8=24
3x9=27
3x10=30
la tabla26 del 4

4x1=4
4x2=8
4x3=12
4x4=16
4x5=20
4x6=24
4x7=28
4x8=32
4x9=36
4x10=40
la tabla26 del 5

5x1=5
5x2=10
5x3=15
5x4=20
```

## Laboratorio 27. Desarrollo de programa utilizando el ciclo for

Diseñar y desarrollar un programa que dibuje la siguiente figura con asteriscos.  
Utilizar el ciclo *for*.

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

### Codificación

```
package ejercicio58;
/**
 *
 * @author jxs_b
 */
public class Ejercicio58 {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {

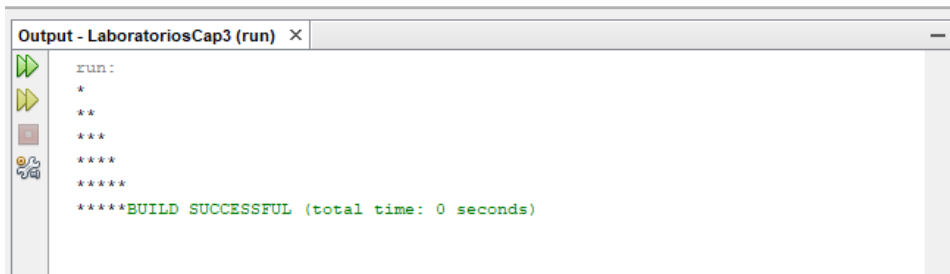
        for (int i27=1; i27<=5; i27++)
        {
            System.out.println("*");
            for (int j27=1; j27<=i; j27++)
            {
                System.out.print("*");
            }
        }
        // Se aplica el código lógico aquí:
    }
}
```



## Ejecución

Figura 27

Ejecución del Laboratorio 27



Laboratorio 28. Desarrollo de programa que genere figura utilizando *for*.  
Diseñar y desarrollar un programa que realice la siguiente figura con asteriscos (utilice *for*).

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

## Codificación

```
package ejercicio61;
/**
 *
 * @author jxs_b
 */
public class Ejercicio61 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int numb28=4;
        for (int i28=1; i28<=5; i28++)
        {
            for (int j28 =1; j28<=numb; j28++)
            {
                System.out.print(" ");
```

```

    }
    for (int j28 =1; j28<=i; j28++)
    {
        System.out.print("*");
    }

    numb28=1;
    System.out.println();
}
// Se aplica el código lógico aquí:

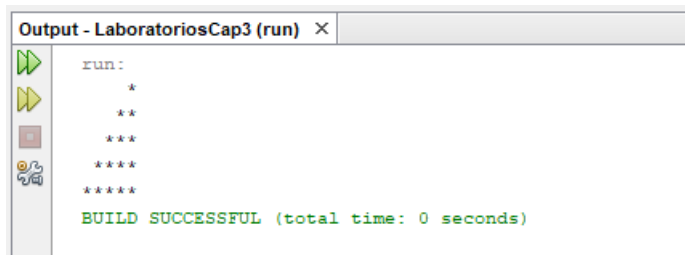
}
}

```

## Ejecución

**Figura 28**

*Ejecución del Laboratorio 28*



## Laboratorio 29. Desarrollo de programa que desarrolle un triángulo de asteriscos

Diseñar y desarrollar un triángulo que realice la siguiente figura con asteriscos:

```

    *
   ***
  *****
 *****
*****

```

## Codificación

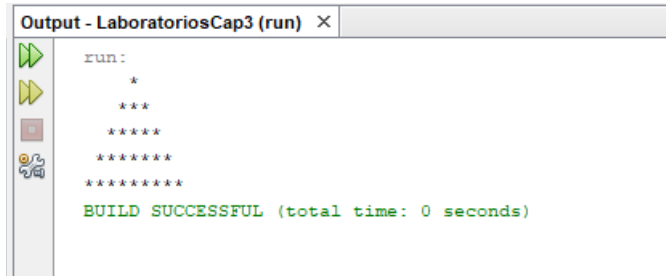
```
package ejercicio60;
/**
 *
 * @author jxs_b
 */
public class Ejercicio60 {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        for (int altura29 = 1; altura29<=5; altura29++) {
            //Espacios en blanco
            for (int blancos29 = 1; blancos29<=5-altura29; blancos29++) {
                System.out.print(" ");
            }
            //Asteriscos29
            for (int asteriscos29=1; asteriscos29<=(altura29*2)-1; asteriscos29++) {
                System.out.print("*");
            }
            System.out.println();
        }
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

Figura 29

Ejecución del Laboratorio 29



Laboratorio 30. Desarrollo de programa que forme cierta figura con asteriscos

Diseñar y desarrollar un programa que entregue la siguiente figura:

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

## Codificación

```
package ejercicio59;
/**
 *
 * @author jxs_b
 */
public class Ejercicio59 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        for (int x30=5; x30>0; x30--)
        {
            for (int j30=1; j30<x+1; j30++)
            {
                System.out.print("*");
            }
        }
    }
}
```

```

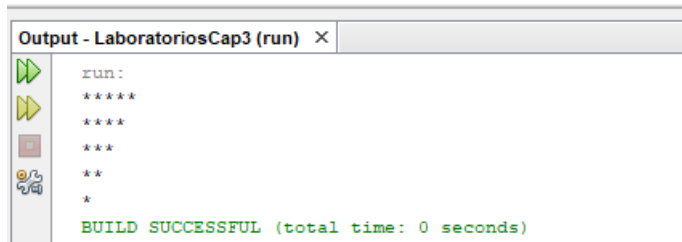
        System.out.println(" ");
    }
}
}

```

## Ejecución

**Figura 30**

*Ejecución del Laboratorio 30*



Laboratorio 31. Desarrollo de programa que componga el reflejo de la figura anterior con asteriscos

Diseñar y desarrollar un programa que entregue la siguiente figura:

```

*****
*****
***
**
*

```

## Codificación

```

package ejercicio62;
/**
 *
 * @author jxs_b
 */
public class Ejercicio62 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int numb31=0;
        for (int x31=5; x31>0; x31--)

```

```

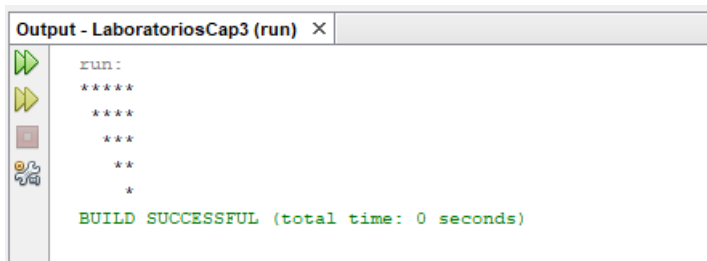
    {
    for (int j31=1; j31<=numb; j31++)
    {
    System.out.print(" ");
    }
    for (int j31=1; j31<=x; j31++) {
    System.out.print("*");
    }
    System.out.println();
    Numb31+=1;
    }
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

**Figura 31**

*Ejecución del Laboratorio 31*



## Laboratorio 32. Desarrollo de programa a partir de datos interactivos que muestre asteriscos

Crear un programa que dibuje gráficos convencionales, y que pida al usuario 5 números cada uno entre 1 y 30. Por cada uno el programa desplegará un igual número de asteriscos (Deitel, 2008).

## Codificación

```
package ejercicio 63;
```

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

```
/**
```

```
*
```

```
* @author jxs_b
```

```
*/
```

```
public class Ejercicio63 {
```

```
/**
```

```
* @param args instrucción para el traspaso de argumentos
```

```
*/
```

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

```
    Scanner entrada32 = new Scanner (System.in);
```

```
    int x32;
```

```
    for (int a32=1; a32<=5; a32++) {
```

```
        System.out.println("teclea un número entero entre el 1 y 30:");
```

```
        x = entrada32. nextInt ();
```

```
        for (int e32 =1; e32<=x; e32++) {
```

```
            System.out.print("*");
```

```
        }
```

```
    }
```

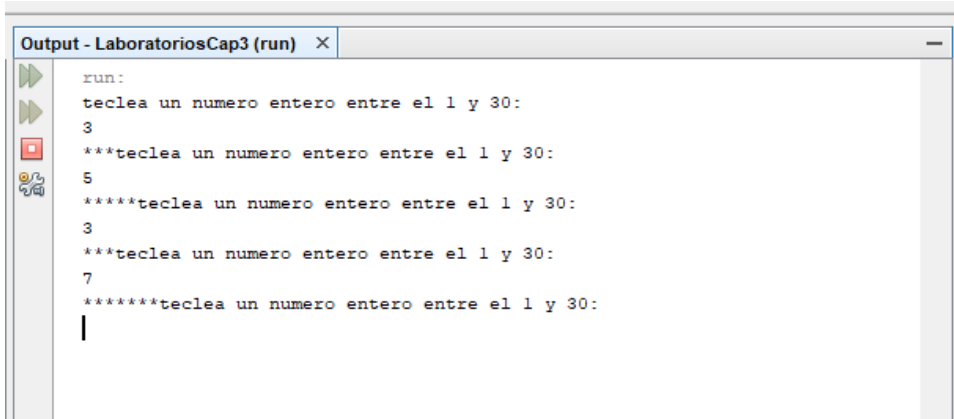
```
    // Se aplica el código lógico aquí:
```

```
}
```

## Ejecución

Figura 32

Laboratorio 32 desarrollado



```
Output - LaboratoriosCap3 (run) X
run:
teclea un numero entero entre el 1 y 30:
3
***teclea un numero entero entre el 1 y 30:
5
*****teclea un numero entero entre el 1 y 30:
3
***teclea un numero entero entre el 1 y 30:
7
*****teclea un numero entero entre el 1 y 30:
|
```

### 3.4. Ejercicios para resolver. Casos prácticos

Codificar un programa que reciba por teclado 15 números e imprima (o despliegue) la suma.

Diseñar y desarrollar un programa para descubrir qué se imprime con la siguiente línea:

```
for (i34 = 0; i34 <4; i34++) {
for (j34=0; j34>0; j34+ +) {
suma = i 34* 10 + 3;
System.out.println(suma);
}}
```

Diseñar un programa que imprima los siguientes triángulos.

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



Implemente un programa que dibuje por medio de asteriscos un rombo. Utilice *for*.

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

Implemente un programa que lea un número entre 1 y 19 que determina el número de filas en el rombo. Utilice ciclos *for*.

Diseñar y desarrollar un programa que despliega un cuadrado utilizando N elementos utilice ciclos *for* "\*" (Deitel, 2008).

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

Diseñe un programa que imprima números del 1 al 20. Utilice ciclos *for*.

Realice un programa con un contador que imprima los múltiplos del 2 terminando en el 1. Utilice ciclos *for*.

Implemente un programa que lea N números dados mediante el teclado y que los sume. Utilice ciclos `for`.

Realice un programa que calcule la suma de los números nones del 1 al 10. Utilice ciclos `for`.

### 3.5. Referencias de consulta

<https://www.programarya.com/Cursos/Java/Ciclos/Ciclo-for>

<https://decodigo.com/java-el-ciclo-foR>

<https://mariocelis.com/java/for/>

### 3.6. Videos de consulta del ciclo `for` en Java

[https://www.youtube.com/watch?v=hWyucXE\\_c-k](https://www.youtube.com/watch?v=hWyucXE_c-k)

<https://www.youtube.com/watch?v=XW7LS1zwTME>

[https://www.youtube.com/watch?v=V1Wgi\\_FschM](https://www.youtube.com/watch?v=V1Wgi_FschM)

# Capítulo 4

## Ciclos *do-while*

### 4.1. Objetivo del capítulo

Que el alumno aprenda formas alternativas de aplicar estructuras repetitivas básicas con el ciclo *do-while*.

### 4.2. Introducción

*Do while* es una estructura repetitiva, para ejecutar al menos una vez su bloque de instrucciones, en contraste con el *while* o del *for* que pueden no ejecutar el bloque de instrucciones porque la condición aparece al principio. Esta estructura repetitiva se utiliza cuando conocemos de antemano que entra al cuerpo del ciclo a ejecutar los estatutos que están dentro por lo menos una vez (Programación Ya, s.f.)

Se establece en la obra de (Dean, 2019) que un ciclo ejecuta una sentencia (o conjunto de instrucciones), hasta que la condición de comprobación válida se evalúa como falsa. La condición se evalúa antes de ejecutar uno o un rango de estatutos, lo que puede producir que el ciclo mencionado se ejecute una sola vez o ninguna.

### 4.3. Ejemplos prácticos realizados, utilizando el comando de repetición *do-while*

Laboratorio 33. Desarrollo de programa para práctica del incremento de índice mediante estatuto *while*

Implementa un programa que despliegue del 1 al 10 con incremento en 1, utilizando un contador.

#### Codificación

```
package prueba dowhile;
/**
 *
 * @author jxs_b
 */
public class PruebaDoWhile {
```

```

/**
 * @param args instrucción para el traspaso de argumentos
 */
public static void main (String [] args) {
    int contador33 = 1;
    do
    {
        System.out.printf("%d\t", contador33);
        ++contador33;
    } while(contador33<=10);
    System.out.println();

    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 33

*Ejecución del Laboratorio 33*

The screenshot shows an IDE output window titled "Output - LaboratoriosCap4 (run)". The output text is as follows:

```

run:
1      2      3      4      5      6      7      8      9      10
BUILD SUCCESSFUL (total time: 0 seconds)

```

## Laboratorio 34. Implementación de programa de práctica de la condición de salida del estatuto *while*

Diseñar y desarrollar un programa que lea un número desde el teclado y despliegue si ese número es positivo o negativo. El proceso se repite hasta que se cumpla con la condición.

### Codificación

```
package ejercicio56;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio56 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada34 = new Scanner (System.in);
        int numero34;
        do
        {
            System.out.print("Teclea un número");
            numero34 = entrada34. nextInt ();
            if(numero34<0) {
                System.out.println("El número es negativo");
            }
            if(numero34>0)
                System.out.println("El número es positivo");
        } while (numero34!= 0);
        System.out.println();
    }
    // Se aplica el código lógico aquí:
}
```

## Ejecución

Figura 34

Ejecución del Laboratorio 34



```
Output - LaboratoriosCap4 (run) ×
run:
Teclea un numero4
El número es positivo
Teclea un numero-3
El número es negativo
Teclea un numero2
El número es positivo
Teclea un numero-5
El número es negativo
Teclea un numero
```

Laboratorio 35. Creación de programa que imprime datos según condición del estatuto *while*

Desarrollar un programa con un contador que imprima la tabla del 12 del mayor al menor.

## Codificación

```
package ejemplo1dowhile;
/**
 *
 * @author jxs_b
 */
public class Ejemplo1dowhile {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador35 = 120;
        do
        {
            System.out.printf("%d\t", contador35);
            Contador35-=12;
        } while(contador35>=12);
        System.out.println();
    }
}
```

```

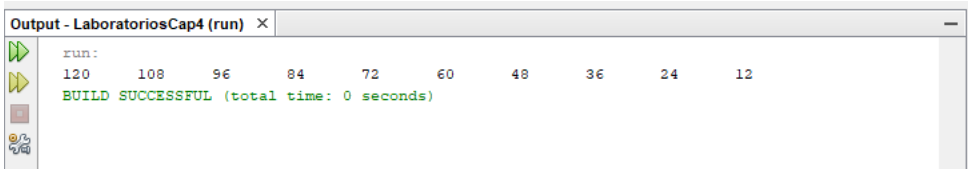
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

**Figura 35**

*Ejecución del Laboratorio 35*



Laboratorio 36. Desarrollo de programa que imprime datos controlados por la condición del estatuto *while*

Desarrollar un programa que use un contador para imprimir los múltiplos del número 15 del mayor hasta llegar al 15.

## Codificación

```

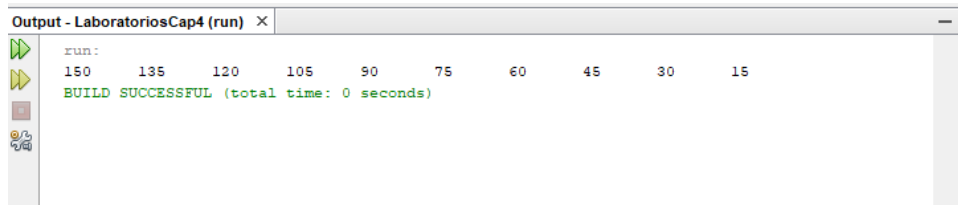
package ejemplo2dowhile;
/**
 *
 * @author jxs_b
 */
public class Ejemplo2dowhile {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador36 = 150;
        do
        {
            System.out.printf("%d\t", contador36);
            Contador36-=15;
        } while(contador36>=15);
        System.out.println();
        // Se aplica el código lógico aquí:
    }
}

```

## Ejecución:

### Figura 36

*Ejecución del Laboratorio 36*



```
Output - LaboratoriosCap4 (run) x
run:
150    135    120    105    90    75    60    45    30    15
BUILD SUCCESSFUL (total time: 0 seconds)
```

Laboratorio 37. Desarrollo de un programa para que a partir de una cantidad recibida esta sea decrementada en 100 hasta alcanzar dicho valor con el apoyo del estatuto `while`

Hacer un programa que decremente el número 10 000 de 100 en 100 hasta llegar al 100 e imprimirlos de mayor a menor.

## Codificación

```
package ejemplo3dowhile;
/**
 *
 * @author jxs_b
 */
public class Ejemplo3dowhile {

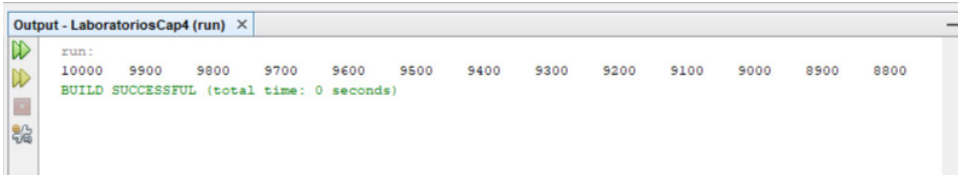
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador37 = 10000;
        do
        {
            System.out.printf("%d\t", contador37);
            Contador37-=100;
        } while(contador37>=100);
        System.out.println();
        // TODO Se aplica el código lógico aquí:
    }
}
```



## Ejecución

Figura 37

Ejecución del Laboratorio 37



```
Output - LaboratoriosCap4 (run) x
run:
10000 9900 9800 9700 9600 9500 9400 9300 9200 9100 9000 8900 8800
BUILD SUCCESSFUL (total time: 0 seconds)
```

### 4.4. Ejercicios para resolver. Casos prácticos

Implemente un programa que imprima los números del 1 al 10. Utilice el ciclo *do-while*.

Hacer un programa que imprima los números del 1 al 100, pero solamente los números pares.

Desarrollar un programa que imprima los números del 100 al 1, pero solamente los números impares.

Hacer un programa que lea mediante el teclado los números del 1 al 50 e imprimir cuántos y cuáles son los números pares.

Llevar a cabo un programa que obtenga el promedio de N números leídos.

Implementar un programa que sume los números de 10 en 10 hasta el 120, que hay entre 1 a 120.

Crear programa que imprima los números comprendidos hasta el 1000 y señale cuántos son impares (Gabriela Márquez, 2011).

Desarrollar un programa que sume los pares e impares de los números comprendidos entre 100 y 200.

Hacer un programa que sume los números primos del número 1000.

Implementar un programa que lea 3 números e imprimir los múltiplos del número mayor dado mediante el teclado por el usuario.

## 4.5. Referencias de consulta

<https://javadesdecero.es/basico/bucles-for-while-do-while-ejemplos/>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Statements/do...while>

<https://www.tutorialesprogramacionya.com/javaya/detalleconcepto.php?codigo=85>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/while>

## 4.6. Videos de apoyo

<https://www.youtube.com/watch?v=dVCTDygWFIE>

[https://www.youtube.com/watch?v=8f\\_nVq7azoI](https://www.youtube.com/watch?v=8f_nVq7azoI)

<https://www.youtube.com/watch?v=bGXSx2c4QBA>

# Capítulo 5

## Ciclos *while*

### 5.1. Objetivo del capítulo

Que el alumno comprenda otra alternativa de repetición sin necesidad de tener un valor inicial numérica como el ciclo *for* ni saber cuándo se va a dar el valor final, además podrá manejar diferentes tipos de valores tales como booleanos que se aplican en este bucle.

### 5.2. Introducción

El ciclo *while* crea un bucle que ejecuta una instrucción o varias mientras se ejecuta la condición, si la condición es verdadera, se ejecuta, pero si la condición es evaluada como falsa se sale del ciclo, o sea ni entra al cuerpo del ciclo.

### 5.3. Ejemplos prácticos realizados utilizando ciclo *while*

Laboratorio 38. Desarrollo de programa de práctica de control del estatuto *while*

Hacer un programa que usa un contador con *while* del 1 al 10 con incremento al 1.

#### Codificación

```
package contadorwhile;
/**
 *
 * @author jxs_b
 */
public class ContadorWhile {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador38=1;
        while (contador38<=10)
```

```

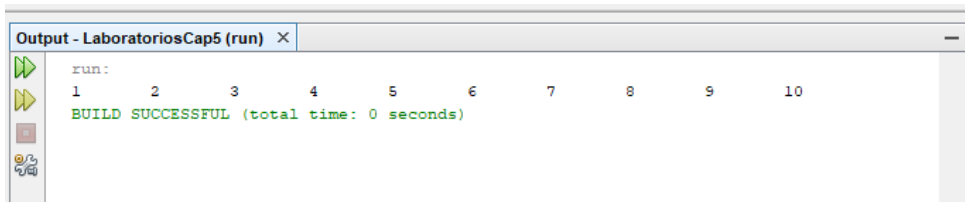
    {
        System.out.printf("%d\t", contador38);
        ++contador38;
    }
    System.out.println();
    // TODO Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 38

*Ejecución del Laboratorio 38*



Laboratorio 39. Desarrollo de programa que imprima valores seleccionados mediante estatuto *while*

Desarrollar un programa para imprimir los múltiplos de 20 menores a 100.

## Codificación

```

package ejemplo51;
/**
 *
 * @author jxs_b
 */
public class Ejemplo51 {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador39=20;

```

```

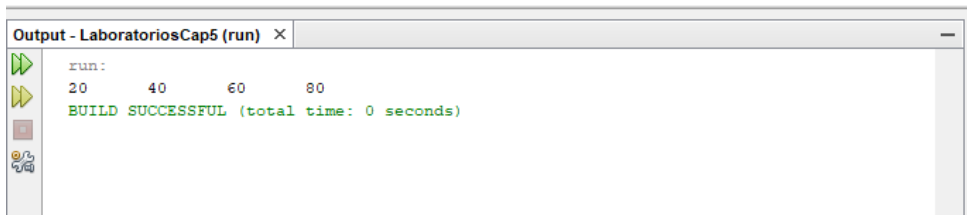
while (contador39 <100)
{
    System.out.printf("%d\t", contador39);
    Contador39+=20;
}
System.out.println();
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 39

*Ejecución del Laboratorio 39*



Laboratorio 40. Desarrollo de programa para práctica del valor del incremento en el estatuto *while*

Crear un programa que incremente en 2 e imprima los números de 2 en 2 hasta el 20.

## Codificación

```

package ejemplo1while;
/**
 *
 * @author jxs_b
 */
public class Ejemplo1while {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador40=2;

```

```

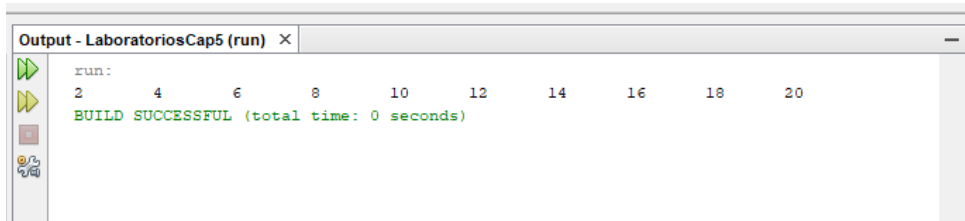
while (contador40 <=20)
{
    System.out.printf("%d\t", contador40);
    Contador40+=2;
}
System.out.println();
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 40

*Ejecución del Laboratorio 40*



```

Output - LaboratoriosCap5 (run) x
run:
2      4      6      8      10     12     14     16     18     20
BUILD SUCCESSFUL (total time: 0 seconds)

```

Laboratorio 41. Desarrollo de programa para práctica del valor del incremento en el estatuto *while*

Llevar a cabo una aplicación que muestre los múltiplos de 4.

## Codificación

```

package ejemplo2while;
/**
 *
 * @author jxs_b
 */
public class Ejemplo2while {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador41=4;
        while (contador41 <=40)
        {

```

```

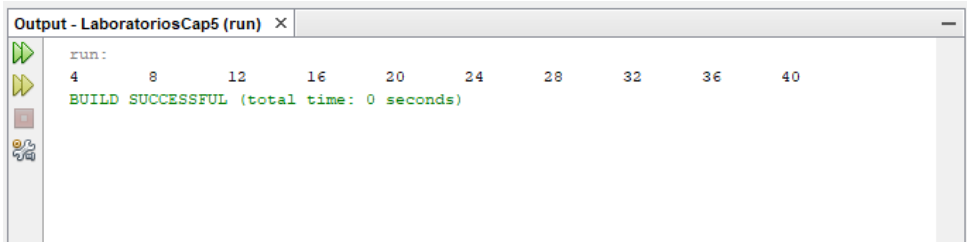
        System.out.printf("%d\t", contador41);
        Contador41+=4;
    }
    System.out.println();
    // Se aplica el código lógico aquí:
}
}
}

```

## Ejecución

### Figura 41

*Ejecución del Laboratorio 41*



```

Output - LaboratoriosCap5 (run) ×
run:
4      8      12     16     20     24     28     32     36     40
BUILD SUCCESSFUL (total time: 0 seconds)

```

Laboratorio 42. Desarrollo de programa para práctica del valor del incremento (6), n el estatuto *while*

Desarrollar un programa que imprima los múltiplos del número 6 hasta el número 60.

## Codificación

```

package ejemplo3while;
/**
 *
 * @author jxs_b
 */
public class Ejemplo3while {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int contador42=6;

        while (contador42 <=60)
        {

```

```

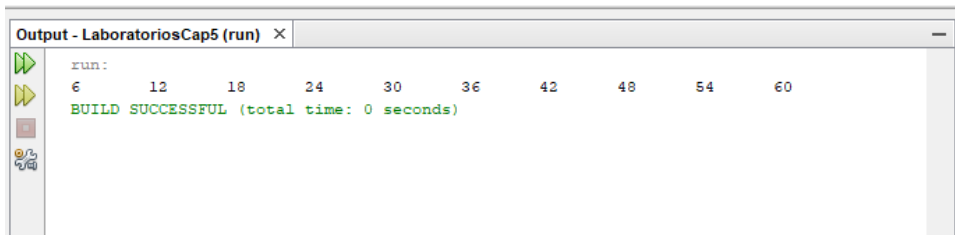
        System.out.printf("%d\t", contador42);
        Contador42+=6;
    }
    System.out.println();
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

**Figura 42**

*Ejecución del Laboratorio 42*



## Laboratorio 43. Desarrollo de programa para práctica del valor del incremento en el estatuto *while* de forma descendente

Hacer un programa con contador para poder imprimir los múltiplos del número 8 hasta el número 80 pero de forma descendente.

## Codificación

```

package ejemplo4while;
/**
 *
 * @author jxs_b
 */
public class Ejemplo4while {
    /**
     * @param args instrucción para el traspaso de argumentos */
    public static void main (String [] args) {
        int contador43=80;
        while (contador43 >=8)
        {
            System.out.printf("%d\t", contador43);

```



```

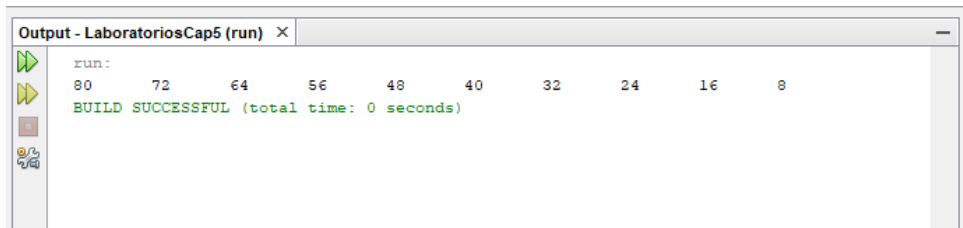
        Contador43-=8;
    }
    System.out.println();
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

**Figura 43**

*Ejecución del Laboratorio 43*



**Laboratorio 44. Desarrollo de programa para práctica del valor del decremento en el estatuto *while***

Implementar un programa que utilice un contador e imprima en forma descendente del número 100 al número 10.

## Codificación

```

package ejemplo5while;
/**
 *
 * @author jxs_b
 */
public class Ejemplo5while {
    /**
     * @param args instrucción para el traspaso de argumentos */
    public static void main (String [] args) {
        int contador44=100;
        while (contador44 >=10)
        {
            System.out.printf("%d\t", contador44);
            Contador44-=10;
        }
    }
}

```

```

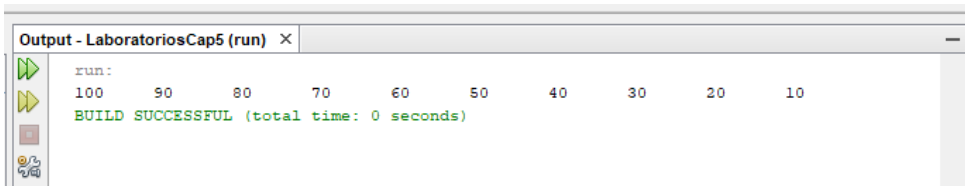
    System.out.println();
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 44

*Ejecución del Laboratorio 44*



## 5.4. Ejercicios para resolver utilizando ciclo while. Casos prácticos

Desarrollar un programa utilizando ciclos *while* que imprima el mensaje “Bravos” mientras el usuario presiona la tecla b o B.

Desarrollar un programa utilizando ciclos *while*, que muestre la conversión a un ciclo infinito.

Analice y desarrolle un programa que capture 10 números y que imprima el total de la suma cuando se le dé un número negativo. Utilice ciclos *while*.

Desarrolle un programa que imprima “Bienvenidos a tu escuela” mientras se teclee un número negativo. Utilice ciclos *while*.

Desarrolle un programa que imprima todos los números del 1 a n, clasificándolos en pares y nones mientras teclees el número 5. Utilice el ciclo *while*.

Desarrolle un programa que calcule la suma de las edades de los N empleados del departamento de producción de la maquiladora LGH hasta que la suma sea mayor 200. Utilice la estructura interactiva *while*.

Desarrolle un programa que convierta kilómetros a millas, dando desde la pantalla el número de kilómetros que desea convertir. Utilice el ciclo *while*.

## 5.5. Referencias de consulta

<https://decodigo.com/java-el-ciclo-while>

<https://decodigo.com/java-el-ciclo-while>

<https://byspel.com/ciclo-while-en-java/>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/while>

## 5.6. Video de apoyo

<https://www.youtube.com/watch?v=5-bvaMv5AnM>

<https://www.youtube.com/watch?v=ypNc3Eok69o>

[https://www.youtube.com/watch?v=eqP\\_Q7cgwJc](https://www.youtube.com/watch?v=eqP_Q7cgwJc)



# Capítulo 6

## Instrucción Selección Múltiple Switch

### 6.1. Objetivo del capítulo

El alumno aprenderá a manejar y a seleccionar dentro de una estructura de control condicional un grupo particular de sentencias entre varias alternativas posibles por medio de un selector.

### 6.2. Introducción

Definición del *switch*:

La instrucción *switch* toma la variable que le pasamos como argumento y obtiene su valor. Después lo compara con los valores que hay junto a las etiquetas *case*, comenzando a ejecutar el código que hay debajo de la etiqueta *case* que coincida. Cuando se encuentre con la instrucción *break* finalizará la ejecución del *switch*. En caso de no encontrar una etiqueta *case* coincidente, ejecutará el código que existe dentro de la etiqueta *default*. La etiqueta *default* es opcional, por tanto, en caso de no tenerla simplemente no hará nada (Ceballos, 2008).

Ceballos (2008) define *switch* como un menú que empieza con un nombre *case* y termina la instrucción de este con un *break*.

A continuación, se presenta una serie de ejemplos utilizando el selector *switch*, con el lenguaje de programación orientado a objetos llamado Java.

### 6.3. Ejemplos prácticos realizados utilizando ciclo switch

Laboratorio 45. Desarrollo de programa que aplica el estatuto *switch* para seleccionar rutinas a ejecutar

Desarrollar un programa que crea un menú utilizando el selector *switch* con las siguientes opciones: área del triángulo, área de un círculo y área de un cuadrado. Obteniendo diferentes resultados de acuerdo con la opción elegida.

## Codificación

```
package pkgswitch;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Switch {
/**
 * @param args instrucción para el traspaso de argumentos
 */
public static void main (String [] args) {
    Scanner entrada45 = new Scanner (System.in);
    int a45;
    int b45;
    int l45;
    int opcion45;
    float area45;
    int r45;
    System.out.println("1.Área del triángulo ");
    System.out.println("2.Área del cuadrado ");
    System.out.println("3.Área del círculo ");
    System.out.println("Qué opción deseas: ");
    opcion = entrada45. nextInt ();
    switch (opcion)
    {
        case 1:
            System.out.println("teclea la base");
            a45 = entrada. nextInt ();
            System.out.println("teclea la altura");
            b45 = entrada. nextInt ();
            area45 = (a45 * b45) /2;
            System.out.println("El resultado es: "+area45);

            break;
        case 2:
            System.out.println("teclea el lado");
            l45 = entrada45. nextInt ();
```

```

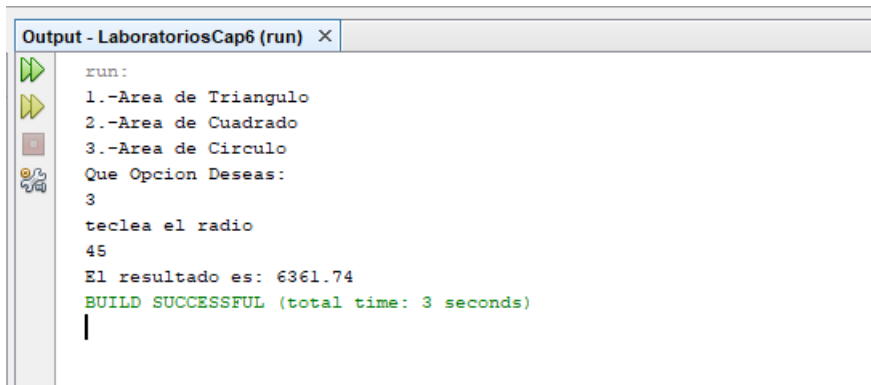
        Area45 = l45 * l45;
        System.out.println("El resultado es: "+area45);
        break;
    case 3:
        System.out.println("teclea el radio");
        r45 = entrada45.nextInt ();
        area45 = (float) (3.1416* r45 * r45);
        System.out.println("El resultado es: "+area45);
        break;
    default:
        System.out.print("OPCIÓN NO VÁLIDA");
        break;
    }
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

**Figura 45**

*Ejecución del Laboratorio 45*



```

Output - LaboratoriosCap6 (run) X
run:
1.-Area de Triangulo
2.-Area de Cuadrado
3.-Area de Circulo
Que Opcion Deseas:
3
teclea el radio
45
El resultado es: 6361.74
BUILD SUCCESSFUL (total time: 3 seconds)
|

```

## Laboratorio 46. Desarrollo de programa que aplica el estatuto *switch* para seleccionar operaciones matemáticas a ejecutar

Diseñar y desarrollar un programa utilizando la instrucción de selección múltiple *switch* que despliegue un menú y que elija una opción mediante el teclado de las operaciones suma, resta, multiplicación y división de 2 enteros.

## Codificación

```
package suma. pkgswitch;
import java. util. Scanner;
/**
 *
 * @author jxs_b
 */
public class SumaSwitch {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada46 = new Scanner (System.in);
        int num146;
        int num246;
        int opcion46;
        float respuesta46;

        System.out.println("1.suma ");
        System.out.println("2.resta ");
        System.out.println("3.multiplicacion ");
        System.out.println("4.division ");
        System.out.println("Qué opción deseas: ");
        opcion46 = entrada46. nextInt ();
        switch (opcion46)
        {
            case 1:
                System.out.println("teclea número 1 ");
                num146 = entrada46. nextInt ();
                System.out.println("teclea número 2");
                num246 = entrada46. nextInt ();
                respuesta46 = num146+num246;
                System.out.println("El resultado es: "+respuesta46);
                break;
            case 2:
                System.out.println("teclea número 1 ");
                num146 = entrada46. nextInt ();
```



```

        System.out.println("teclea número 2");
        num246 = entrada46. nextInt ();
        respuesta46 = num146-num246;
        System.out.println("El resultado es: "+respuesta46);
        break;
    case 3:
        System.out.println("teclea número 1 ");
        num146 = entrada46. nextInt ();
        System.out.println("teclea número 2");
        num246 = entrada46. nextInt ();
        respuesta46 = num146*num246;
        System.out.println("El resultado es: "+respuesta46);
    case 4:
        System.out.println("teclea número 1 ");
        num146 = entrada46. nextInt ();
        System.out.println("teclea número 2");
        num246 = entrada46. nextInt ();
        respuesta46 = num146/num246;
        System.out.println("El resultado es: "+respuesta46);
        break;
    default:
        System.out.print("OPCIÓN NO VÁLIDA");
        break;
    }
    // Se aplica el código lógico aquí:

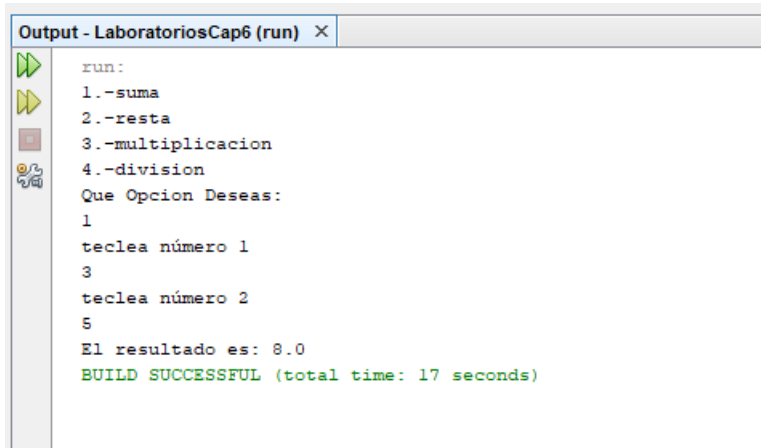
}
}

```

## Ejecución

**Figura 46**

*Ejecución del Laboratorio 46*



```
Output - LaboratoriosCap6 (run) ×
run:
1.-suma
2.-resta
3.-multiplicacion
4.-division
Que Opcion Deseas:
1
teclea número 1
3
teclea número 2
5
El resultado es: 8.0
BUILD SUCCESSFUL (total time: 17 seconds)
```

Laboratorio 47. Desarrollo de programa que aplica el estatuto `switch` para seleccionar rutinas de cálculo de venta a partir de clave de producto y cantidad vendida

Desarrollar un programa de una librería de pedidos por correo llamada La Romántica, el negocio vende 5 productos cuyos precios son los siguientes: Comics #1 = 40, Libros#2 = 4.50, Revistas#3 = 20, Periódicos #4 = 25, Cuentos#5 = 6.87. Deberá de leer una serie de pares de números que incluyan el número de productos y la cantidad vendida, su programa deberá utilizar un `switch` para determinar el precio de venta de cada producto. Debe calcular y mostrar el valor total de venta de todos los productos vendidos. Use un ciclo controlado por centinela para determinar cuándo el programa dejará de integrar para mostrar los resultados finales.

## Codificación

```
package almacendeproductos;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Almacendeproductos {
/**
 * @param args instrucción para el traspaso de argumentos
```

```

*/
public static void main (String [] args) {
Scanner entrada47 = new Scanner (System.in);
    int opcion47;
    float totaldeventas47=0;
    int venta47;
    System.out.println("1.Cómics ");
    System.out.println("2.Libros ");
    System.out.println("3.Revistas");
    System.out.println("4.Periódicos ");
    System.out.println("5.Cuentos");
    System.out.println("Qué producto deseas: ");
    System.out.println("-1 para terminar");
    opcion47 = entrada47. nextInt ();
    while (-6! =opcion47) {
    switch (opcion47)
    {
//////////
//////////
        case 1:
        {
System.out.println("Cuántos productos deseas: ");
venta = entrada47. nextInt ();
System.out.println("Precio de esta venta:"+venta*2.98);
totaldeventas47+=venta*2.98;
break;
        }
//////////
//////////
        case 2:
        {
System.out.println("Cuántos productos deseas: ");
venta = entrada47. nextInt ();
System.out.println("Precio de esta venta:"+venta*4.50);
totaldeventas47+=venta*4.50;
break;
        }
//////////
//////////

```

```
//////
    case 3:
    {
        System.out.println("Cuántos productos deseas: ");
        venta = entrada47. nextInt ();
        System.out.println("Precio de esta venta:"+venta*9.98);
        totaldeventas47+=venta47*9.98;
        break;
    }
    //////////////////////////////////////////
//////
    case 4:
    {
        System.out.println("Cuántos productos deseas: ");
        venta = entrada47. nextInt ();
        System.out.println("Precio de esta venta:"+venta*4.49);
        totaldeventas47+=venta47*4.49;
        break;
    }
    //////////////////////////////////////////
//////
    case 5:
    {
        System.out.println("Cuántos productos deseas: ");
        venta = entrada47. nextInt ();
        System.out.println("Precio de esta venta:"+venta*6.87);
        totaldeventas47+=venta47*6.87;
        break;
    }
    //////////////////////////////////////////
//////
    default:
    {
        System.out.print("introduzca opcion47 valida");
        break;
    }
}
System.out.println("Cuántos productos deseas");
```

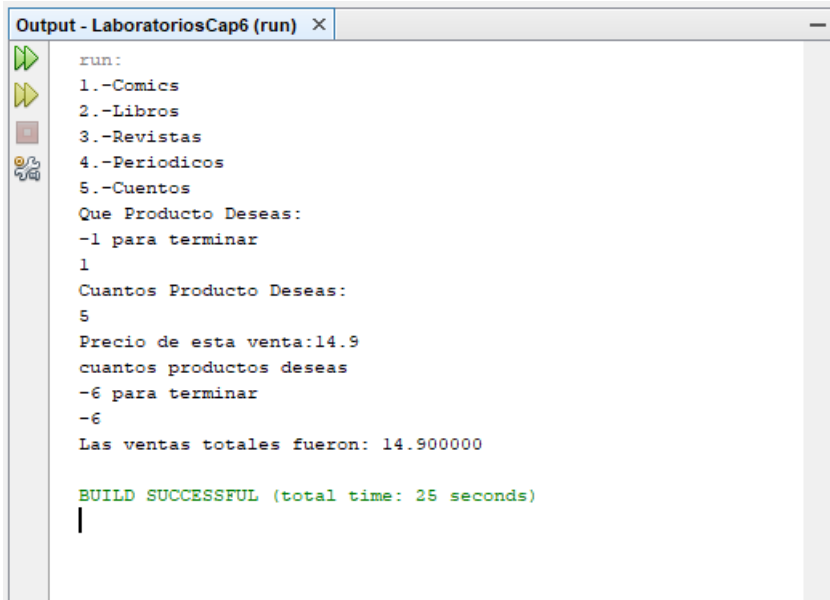
```
System.out.println("-6 para terminar");
opción = entrada47.nextInt ();

}
System.out.printf("Las ventas totales fueron: %f\n\n", totaldeventas);
}
}
```

## Ejecución

### Figura 47

*Ejecución del Laboratorio 47*



```
Output - LaboratoriosCap6 (run) X
run:
1.-Comics
2.-Libros
3.-Revistas
4.-Periodicos
5.-Cuentos
Que Producto Deseas:
-1 para terminar
1
Cuantos Producto Deseas:
5
Precio de esta venta:14.9
cuantos productos deseas
-6 para terminar
-6
Las ventas totales fueron: 14.900000

BUILD SUCCESSFUL (total time: 25 seconds)
|
```

## 6.4. Ejercicios para resolver. Casos prácticos

Implementar una serie de programas que desplieguen y operen con los siguientes formatos:

Menú escolar
1. Altas de alumnos
2. Bajas de alumnos
3. Cambio de alumnos
4. Reporte de alumnos
5. Salida
Teclee una opción _____

Implementar una serie de programas que desplieguen y operen con los siguientes formatos:

Restaurant Egg and I Menú a la carta
1. Huevos revueltos
2. Huevos a la mexicana
3. Huevos rancheros
4. Huevos estrellados
5. Huevos con machacado
6. Salida
Escoge una opción _____

Implementar una serie de programas que desplieguen y operen con los siguientes formatos:

Autobuses Correcaminos Boletos
1. Primera clase
2. Segunda clase
3. Tercera clase
4. Cuarta clase
5. Salida
Escoge una opción _____

Implementar una serie de programas que desplieguen y operen con los siguientes formatos:

Ensaladeras Cinco Pollos

1. Ensalada de pollo
2. Ensalada de pollo con uva
3. Ensalada de pollo con gravy
4. Ensalada de pollo con coditos
5. Ensalada de pollo con crema
6. Salida

Escoge una opción \_\_\_\_

Implementar una serie de programas que desplieguen y operen con los siguientes formatos:

Tornillería El Apretón

1. Tornillos de media
2. Tornillos de tres cuartos
3. Tornillos de pulgada
4. Tornillos de acero
5. Tornillos galvanizados
6. Salidas

Escoge una opción \_\_\_\_

## 6.5. Referencias de consulta

<https://javadesdecero.es/basico/sentencia-switch-java-ejemplo/>

<http://www.edu4java.com/es/java/sentencia-switch-java.html>

<https://codigofacilito.com/articulos/ejemplos-switch-java>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/while>

## 6.6. Videos de apoyo

<https://www.youtube.com/watch?v=3wugC8uyb4o>

<https://www.youtube.com/watch?v=9HKDSbH2fmw>

<https://www.youtube.com/watch?v=tL2ArvkKAs8>



# Capítulo 7

## Estructuras de control anidadas

### 7.1. Objetivo del capítulo

Que el estudiante aplique ciertas técnicas de control como el anidamiento dentro de diferentes alternativas cíclicas. Además, conocerá que la estructura de control interior y exterior puede ser de diferente tipo de datos.

### 7.2. Introducción

Programas con *if-else* anidados. Es decir que hay un ciclo dentro de otro ciclo, de hecho, es muy común que se presenten estas estructuras anidadas. Adicionalmente, están las estructuras *while*, *for*, *do while*. Todas pueden ir una dentro de la otra con distinta variable de control. En los siguientes ejemplos el alumno aprenderá a trabajar con estructuras anidadas utilizando diferentes combinaciones de estructuras cíclicas, una dentro de la otra.

### 7.3. Ejemplos prácticos realizados utilizando comandos anteriores

Laboratorio 48. Desarrollo de programa que genera resultados a partir de dos datos

Desarrollar un programa que lea 2 números y que imprima si el primer número es múltiplo del segundo.

#### Codificación

```
package ejemplo52;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejemplo52 {

    /**
     * @param args instrucción para el traspaso de argumentos
```

```

*/
public static void main (String [] args) {
    Scanner entrada48 =new Scanner (System.in);
    System.out.println("Ingrese el primer número");
    int numero148=entrada48.nextInt();
    System.out.println("Ingrese el segundo número");
    int numero248=entrada48.nextInt();
    System.out.println("*****");
    if((numero148%2==0) &&(numero248%2==0))
    {
        System.out.println("Son múltiplos");
    }
    else {
        System.out.println("No son múltiplos");
        // Se aplica el código lógico aquí:
    }
}
}

```

## Ejecución

### Figura 48

*Ejecución del Laboratorio 48*

```

Output - LaboratoriosCap7 (run) ×
run:
Ingrese el primer número
4
Ingrese el segundo número
2
*****
Son multiplos
BUILD SUCCESSFUL (total time: 4 seconds)
|

```

## Laboratorio 49. Desarrollo de programa que genera resultados de su operación a partir de dos datos

Desarrollar un programa que lea 2 números y que imprima si son múltiplos.

### Codificación

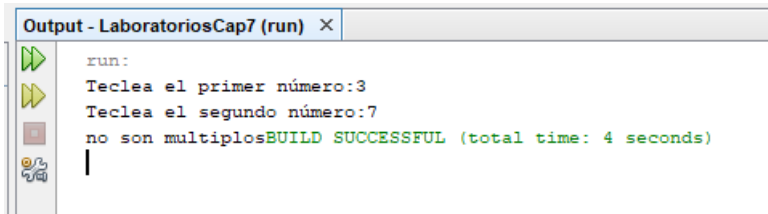
```
package práctica15;
import java. util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Práctica15 {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada49 = new Scanner (System.in);
        int numero149;
        int numero249;
        System.out.printf("Teclea el primer número:");
        numero149 = entrada49. nextInt ();
        System.out.printf("Teclea el segundo número:");
        numero249 = entrada49. nextInt ();
        if(numero249%2==0)
        {
            System.out.printf("si son múltiplos %d & %d", numero149, numero249);
        }
        else
        {
            System.out.print("no son múltiplos");
        }
        }
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

### Figura 49

*Ejecución del Laboratorio 49*



```
Output - LaboratoriosCap7 (run) x
run:
Tecllea el primer número:3
Tecllea el segundo número:7
no son multiplosBUILD SUCCESSFUL (total time: 4 seconds)
|
```

## Laboratorio 50. Desarrollo de programa dado un número de 5 dígitos los separe con espacios

Diseñar y desarrollar un programa que reciba del usuario un número compuesto por 5 dígitos y estos se desplieguen separados por 5 espacios.

## Codificación

```
package práctica16;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Práctica16 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada50 = new Scanner (System.in);
        int numero50, V50, W50, X50, Y50, Z50;
        System.out.print("Dame un número de 5 dígitos: ");
        Numero50=entrada50. nextInt ();
        if((numero50>=10000) &&(numero50<=99999))
        {
            V50 = numero50/10000;
            W50 = (numero50 %10000) /1000;
            X50 = (numero50%10000) %1000/100;
            Y50 = ((numero50%10000) %1000) %100/10;
```

```

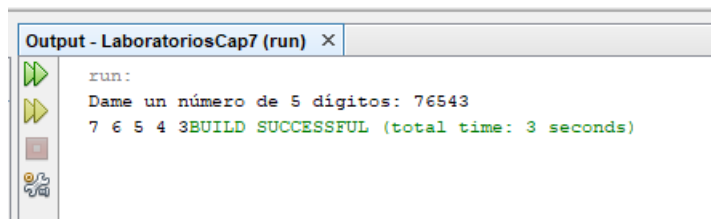
        Z50= (((numero50%10000) %1000) %100) %10;
        System.out.printf("%d %d %d %d %d", V50, W50, X50, Y50, Z50);
    }
    else
    {
        System.out.print("No es un número de 5 dígitos");
    }
}
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 50

*Ejecución del Laboratorio 50*



Laboratorio 51. Desarrollo de programa para que a partir de un número reporte si es par o impar

Crear un programa que lea un número y detecte si es par o impar, usar el símbolo residuo %.

## Codificación

```

import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio13 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada51 = new Scanner (System.in);
    }
}

```

```

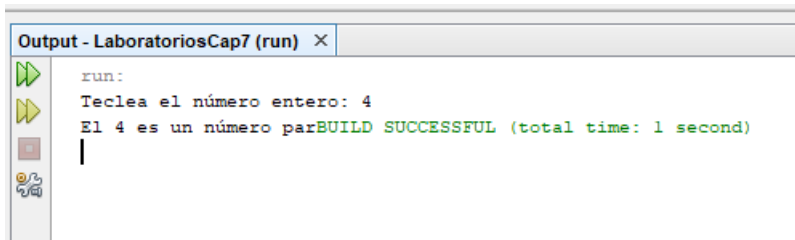
int num51;
System.out.print("Teclea el número entero: ");
num51 = entrada51. nextInt ();
if(num%2==0)
{
    System.out.printf("El %d es un número par", num51);
}
else
{
    System.out.printf("El %d es un número impar", num51);
}
}
}

```

## Ejecución

### Figura 51

*Ejecución del Laboratorio 51*



## Laboratorio 52. Desarrollo de programa para la práctica del estatuto *break*

Hacer un programa donde se aplique la función *break*.

### Codificación

```

package prueba
break;
/**
 *
 * @author jxs_b
 */
public class PuebaBreak {
    /**

```

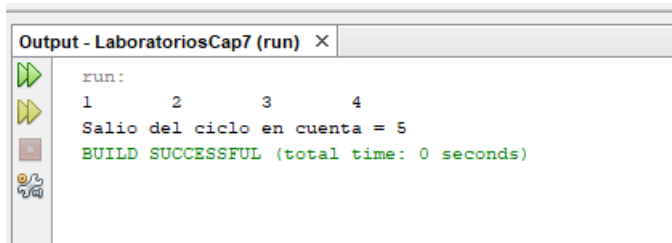
\* @param args instrucción para el traspaso de argumentos

```
*/  
public static void main (String [] args) {  
    int cuenta52;  
    for (cuenta52=1; cuenta52<=10; cuenta52++)  
    {  
        if (cuenta52 ==5)  
            break;  
        System.out.printf("%d\t", cuenta52);  
    }  
    System.out.printf("\nSalio del ciclo en cuenta52 = %d\n", cuenta52);  
    // Se aplica el código lógico aquí:  
}  
  
}
```

## Ejecución

**Figura 52**

*Ejecución del Laboratorio 52*



## Laboratorio 53. Implementación de un programa para aplicar el estatuto *continue*

Diseñe y desarrolle un programa donde se aplique la función de *continue*.

## Codificación

```
package pruebacontinue;  
/**  
 *  
 * @author jxs_b  
 */
```

```

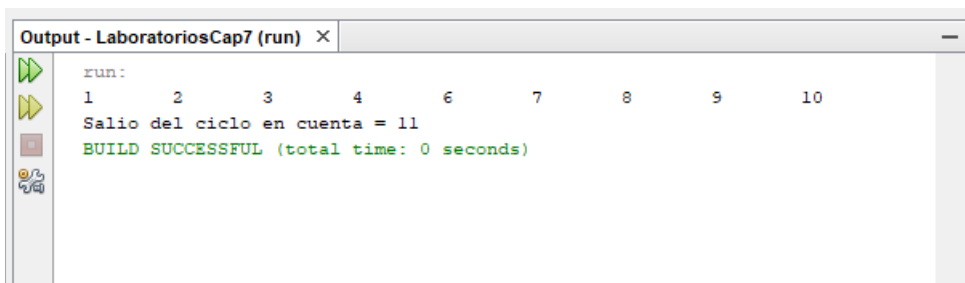
public class PruebaContinue {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int cuenta53;
        for (cuenta53=1; cuenta53<=10; cuenta53++)
        {
            if (cuenta53 ==5)
                continue;
            System.out.printf("%d\t", cuenta53);
        }
        System.out.printf("\nSalió del ciclo en cuenta = %d\n", cuenta53);
        // Se aplica el código lógico aquí:
    }
}

```

## Ejecución

### Figura 53

*Ejecución del Laboratorio 53*



```

Output - LaboratoriosCap7 (run) ×
run:
1      2      3      4      6      7      8      9      10
Salió del ciclo en cuenta = 11
BUILD SUCCESSFUL (total time: 0 seconds)

```



## Laboratorio 54. Desarrollo de programa que se aplica para cálculos de nómina

Implementar un programa que pida al usuario 10 sueldos e imprimir la suma total de todos los sueldos y cuántos de esos son mayores a 10 000.

### Codificación

```
package ejercicio55;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio55 {

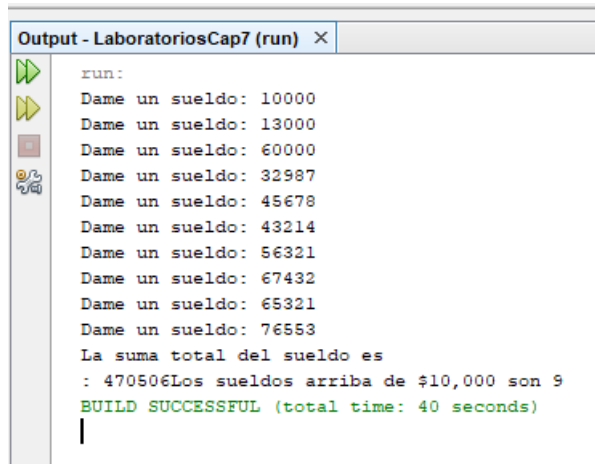
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int sueldos54;
        int contador54=0;
        int contador254=0;
        Scanner entrada54 = new Scanner (System.in);
        for (int i54 = 1; i54<=10; i54++)
        {
            System.out.print("Dame un sueldo: ");
            sueldos54 =entrada54. nextInt ();
            contador54 = contador54 + sueldos54;

            if(sueldos54>10000) {
                contador254 = contador254 + 1;
            }
        }
        System.out.printf("La suma total del sueldo es\n: "+contador54);
        System.out.printf("Los sueldos arriba de $10,000 son %d\n", contador254);
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

### Figura 54

*Ejecución del Laboratorio 54*



```
Output - LaboratoriosCap7 (run) x
run:
Dame un sueldo: 10000
Dame un sueldo: 13000
Dame un sueldo: 60000
Dame un sueldo: 32987
Dame un sueldo: 45678
Dame un sueldo: 43214
Dame un sueldo: 56321
Dame un sueldo: 67432
Dame un sueldo: 65321
Dame un sueldo: 76553
La suma total del sueldo es
: 470506Los sueldos arriba de $10,000 son 9
BUILD SUCCESSFUL (total time: 40 seconds)
```

Laboratorio 55. Desarrollo de programa que valide si algún dato capturado es negativo

Implementar un programa que introduzca desde el teclado 10 números e imprimir si se ha introducido algún número negativo.

## Codificación

```
package ejecicio50;
```

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

```
/**
```

```
*
```

```
* @author jxs_b
```

```
*/
```

```
public class Ejecicio50 {
```

```
/**
```

```
* @param args instrucción para el traspaso de argumentos
```

```
*/
```

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

```
Scanner entrada55 = new Scanner (System.in);
```

```

for (int i55=1; i55 <=10; i55++)
{
    int numeros55;
    System.out.print("Dame un número:");
    numeros55=entrada55.nextInt ();
    if(numeros55<=0)
    {
        System.out.println("Es negativo este número");
    }

}

// Se aplica el código lógico aquí:
}

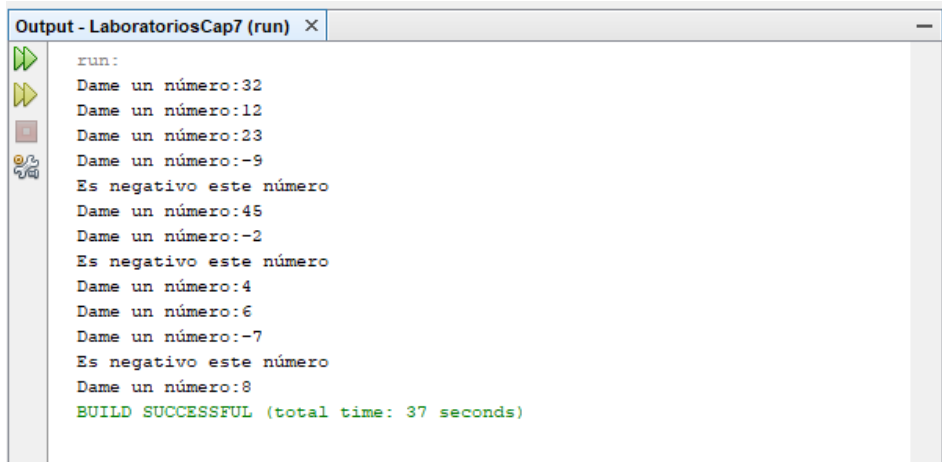
}

```

## Ejecución

**Figura 55**

*Ejecución del Laboratorio 55*



```

Output - LaboratoriosCap7 (run) x
run:
Dame un número:32
Dame un número:12
Dame un número:23
Dame un número:-9
Es negativo este número
Dame un número:45
Dame un número:-2
Es negativo este número
Dame un número:4
Dame un número:6
Dame un número:-7
Es negativo este número
Dame un número:8
BUILD SUCCESSFUL (total time: 37 seconds)

```

Laboratorio 56. Desarrollo de programa que selecciona el menor de N datos  
Hacer un programa que imprima el menor de varios enteros. El primer dato especifica el número de enteros que se van a leer.

### Codificación

```
package ejercicio64;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Ejercicio64 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada56 = new Scanner (System.in);
        int numero56;
        int n56;
        int menor56=999999999;

        System.out.print("teclea el número de veces ");
        numero=entrada56. nextInt ();
        for (int i56=1; i56<=numero56; i56++) {

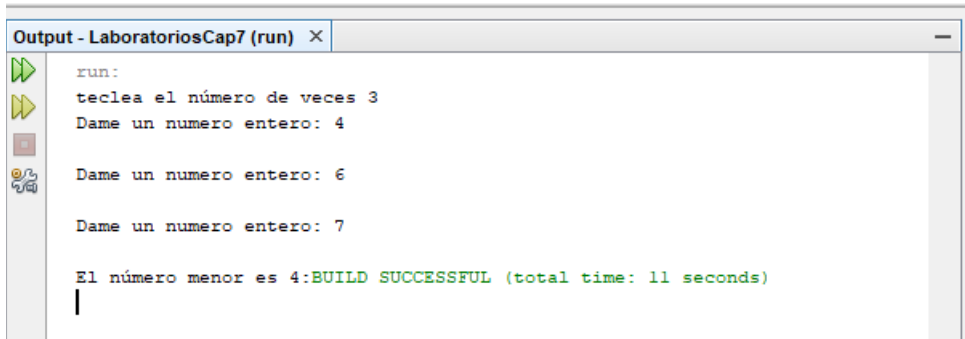
            System.out.printf("Dame un número entero:");
            n56=entrada56. nextInt ();
            if(n56<menor56)
                menor56=n;
            System.out.println();
        }
        System.out.printf("El número menor es %d:", menor);
        // Se aplica el código lógico aquí:

    }
}
```

## Ejecución

Figura 56

*Ejecución del Laboratorio 56*



```
Output - LaboratoriosCap7 (run) ×
run:
  teclea el número de veces 3
  Dame un numero entero: 4
  Dame un numero entero: 6
  Dame un numero entero: 7
  El número menor es 4:BUILD SUCCESSFUL (total time: 11 seconds)
  |
```

Laboratorio 57. Desarrollo de programa que lea N datos y calcule el producto de 10 impares

Crear un programa que imprima el producto de los primeros 10 números impares.

## Codificación

```
package ejercicio54;
/**
 *
 * @author jxs_b
 */
public class Ejercicio54 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args)
    {
        int numimpar57=1;
        for (int i57 =1; i57<=10; i57++)
        {

            if(i57%2==1)
            {

                numimpar57= numimpar57*i57;
            }
        }
    }
}
```

```

}
System.out.printf("El producto es: %d", numimpar57);
    // Se aplica el código lógico aquí:

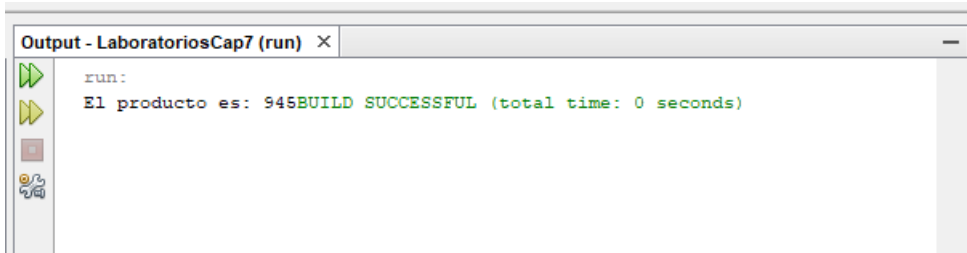
}
}

```

## Ejecución

### Figura 57

*Ejecución del Laboratorio 57*



## Laboratorio 58. Desarrollo de programa que reporte los números primos para N datos proporcionados

Desarrolle un programa que lea un número N e imprima cuántos números primos hay entre 1 y N.

## Codificación

```

package numerosprimos;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Numerosprimos {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada58 = new Scanner (System.in);
        int j58, num58, cont_pri58;
        boolean primo58;

```

```

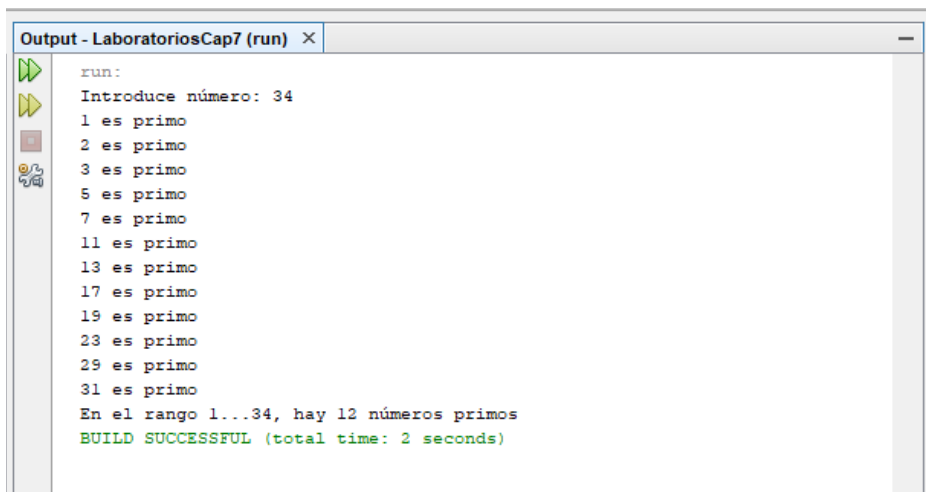
cont_pri58=0;
System.out.print("Introduce número: ");
num58=entrada58.nextInt ();
for (int i58=1; i58<=num58; i58++) {
    primo58=true;
    j58=2;
    while (j58<=i58-1 && primo58==true)
    {
        if (i58%j58==0)
            primo58=false;
        j58++;
    }
    if (primo58==true) {
        cont_pri58++;
        System.out.println(i+ (" es primo"));
    }
}
System.out.println("En el rango 1..." + num58 + ", hay "+ cont_pri58 + " números
primos");
}

```

## Ejecución

### Figura 58

*Ejecución del Laboratorio 58*



```

Output - LaboratoriosCap7 (run) x
run:
Introduce número: 34
1 es primo
2 es primo
3 es primo
5 es primo
7 es primo
11 es primo
13 es primo
17 es primo
19 es primo
23 es primo
29 es primo
31 es primo
En el rango 1...34, hay 12 números primos
BUILD SUCCESSFUL (total time: 2 seconds)

```

## Laboratorio 59. Desarrollo de un programa que calcule el monto a pagar según las horas que los vehículos estuvieron estacionados

Un estacionamiento llamado Tigres, cobra una cuota mínima de \$10 la hora por estacionarse con un límite máximo de 3 horas, al exceder ese plazo el estacionamiento cobrará \$.50 adicionales por cada hora. El pago de la cuarta hora será de 10.5, el de la quinta hora será de 11.00, el de la sexta será de 11.5 y así sucesivamente, de tal forma que el precio máximo por pagar en una estancia de 24 horas sea de \$24 por hora. Suponga que los clientes estacionan su automóvil sin rebasar invariablemente las 24 horas. Escriba una aplicación que calcule e imprima los cargos por estacionamiento para cada cliente que se haya estacionado. Deberá introducir las horas de estacionamiento para cada cliente, deberá imprimir el cargo para el cliente que sale del estacionamiento y calcular e imprimir el total de los recibos de la jornada.

### Codificación

```
package estacionamiento. tigres;
import java. util. Scanner;
/**
 *
 * @author obrya
 */
public class Estacion59
{
    public void Entrada59 ()
    {
        Scanner entrada59 = new Scanner (System.in);
        double horas59 = 0;
        double cargos59 = 0;
        double total59 = 0;
        int contador59 = 1;
        System.out.println("\n=====
        ===");
        System.out.println("\n===Bienvenido al sistema de cobro del estacionamiento===");
        System.out.println("\n=====
        ===");

        System.out.printf("Por favor introduzca el número de horas del coche %d: ",
        contador++);
```



```

System.out.print("\n (-1 para terminar) \n");
horas59 = entrada59.nextDouble ();

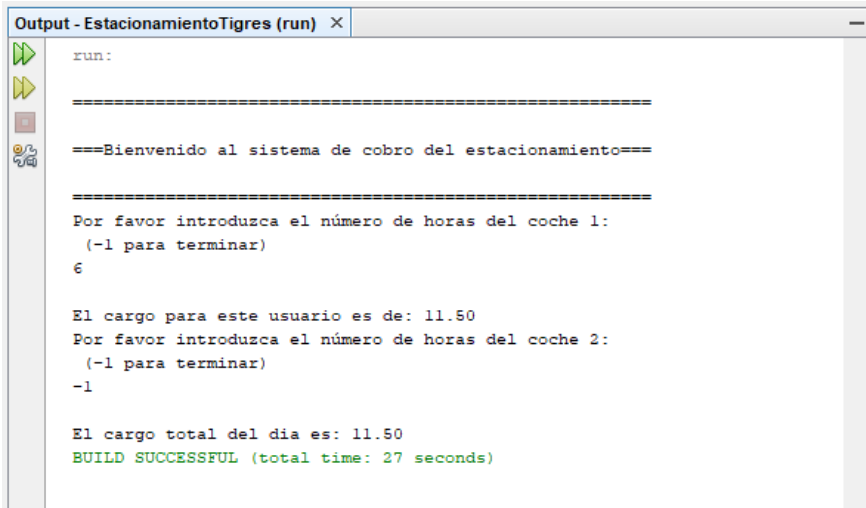
while ( -1 != horas59)
{
System.out.printf("\nEl cargo para este usuario es de: %.2f\n", Cargo59(horas59));
Total59 += Cargo59(horas59);
System.out.printf("Por favor introduzca el número de horas del coche %d: ",
contador59++);
System.out.print("\n (-1 para terminar) \n");
horas59 = entrada59.nextDouble ();
}
System.out.printf("\nEl cargo total del día es: %.2f\n", total59);
}
public double Cargo59 (double x59)
{
if (3 >= x59)
return 10.0;
else
return (10 + 0.5*Math.ceil(x59 - 3)) <= 24? 10 + 0.5*Math.ceil(x59 -3): 24;
}
}
package estacionamiento. tigres;
/**
 *
 * @author obrya
 */
public class EstacionamientoTigres
{
/**
 * @param args instrucción para el traspaso de argumentos
 */
public static void main (String [] args)
{
Estacion Estacionamiento59 = new Estacion59 ();
Estacionamiento59.Entrada59();
}
}

```

## Ejecución

Figura 59

Ejecución del Laboratorio 59



```
run:
=====
===Bienvenido al sistema de cobro del estacionamiento===
=====
Por favor introduzca el número de horas del coche 1:
(-1 para terminar)
6
El cargo para este usuario es de: 11.50
Por favor introduzca el número de horas del coche 2:
(-1 para terminar)
-1
El cargo total del día es: 11.50
BUILD SUCCESSFUL (total time: 27 seconds)
```

### 7.4. Ejercicios para resolver. Casos prácticos

Programa con contador que imprima números del 100 al 1.

Implementar un programa que utilice ciclo `for` con contador para imprimir los números del 7 al 77.

Hacer un programa con contador que imprima de 1 al 100. Utilice el ciclo `while`.

Desarrollar un programa que imprima los números de 2 en 2 empezando en el número 20 y terminando en el número 2, utilizando el ciclo `for`.

### 7.5. Referencias de consulta

- <https://www.tutorialesprogramacionya.com/javaya/detalleconcepto.php?codigo=81&>
- <https://creatividadcodificada.com/java/estructuras-de-control-de-java/>
- <http://puntocomnoesunlenguaje.blogspot.com/2012/04/estructuras-de-control.html#:~:text=Las%20estructuras%20de%20control%20determinan,las%20>
- <https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/while>

## 7.6. Videos de apoyo

<https://www.youtube.com/watch?v=S2hPOUXcoCg>

<https://www.youtube.com/watch?v=4UbUiQ7qRCA>

<https://www.youtube.com/watch?v=MZifO1PKL4M>



# Capítulo 8

## Arreglos unidimensionales

### 8.1. Objetivo del capítulo

El alumno aprenderá a introducir con diferentes alternativas varios datos en una variable de arreglo de una dimensión.

### 8.2. Introducción

Los ejemplos anteriores se trabajaron con variables que manejan un solo dato simple. En este capítulo, empieza el alumno a conocer que una variable puede tener distintos datos de un solo tipo. Estos arreglos ayudarán al alumno a comprender donde se podrían aplicar estos conocimientos prácticos.

### 8.3. Ejemplos prácticos utilizando arreglos de una sola dimensión

Laboratorio 60. Desarrollo de programa para la práctica de imprimir un vector

Crear un programa que imprima un arreglo unidimensional con valores ya establecidos.

#### Codificación

```
package arreglo;
/**
 *
 * @author jxs_b
 */
public class ARREGLO60 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int arreglo60 [] = {32,27,64,18,95,14,90,70,60,37};
        System.out.printf("%s%8s\n", "ÍNDICE", VALOR);
        for (int contador60=0; contador60<arreglo.length; contador60++)
```

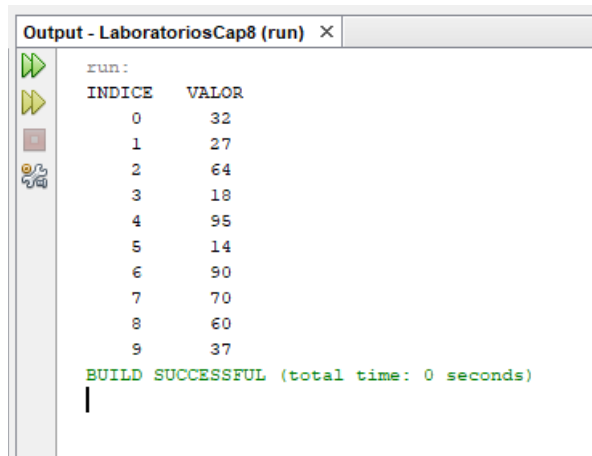
```
System.out.printf("%5d%8d\n", contador60, arreglo60[contador60]);  
// Se aplica el código lógico aquí:
```

```
}}
```

## Ejecución

### Figura 60

*Ejecución del Laboratorio 60*



```
run:  
INDICE  VALOR  
0       32  
1       27  
2       64  
3       18  
4       95  
5       14  
6       90  
7       70  
8       60  
9       37  
  
BUILD SUCCESSFUL (total time: 0 seconds)
```

Laboratorio 61. Desarrollo de programa que imprima los datos almacenados en una matriz o arreglo bidimensional

Diseñar y desarrollar un programa que contenga un arreglo bidimensional, datos ya establecidos e imprimirlos.

## Codificación

```
package arreglo1;  
/**  
 *  
 * @author jxs_b  
 */  
public class Arreglo61 {  
    /**  
     * @param args instrucción para el traspaso de argumentos     */  
    public static void main (String [] args) {  
        int arreglo61[] [] = {{1, 2, 3 ,5}, {4, 5, 6 ,3}};  
        System.out.println("Los valores en arreglo1 por filas son");
```

```

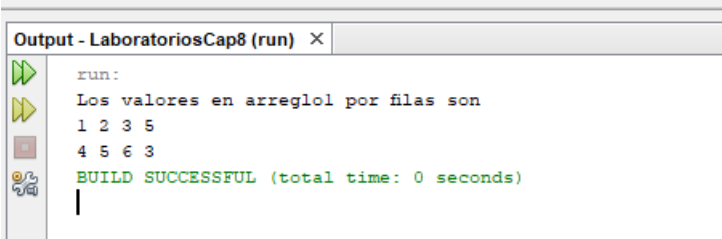
    imprimirArreglo (arreglo61);
}
public static void imprimirArreglo61 (int arreglo61 [] []) {
    for (int [] arreglo61: arreglo61) {
        for (int columna61 = 0; columna61 < arreglo61.length; columna61++) {
            System.out.printf("%d ", arreglo61[columna61]);
        }
        System.out.println();
    } }
}
}
}

```

## Ejecución

### Figura 61

*Ejecución del Laboratorio 61*



```

Output - LaboratoriosCap8 (run) x
run:
Los valores en arreglo1 por filas son
1 2 3 5
4 5 6 3
BUILD SUCCESSFUL (total time: 0 seconds)
|

```

## Laboratorio 62. Desarrollo de programa para practicar manejo de contenidos de un vector

Diseñar y desarrollar un programa que obtenga un arreglo con 10 elementos e imprima un encabezado llamada *índice* y otro llamado *valor*, además deberá imprimir el arreglo desde 0 en el valor.

## Codificación

```

package arreglo2;
/**
 *
 * @author jxs_b
 */
public class Arreglo2 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {

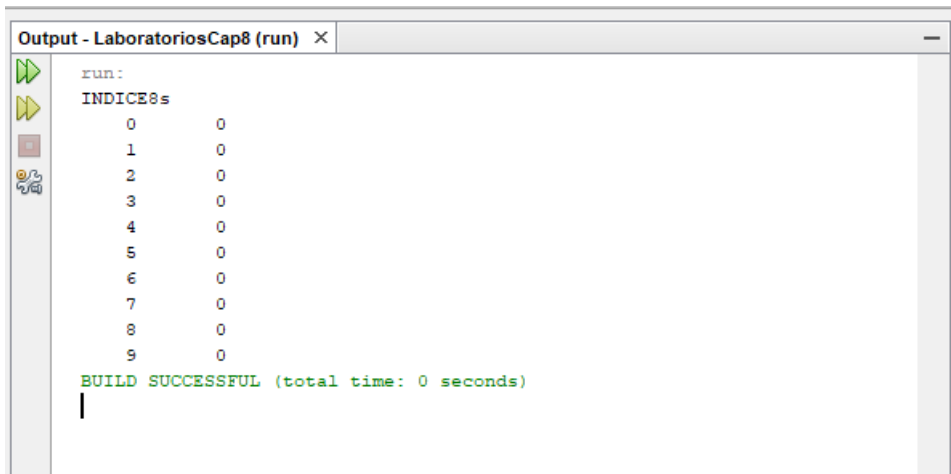
```

```
int arreglo [];  
arreglo62 = new int [10];  
System.out.printf("%s8s\n",ÍNDICE",VALOR");  
for (int contador62=0; contador62<arreglo. length; contador62++)  
  
System.out.printf("%5d%8d\n", contador62, arreglo62[contador62]);  
// Se aplica el código lógico aquí:  
  
}  
}
```

## Ejecución

### Figura 62

*Ejecución del Laboratorio 62*



```
Output - LaboratoriosCap8 (run) ×  
run:  
ÍNDICE8s  
0      0  
1      0  
2      0  
3      0  
4      0  
5      0  
6      0  
7      0  
8      0  
9      0  
BUILD SUCCESSFUL (total time: 0 seconds)
```



## Laboratorio 63. Desarrollo de programa para imprimir los contenidos de dos matrices

Diseñe y desarrolle un programa que maneje 2 arreglos bidimensionales con valores ya establecidos por el usuario y los imprima.

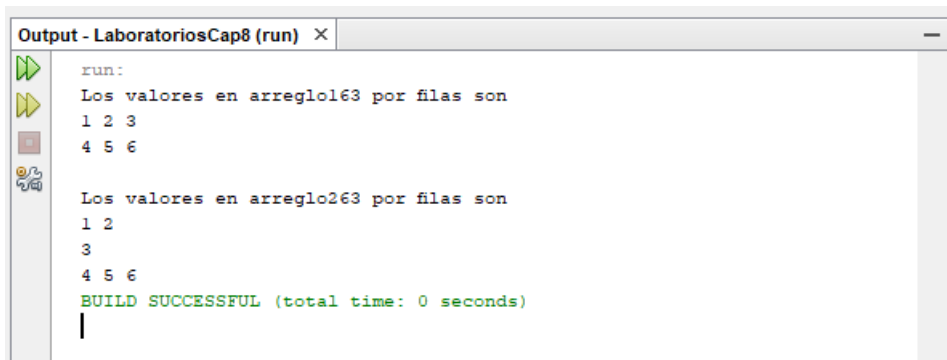
### Codificación

```
package arreglomultidimensional;
/**
 *
 * @author jxs_b
 */
public class Arreglomultidimensional63 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int arreglo163[] [] = {{1, 2, 3}, {4, 5, 6}};
        int arreglo263[] [] = {{1, 2}, {3}, {4, 5, 6}};
        System.out.println("Los valores en arreglo163 por filas son");
        imprimirArreglo (arreglo163);
    // muestra arreglo163 por filas
        System.out.println("\nLos valores en arreglo263 por filas son");
        imprimirArreglo (arreglo263); // muestra arreglo263 por filas 17
    } // fin de main // imprime filas y columnas de un arreglo bidimensional
    public static void imprimirArreglo (int arreglo63 [] []) {
        // itera a través de las filas del arreglo
        for (int [] arreglo163: arreglo63) {
            // itera a través de las columnas de la fila actual
            for (int columna63 = 0; columna63 < arreglo163.length; columna63++) {
                System.out.printf("%d ", arreglo163[columna63]);
            }
            System.out.println(); // inicia nueva línea de salida
        } // fin de for externo
    } // fin del método imprimirArreglo
}
```

## Ejecución

### Figura 63

Ejecución del Laboratorio 63



```
Output - LaboratoriosCap8 (run) x
run:
Los valores en arreglo163 por filas son
1 2 3
4 5 6

Los valores en arreglo263 por filas son
1 2
3
4 5 6
BUILD SUCCESSFUL (total time: 0 seconds)
```

Laboratorio 64. Desarrollo de programa para crear un vector con 5 datos alimentados por teclado

Crear un programa que genere un arreglo con 5 datos desde el teclado e imprimirlos.

## Codificación

```
package arreglo5datos;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Arreglo564datos {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int i64;
        int [] numeros64= new int [5];
        Scanner teclado64 = new Scanner (System.in);
        for (i64=0; i<numeros64. length; i64++)
        {
            System.out.printf("Introduzca número %d:", i64+1);
            Numeros64[i]=teclado64. nextInt ();
        }
    }
}
```

```

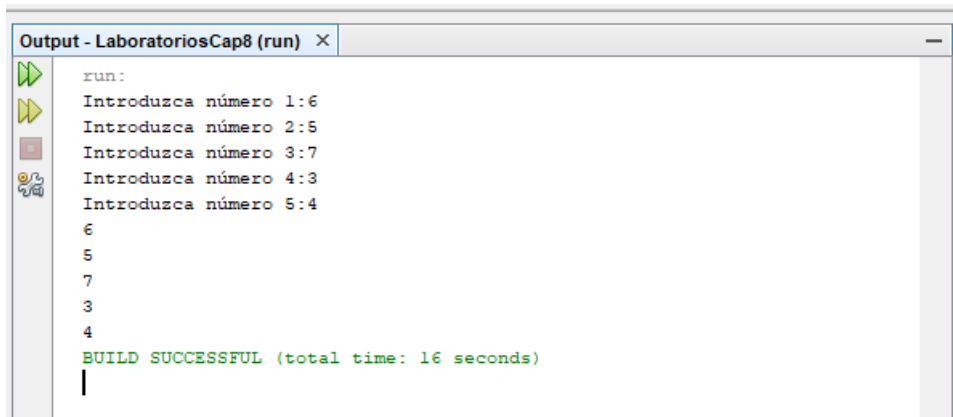
    }
    for (i64=0; i<numeros64.length; i64++) {
        System.out.println(numeros64[i64]);
    }
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

**Figura 64**

*Ejecución del Laboratorio 64*



```

Output - LaboratoriosCap8 (run) x
run:
Introduzca número 1:6
Introduzca número 2:5
Introduzca número 3:7
Introduzca número 4:3
Introduzca número 5:4
6
5
7
3
4
BUILD SUCCESSFUL (total time: 16 seconds)
|

```

Laboratorio 65. Desarrollo de programa que crea un vector con 4 datos alimentados y calcula su suma

Implementar un programa que produzca un arreglo con 4 elementos introducidos desde el teclado por el usuario e imprimir la suma de su contenido.

## Codificación

```

package arreglo4datossuma;
import java.util. Scanner;

/**
 *
 * @author jxs_b
 */
public class Arreglo4datossuma {

```

```

/**
 * @param args instrucción para el traspaso de argumentos
 */
public static void main (String [] args) {
    int i65;
    int total65 = 0;
    int [] numeros65= new int [4];
    Scanner teclado65 = new Scanner (System.in);
    for (i65=0; i65<numeros. length; i65++)
    {
        System.out.printf("Introduzca número %d:", i65+1);
        numeros65[i65]=teclado65. nextInt ();
    }
    for (int contador65 =0; contador65<numeros65. length; contador65++) {

        total65 += numeros65[contador65];
    }
    int contador65=0;

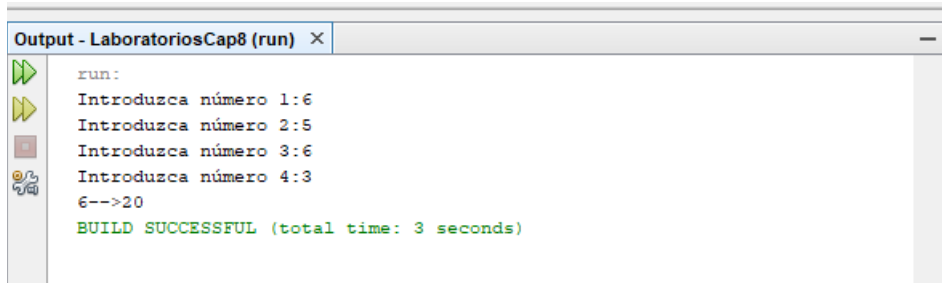
    System.out.println(numeros[contador65]+"-->" +total65);
    }
    // Se aplica el código lógico aquí:
}

```

## Ejecución

### Figura 65

*Ejecución del Laboratorio 65*



```

Output - LaboratoriosCap8 (run) x
run:
Introduzca número 1:6
Introduzca número 2:5
Introduzca número 3:6
Introduzca número 4:3
6-->20
BUILD SUCCESSFUL (total time: 3 seconds)

```

## Laboratorio 66. Desarrollo de programa que seleccione el menor y el mayor de los datos contenidos en un vector

Hacer un programa para crear un arreglo llamado lista de números con 5 datos ya establecidos e imprimir el máximo y el mínimo en ese arreglo.

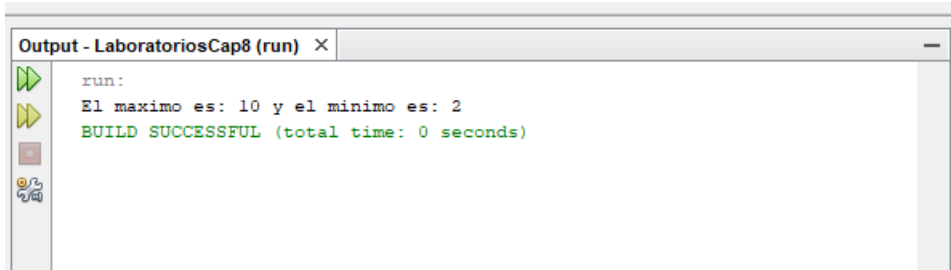
### Codificación

```
package arregloobtenermayor;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Arregloobtenermayor {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int [] listaNumeros66 = {2,4,6,8,10};
        Scanner obtenerNumero66 = new Scanner (System.in);
        int i66, max66, min66;
        min66=max66=listaNumeros66 [0];
        for (i66=0; i66<listaNumeros66.length; i66++)
        {
            if(min66>listaNumeros66[i66])
            {
                Min66=listaNumeros66[i66];
            }
            if(max66<listaNumeros66[i66])
            {
                Max66=listaNumeros66[i66];
            }
        }
        System.out.println("El máximo es: "+ max66 + "y el mínimo es: "+ min66);
        // Se aplica el código lógico aquí:
    }
}
```

## Ejecución

### Figura 66

*Ejecución del Laboratorio 66*



Laboratorio 67. Desarrollo de programa que copie los datos contenidos en un vector en un segundo arreglo y ambos se impriman

Desarrollar un programa que genere un arreglo A con 5 datos por medio de la asignación y copiarlo a otro arreglo B e imprimir los 2 arreglos con su posición.

## Codificación

```
package copiardeunarreglootro;
/**
 *
 * @author jxs_b
 */
public class Copiardeunarreglootro {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int [] arreglo67;
        arreglo67 = new int [5];
        arreglo67 [0]=1;
        arreglo67 [1]=2;
        arreglo67 [2]=3;
        arreglo67 [3]=4;
        arreglo67 [4]=5;
        int arreglo672 [] = new int [5];
        for (int i67 = 0; i67 <5; i67++) {
            arreglo672 [i67] = arreglo67 [i67];
        }
    }
}
```

```

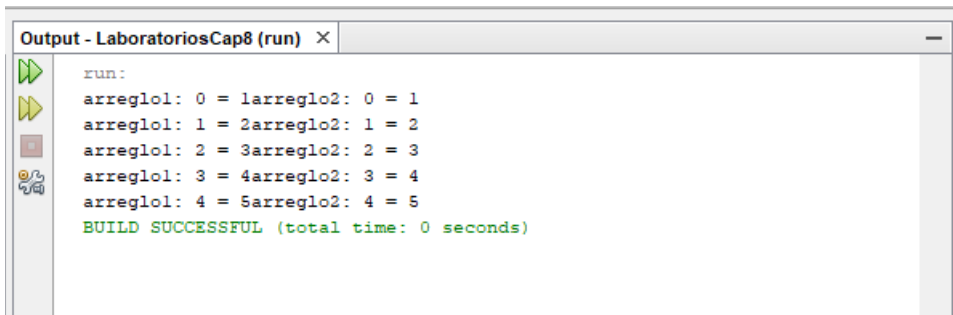
System.out.print("arreglo1: "+i+" = " + arreglo67[i67]);
System.out.println("arreglo2: "+i+" = " + arreglo672[i67]);
}
// Se aplica el código lógico aquí:
}
}
}

```

## Ejecución

**Figura 67**

*Ejecución del Laboratorio 67*



```

Output - LaboratoriosCap8 (run) x
run:
arreglo1: 0 = 1arreglo2: 0 = 1
arreglo1: 1 = 2arreglo2: 1 = 2
arreglo1: 2 = 3arreglo2: 2 = 3
arreglo1: 3 = 4arreglo2: 3 = 4
arreglo1: 4 = 5arreglo2: 4 = 5
BUILD SUCCESSFUL (total time: 0 seconds)

```

## 8.4. Resolución de casos prácticos

Implementar un programa con 2 arreglos con 5 datos ya establecidos llamados a y b, copiar del arreglo A al arreglo B y obtener la multiplicación de ambos e imprimir resultados.

Crear un programa que obtenga un arreglo con 5 datos asignados por el usuario, copiar ese arreglo, pero en forma inversa e imprimir los 2 arreglos.

Desarrollar un programa que obtenga un arreglo que capture 5 elementos mediante el teclado al azar, copiar ese arreglo acomodar de mayor a menor e imprimir los 2 arreglos.

Implementar un programa que obtenga un arreglo con 5 elementos capturados mediante el teclado, sumar los números pares e imprimir resultado.

Hacer un programa para crear un arreglo B con 5 elementos dados desde el teclado y obtenga de los números nones el valor mayor e imprimirlo.

### **8.5. Referencias de consulta**

<http://puntocomnoesunlenguaje.blogspot.com/2012/10/arrays-unidimensionales-en-java.html>

<https://educandocontic.com/arrays-unidimensionales-en-java/>

<https://javadesdecero.es/arrays/unidimensionales-multidimensionales/>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/while>

### **8.6. Video de apoyo**

<https://www.youtube.com/watch?v=ADa8uA1Te44>

<https://www.youtube.com/watch?v=Sw1fEnn-VIY>

<https://www.youtube.com/watch?v=us4mIbaiVrQ>



# Capítulo 9

## Arreglos multidimensionales

### 9.1. Objetivos del capítulo

El alumno aprenderá a desarrollar diferentes aplicaciones en la práctica como inventarios, ventas generales, detectar qué día se vende más, qué producto se vendió más y en qué día.

### 9.2. Introducción

En este capítulo se presentan programas con matrices y subíndices con más de una dimensión. Todos los programas trabajan con variables simples cuyo contenido son números enteros.

### 9.3. Ejemplos prácticos realizados utilizando arreglos multidimensionales

Laboratorio 68. Desarrollo de programa que lea los 9 datos en una matriz de 3 x 3 y que los imprima

Diseñar y desarrollar un programa que capture un arreglo de 3\*3 llamado A1 con datos introducidos por el usuario e imprimir ese arreglo.

#### Codificación

```
package arreglo2.pkg8abril;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Arreglo68abril {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
```

```

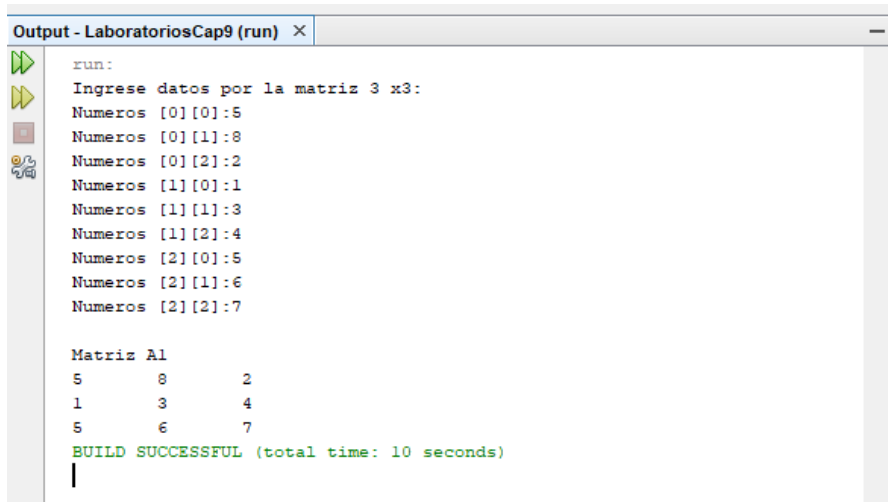
Scanner entradaDatosMatriz68 = new Scanner (System.in);
int numeros68 [] [] = new int [3][3];
System.out.println("Ingrese datos por la matriz 3 x3:");
for (int h68=0; h68<numeros. length; h68++) {
    for (int j68=0; j68<numeros. length; j68++) {
        System.out.print("Números ["+h68+"]["+j68+"]");
        numeros [h68][j68]=entradaDatosMatriz68.nextInt();
    }
}
entradaDatosMatriz68.close();
System.out.println("\nMatriz A1");
for (int []num68: numeros68) {
    for (int elem68:num68) {
        System.out.print(elem68+"\t");
    }
    System.out.print("\n");
}
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 68

*Ejecución del Laboratorio 68*



```

Output - LaboratoriosCap9 (run) ×
run:
Ingrese datos por la matriz 3 x3:
Numeros [0][0]:5
Numeros [0][1]:8
Numeros [0][2]:2
Numeros [1][0]:1
Numeros [1][1]:3
Numeros [1][2]:4
Numeros [2][0]:5
Numeros [2][1]:6
Numeros [2][2]:7

Matriz A1
5      8      2
1      3      4
5      6      7
BUILD SUCCESSFUL (total time: 10 seconds)
|

```

## Laboratorio 69. Desarrollo de programa para imprimir una matriz de datos de 2 x 4

Desarrollar un programa que obtenga un arreglo de 2x4 “llamado B1” con datos ya establecidos por el usuario e imprimir el arreglo.

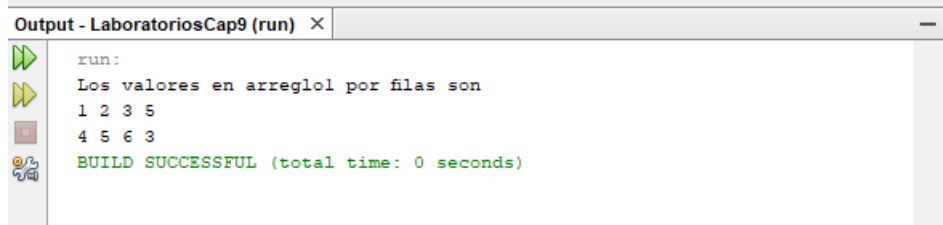
### Codificación

```
package arreglo1;
/**
 *
 * @author jxs_b
 */
public class Arreglo1 {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        int arreglo169[] [] = {{1, 2, 3 ,5}, {4, 5, 6 ,3}};
        System.out.println("Los valores en arreglo1 por filas son");
        imprimirArreglo69 (arreglo169);
    }
    public static void imprimirArreglo69 (int arreglo69 [] []) {
        for (int [] arreglo169: arreglo69) {
            for (int columna69 = 0; columna69 < arreglo169.length; columna69++) {
                System.out.printf("%d ", arreglo169[columna69]);
            }
            System.out.println();
        }
    }
}
```

## Ejecución

### Figura 69

*Ejecución del Laboratorio 69*



```
Output - LaboratoriosCap9 (run) ×
run:
Los valores en arreglo1 por filas son
1 2 3 5
4 5 6 3
BUILD SUCCESSFUL (total time: 0 seconds)
```

Laboratorio 70. Desarrollo de programa que sume los datos del primer renglón y la suma de la tercera columna de una matriz de 3 x 3

Hacer un programa que obtenga la suma del renglón 1 e imprima el resultado, además debe obtener la suma de la columna 3 de un arreglo bidimensional de 3 x 3.

## Codificación

```
package arreglo3.pkg8abril;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Arreglo70abril {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner entrada70 = new Scanner (System.in);
        int i70, j70, fila70, columna70, sumafila70=0, sumacolumna70=0;

        int arreglo70 [] []= new int [3][3];
        System.out.println("Ingresar los valores del arreglo:");
        for (i70=0; i70<arreglo70. length; i70++)
        {
            System.out.println();
            for (j70=0; j70<arreglo70. length; j70++) {
```

```

        System.out.print("[ "+i+" ]"+"[ "+j+" ]"+"=");
        Arreglo70[i70][j70]=entrada70. nextInt ();
    }
}
for (i70=0; i70<arreglo. length; i70++) {
    System.out.println();
    for (j70=0; j70<arreglo. length; j70++) {
        System.out.print(arreglo70[i70][j70]+"\\t");

    }

}
System.out.print("SUMA DEL RENGLÓN:");
fila70=entrada70. nextInt ();
for (i70=0; i70<arreglo70. length; i70++) {
    sumafila70=sumafila70 + arreglo70[fila70-1][i70];

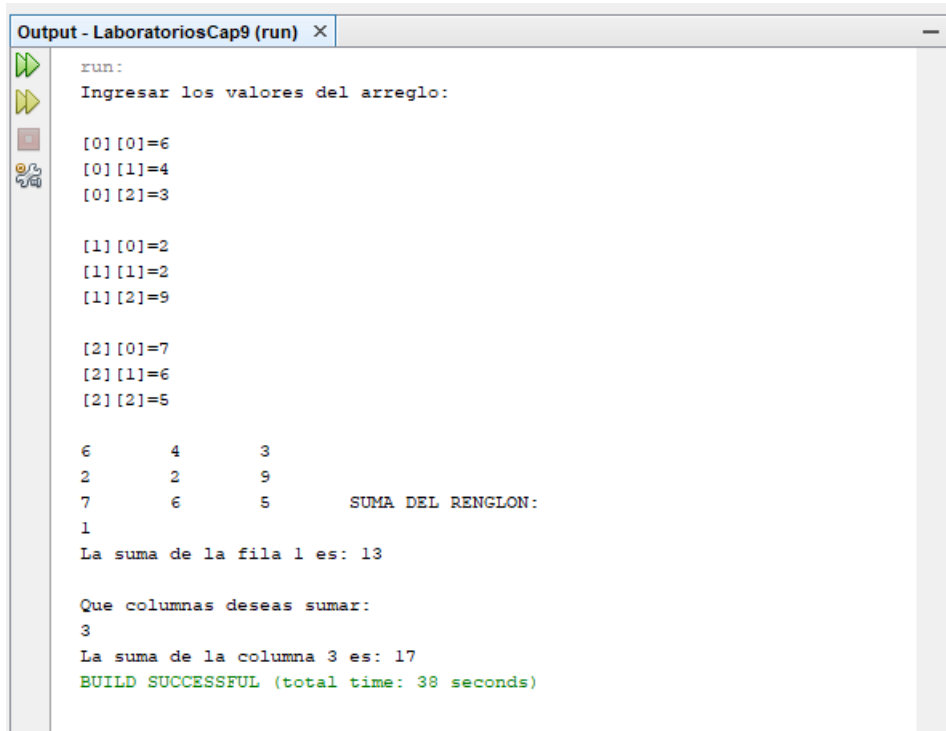
}
System.out.println("La suma de la fila "+fila+" es: "+sumafila70);
System.out.println();
System.out.println("Que columnas deseas sumar: ");
Columna70=entrada70. nextInt ();
for (i70=0; i70<arreglo. length; i70++) {
    sumacolumna70=sumacolumna70 + arreglo70[i70][columna70-1];
}
System.out.println("La suma de la columna "+columna70+" es: "+sumacolumna70);
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

### Figura 70

*Ejecución del Laboratorio 70*



```
Output - LaboratoriosCap9 (run) x
run:
Ingresar los valores del arreglo:

[0][0]=6
[0][1]=4
[0][2]=3

[1][0]=2
[1][1]=2
[1][2]=9

[2][0]=7
[2][1]=6
[2][2]=5

6      4      3
2      2      9      SUMA DEL RENGLON:
7      6      5
1
La suma de la fila 1 es: 13

Que columnas deseas sumar:
3
La suma de la columna 3 es: 17
BUILD SUCCESSFUL (total time: 38 seconds)
```

Laboratorio 71. Desarrollo de programa que calcule la suma de los datos de la diagonal de una matriz con datos

Implementar un programa que sume la diagonal principal e imprimirla.

## Codificación

```
package arreglo4.pkg8abril;
```

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

```
/**
```

```
*
```

```
* @author jxs_b
```

```
*/
```

```
public class Arreglo48abril {
```

```
/**
```

```

* @param args instrucción para el traspaso de argumentos
*/
public static void main (String [] args) {
    Scanner sc71 = new Scanner (System.in);
    int suma71 = 0;
    int A171[] [];

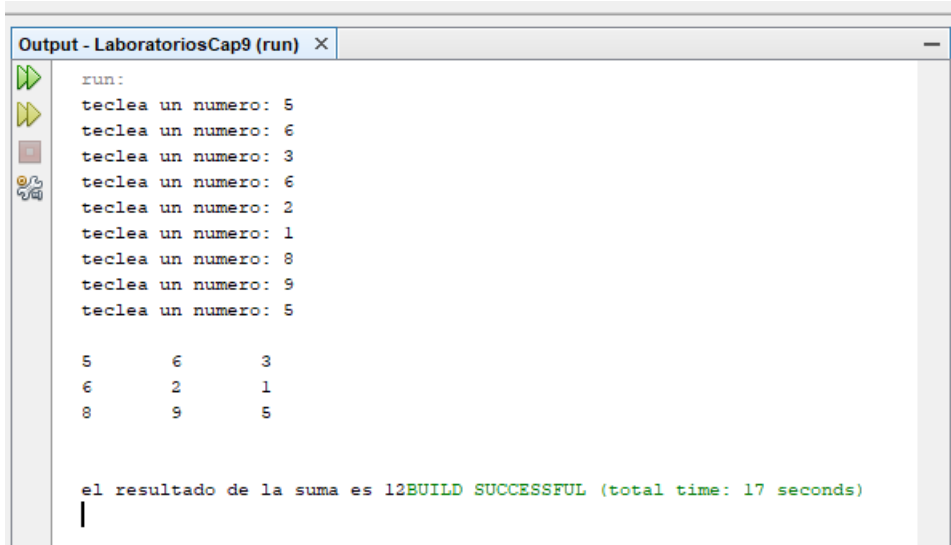
    A171=new int [3][3];
    for (int x71=0; x71<=2; x71++) {
        for (int y71=0; y71<=2; y71++) {
            System.out.printf("teclea un numero: ");
            A171[x][y]=sc.nextInt ();
        }
    }
    for (int x71=0; x71<=2; x71++) {
        System.out.println();
        for (int y71=0; y71<=2; y71++) {
            System.out.printf("%d\t", A171[x71][y71]);
        }
    }
    for (int x71=0; x71<=2; x71++) {
        System.out.println();
        for (int y71=0; y71<= 2; y71++) {
            if(x71==y71)
                suma71=suma71+A171[x71][y71];
        }
    }
    System.out.printf("el resultado de la suma es %d", suma71);
    // Se aplica el código lógico aquí:
}
}

```

## Ejecución

Figura 71

Ejecución del Laboratorio 71



The screenshot shows an IDE output window titled "Output - LaboratoriosCap9 (run)". The output text is as follows:

```
run:
teclea un numero: 5
teclea un numero: 6
teclea un numero: 3
teclea un numero: 6
teclea un numero: 2
teclea un numero: 1
teclea un numero: 8
teclea un numero: 9
teclea un numero: 5

5      6      3
6      2      1
8      9      5

el resultado de la suma es 12BUILD SUCCESSFUL (total time: 17 seconds)
```

Laboratorio 72. Desarrollo de programa que calcule el mayor de los elementos de la diagonal de una matriz con datos

Desarrollar un programa que obtenga de un arreglo bidimensional el mayor de la diagonal principal e imprimir resultados.

## Codificación

```
package arreglo5.pkg8abril;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Arreglo72abril {
    /**
     * @param args instrucción para el traspaso de argumentos
     */
    public static void main (String [] args) {
        Scanner sc72 = new Scanner (System.in);
        int A172[] [];
```



```

int mayor72 =0;
A172 = new int [3][3];

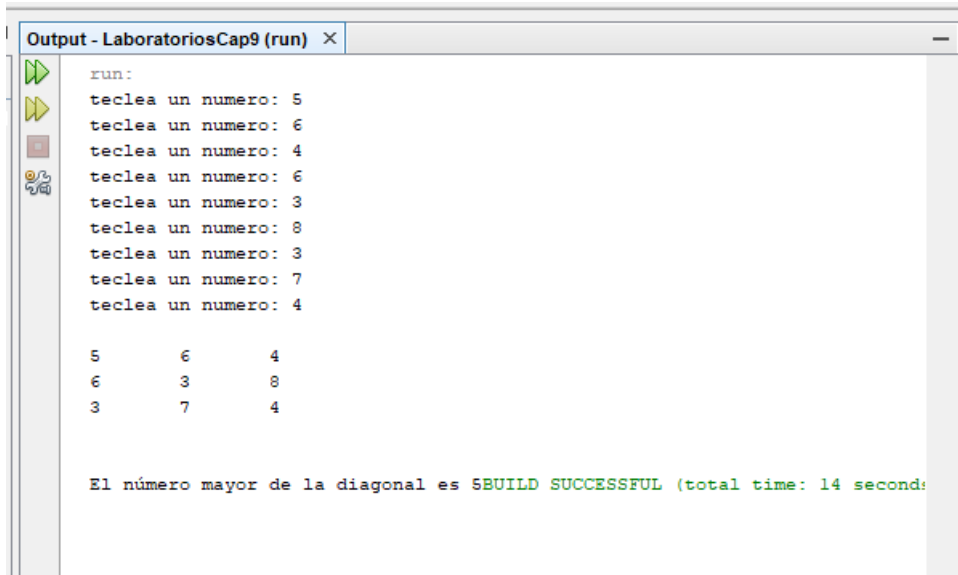
for (int x72=0; x72<=2; x72++) {
for (int y72=0; y72<=2; y72++) {
    System.out.printf("teclea un número: ");
    A172[x72][y72]=sc72. nextInt ();
}
}
for (int x72=0; x72<=2; x72++) {
    System.out.println();
    for (int y72=0; y72<=2; y72++) {
        System.out.printf("%d\t", A172[x72][y72]);
    }
}
for (int x72=0; x72<=2; x72++) {
    System.out.println();
    for (int y72=0; y72<=2; y72++) {
        if((x72==y) &&(A172[x][y]>mayor72))
        {
            Mayor72=A172[x72][y72];
        }
        else
            System.out.printf("");
    }
}
System.out.printf("El número mayor de la diagonal es %d", mayor72);
// Se aplica el código lógico aquí:
}
}

```

## Ejecución

Figura 72

Ejecución del Laboratorio 72



```
run:
teclea un numero: 5
teclea un numero: 6
teclea un numero: 4
teclea un numero: 6
teclea un numero: 3
teclea un numero: 8
teclea un numero: 3
teclea un numero: 7
teclea un numero: 4

5      6      4
6      3      8
3      7      4

El número mayor de la diagonal es 5BUILD SUCCESSFUL (total time: 14 seconds)
```

Laboratorio 73. Desarrollo de programa calcule el mayor de los elementos nones de una matriz con datos

Crear un programa que obtenga el mayor de los números nones de todo el arreglo bidimensional capturados por el usuario mediante el teclado e imprimir el resultado.

## Codificación

```
package arreglo6.pkg8abril;
import java.util. Scanner;
/**
 *
 * @author jxs_b
 */
public class Arreglo73abril {

    /**
     * @param args instrucción para el traspaso de argumentos */
    public static void main (String [] args) {
        Scanner sc73 = new Scanner (System.in);
```

```

int mayor73 = 0;
int A173[] [];
A173 = new int [3][3];
int arreglo173[];
arreglo173 = new int [5];
int a73 =0;
    for (int x73=0; x73<=2; x73++) {
for (int y73=0; y73<=2; y73++) {
    System.out.printf("teclea un número: ");
    A173[x73][y73]=sc73. nextInt ();
}
}
    for (int x73=0; x73<=2; x73++) {
        System.out.println();
        for (int y73=0; y73<=2; y73++) {
System.out.printf("%d\t", A173[x73][y73]);

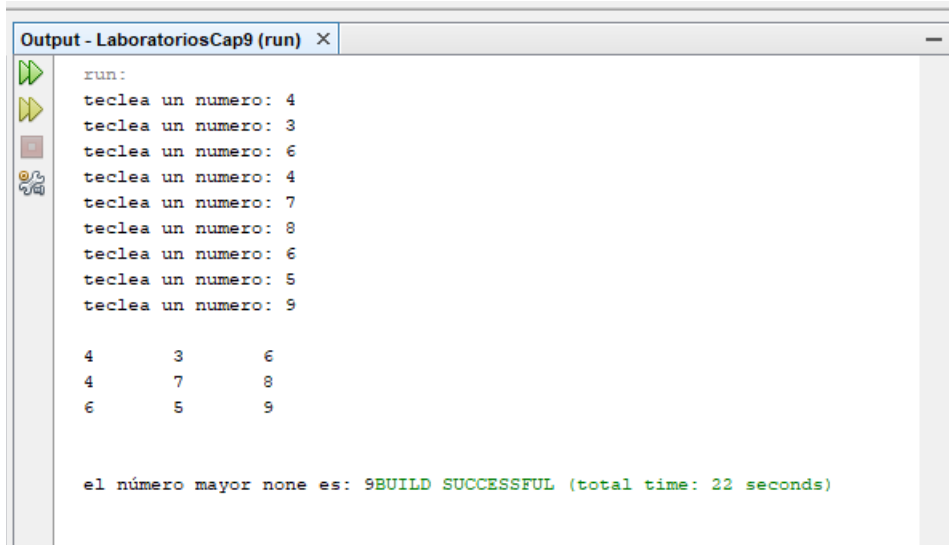
        }
}
    for (int x73=0; x73<=2; x73++) {
        System.out.println();
        for (int y73=0; y73<=2; y73++) {
            if(A173[x73][y73]%2==1) {
                if(A173[x73][y73]>arreglo173[a73])
                    arreglo173[a73]=A173[x73][y73];
            }
        }
    }
    // Se aplica el código lógico aquí:
}
System.out.printf("el número mayor none es: %d", arreglo173[a73]);
}
}

```

## Ejecución

Figura 73

Ejecución del Laboratorio 73



```
run:
teclea un numero: 4
teclea un numero: 3
teclea un numero: 6
teclea un numero: 4
teclea un numero: 7
teclea un numero: 8
teclea un numero: 6
teclea un numero: 5
teclea un numero: 9

4      3      6
4      7      8
6      5      9

el número mayor none es: 9BUILD SUCCESSFUL (total time: 22 seconds)
```

Laboratorio 74. Desarrollo de programa que da los totales de cartas de póker

## Codificación

```
package carta;
/**
 *
 * @author Jesús Ramos
 */
public class Carta {

    /**
     * @param args instrucción para el traspaso de argumentos
     */
    private String cara74; // cara de la carta ("As", "Dos", ...)
    private String palo74; // palo de la carta ("Corazones", "Diamantes", ...)
    // el constructor de dos argumentos inicializa la cara y el palo de la carta
    public Carta74 (String caraCarta74, String paloCarta74)
    {
```

```

        Cara74 = caraCarta74;// inicializa la cara de la carta
        Palo74 = paloCarta74; // inicializa el palo de la carta
    } // fin del constructor de carta con dos argumentos
    // devuelve representación String de Carta
    public String toString ()
    {
        return cara74 + " de " + palo74;
    } // fin del método toString
} // fin de la clase Carta
package carta;
import java. util. Random;
/**
 *
 * @author Jesús Ramos
 */
public class PaqueteDeCartas74
{
    private Carta paquete74 [];
    // arreglo de objetos Carta
    private int cartaActual74; // subíndice de la siguiente carta a repartir
    private int NÚMERO_DE_CARTAS74 = 52; // número constante de cartas
    private Random númerosAleatorios74; // generador de números aleatorios
    // el constructor llena el paquete de cartas

    public PaqueteDeCartas74 ()
    {
        String caras74 [] = {"As", "Dos", "Tres", "Cuatro", "Cinco", "Seis",
            "Siete", "Ocho", "Nueve", "Diez", "Jack", "Queen", "Rey"};
    };
        String palos74 [] = {"Corazones", "Diamantes", "Tréboles", "Espadas"};
    Paquete74 = new Carta74 [ NÚMERO_DE_CARTAS74]; // crea arreglo de objetos carta
    cartaActual74 = 0; // establece cartaActual para que la primera carta repartida sea
    paquete [ 0]
    númerosAleatorios74 = new Random (); // crea generador de números aleatorios
    // llena el paquete con objetos carta
    for (int cuenta74 = 0; cuenta74 < paquete74. length; cuenta74++)
    {
    Paquete74 [ cuenta74] =

```

```

new Carta74 (caras74 [ cuenta74 % 13], palos74 [ cuenta74 / 13]);
}
// fin del constructor de PaqueteDeCartas
// baraja el paquete de Cartas con algoritmo de una pasada
public void barajar ()
{
// después de barajar, la repartición debe empezar en paquete [ 0] otra vez
cartaActual74 = 0; // reinicializa cartaActual
// para cada Carta, selecciona otra Carta aleatoria y las intercambia
for (int primera74 = 0; primera74 < paquete74. length; primera74++)
{
// selecciona un número aleatorio entre 0 y 51
int segunda74 = numerosAleatorios74.nextInt(NÚMERO_DE_CARTAS74);
// intercambia Carta actual con la Carta seleccionada al azar
Carta temp74 = paquete74 [ primera74];
Paquete74 [ primera74] = paquete74 [ segunda74];
Paquete74 [ segunda74] = temp74;} // fin de for
} // fin de método barajar
// reparte una carta
public Carta repartirCarta74 ()
{
// determina si quedan Cartas por repartir
if (carta Actual74 < paquete74. length)
return paquete74 [ carta Actual74++]; // devuelve la carta actual en el arreglo
else
return null; // devuelve null para indicar que se repartieron todas las cartas
} // fin del método repartirCarta
} // fin de la clase PaqueteDeCartas
package carta;

/**
 *
 * @author Jesús Ramos
 */
public class PruebaDePaqueteDeCartas
{
public static void main (String args [])
{

```

```

PaqueteDeCartas miPaqueteDeCartas74 = new PaqueteDeCartas74 ();

miPaqueteDeCartas74.barajar(); // coloca las cartas en orden aleatorio
// imprime las 52 Cartas en el orden en el que se reparten
for (int i74 = 0; i74 < 13; i74++)
{
    // reparte e imprime 4 Cartas
    System.out.printf("%-20s%-20s%-20s%-20s\n",
        miPaqueteDeCartas74.repartirCarta74(), miPaqueteDeCartas74.
repartirCarta74(),
        miPaqueteDeCartas74.repartirCarta74(), miPaqueteDeCartas74.
repartirCarta74());
    } // fin de for 20
}
}

```

## Ejecución

**Figura 74**

*Ejecución del Laboratorio 74*

```

Output - LaboratoriosCap9 (run) x
run:
Siete de Diamantes      Tres de Diamantes      Siete de Corazones     Cinco de Diamantes
Quina de Diamantes     Cuatro de Espadas     Seis de Diamantes      Cuatro de Corazones
Dos de Espadas         Cuatro de Diamantes  Diez de Espadas        Seis de Espadas
Rey de Corazones      Tres de Treboles     Rey de Espadas         Quina de Treboles
Rey de Treboles       Siete de Treboles    Cinco de Espadas       Ocho de Espadas
Cinco de Corazones    Nueve de Espadas     Diez de Corazones      Seis de Corazones
Nueve de Corazones    Joto de Corazones    As de Treboles         Nueve de Treboles
Cuatro de Treboles    Quina de Corazones   Joto de Treboles       Nueve de Diamantes
Ocho de Treboles      Cinco de Treboles    Seis de Treboles      Quina de Espadas
As de Espadas         Joto de Diamantes    Rey de Diamantes       Ocho de Diamantes
Tres de Espadas       Diez de Treboles     Diez de Diamantes      Dos de Treboles
Siete de Espadas      As de Diamantes      Joto de Espadas        As de Corazones
Ocho de Corazones     Dos de Corazones     Tres de Corazones      Dos de Diamantes
BUILD SUCCESSFUL (total time: 0 seconds)

```

## 9.4. Ejercicios para resolver. Casos prácticos

Desarrollar un arreglo de 4\*4 que lea una tabla de números y sume las columnas de cada fila y las filas de cada columna.

Crear un arreglo de 4\*4 que lea los datos mediante el teclado y obtenga la suma de la diagonal de derecha a izquierda.

Haga un arreglo de 4\*4 que lea los datos mediante el teclado y obtenga el promedio de la suma de la diagonal de derecha a izquierda.

Implemente un arreglo de 4\*4 que lea los datos mediante el teclado y obtenga el menor de los números pares e imprimirlo.

Llevar a cabo un programa que cree dos arreglos de 4\*4 llamado A y B. y obtenga la suma de los dos arreglos, pero los números pares.

## 9.5. Referencias de consulta

<https://dcodingames.com/arreglos-bidimensionales-java/>

[https://www.aprenderaprogramar.com/index.php?option=com\\_content&view=article&id=233:arrays-arreglos-multidimensionales-en-java-declaracion-y-uso-ejemplos-y-ejercici](https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=233:arrays-arreglos-multidimensionales-en-java-declaracion-y-uso-ejemplos-y-ejercici)

<https://es.slideshare.net/danihotmailcom/arreglos-bidimensionales-java-netbeans>

<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/while>

## 9.6. Video de apoyo

[https://www.youtube.com/watch?v=\\_tUncS0AsNE](https://www.youtube.com/watch?v=_tUncS0AsNE)

<https://www.youtube.com/watch?v=SMIyVS3k848>

<https://www.youtube.com/watch?v=bPVcx44wgmQ>



## Referencias

- Ceballos, J. (2008). *Java*. Editorial Ra-Ma.
- Conahcyt (2023) *Programas Nacionales Estratégicos, Educación* obtenidos de: <https://conahcyt.mx/pronaces/pronaces-educacion/>
- Dean, J. (2019) *Web Programming with HTML5, CSS, and JavaScript*. Jones & Bartlett Publishers. <https://es.llib.mx/book/3698316/3e1f79>
- Deitel, D. (2008). *Cómo Programar Java*. Pearson Educación.
- Deitel, H. (2007). *C# Cómo Programar*, 2ª. Ed. (L. M. Castillo, Ed.) Pearson Educación.
- Geeksforgeeks (s.f). *Cursos para geeks*, <https://www.geeksforgeeks.org/>
- González, J. (2018). *Programar Ya*. Obtenido de Programar Ya: <https://www.programarya.com/Cursos/Java/Ciclos/Ciclo-for>
- Gottfried, B. (1994). *Programación Pascal*, 1ª. Ed. Mc Graw Hill.
- Jiménez, A. y Pérez, F. (2006). *Aprende a programar en JAVA*, 2ª. Ed., Ediciones Paraninfo, S.A.
- Márquez, T., Osorio, S. y Olvera, E. (2011). *Introducción a la programación estructurada en C*, 1ª. Ed. Pearson.
- Programación Ya (s.f). *Estructura repetitiva do while*. <https://www.tutorialesprogramacionya.com/javaya/detalleconcepto.php?codigo=85&punto=&inicio=>
- Sznajdleder, P. (2013). *Java a fondo: estudio del lenguaje y desarrollo de aplicaciones*, 2ª. Ed. Alfaomega.



## De los autores

### José Rafael Baca Pumarejo

Doctor en Educación Internacional por la Universidad Autónoma de Tamaulipas. Profesor Investigador de Tiempo Completo de la Universidad Autónoma de Tamaulipas, adscrito a la Facultad de Comercio y Administración Victoria. Perfil Prodep. Pertenece al SNI Nivel 1. Líder del Cuerpo Académico Consolidado Nuevas Tecnologías, Capital Humano y Competitividad. Línea de Investigación (LGAC): Nuevas Tecnologías para la Gestión del Conocimiento, la Competitividad y la Innovación.

Correo electrónico: *rbaca@docentes.uat.edu.mx*

### Vicente Villanueva Hernández

Doctor en Investigación Educativa por la Escuela Normal Superior de Ciudad Madero. Profesor de Tiempo Completo de la Unidad Académica Multidisciplinaria Reynosa Rodhe de la Universidad Autónoma de Tamaulipas. Perfil Prodep. Pertenece al SNI Nivel 1. Integrante del Cuerpo Académico Consolidado Nuevas Tecnologías, Capital Humano y Competitividad. Línea de Investigación (LGAC): Nuevas Tecnologías para la Gestión del Conocimiento, la Competitividad y la Innovación.

Correo electrónico: *vhernand @docentes.uat.edu.mx*



*Aprender y practicar lenguaje de programación Java*  
de José Rafael Baca Pumarejo y Vicente Villanueva Hernández,  
publicado por la Universidad Autónoma de Tamaulipas en octubre de 2023. La  
revisión y diseño editorial correspondieron al Consejo de Publicaciones UAT.

Este libro de programación intermedia avanzada contribuye a desarrollar las habilidades de programación de los alumnos inmersos en el lenguaje de programación Java. Incide en el perfil profesional de los inscritos en la carrera de Tecnologías de la Información al proporcionarles los elementos necesarios para seguir con éxito el aprendizaje secuencial de programación avanzada que les permita diseñar aplicaciones destinadas al sector productivo.

Este libro teórico-práctico incide en el Programa Nacional Estratégico de Educación (Pronaces-Educación) y representa un factor innovador porque elevará la calidad de la educación y entrenamiento en esta área de la informática tan importante para la competitividad de nuestros estudiantes. La obra enfatiza la reflexión sobre la práctica al abordar la actividad de programación de los alumnos encaminada al abordaje de la tecnología representada por este superlenguaje computacional.

ISBN 978-607-8888-22-1



ISBN UAT: 978-607-8888-22-1