

UNIVERSIDAD CATOLICA BOLIVIANA
"SAN PABLO"
FACULTAD DE CIENCIAS EXACTAS E INGENIERIA
INGENIERIA DE SISTEMAS
LA PAZ – BOLIVIA

PROGRAMACION II GUIA DE JAVA + EJERCICIOS 2007

INTEGRANTES:

OSSIO MARIN SERGIO WILLY
PRUDENCIO ROBINSON ANDRES MAURICIO
SALGADO FALLER IGNACIO
VALDA SANCHEZ FREDDY SALVADOR

DOCENTE: M.Sc. IRMA PRADO

SEMESTRE I AÑO 2007

INDICE

INDICE	1
PROLOGO	2
INTRODUCCIÓN.....	3
Programacion Orientada a Objetos.....	4
Particularidades de Java.....	5
El interprete de Java	8
Diferencias y similitudes con C++	11
Clases útiles	16
Lenguaje Java.....	19
EJERCICIOS RESUELTOS.....	22
Sentencias Básicas (Condicionales, Aritmeticas y Contadores).....	22
PROBLEMAS PROPUESTOS.....	28
Ciclos y Series.....	30
PROBLEMAS PROPUESTOS.....	42
Funciones y el uso de la clase Math	44
PROBLEMAS PROPUESTOS.....	57
Arreglos	59
Aplicación de vectores – Programa estadistico	83
PROBLEMAS PROPUESTOS.....	89
Matrices	91
Aplicación de Matrices – Calculos Matriciales	115
Aplicación de matrices – Inversion de Matrices por Faddevva	123
PROBLEMAS PROPUESTOS.....	126
Recursividad.....	129
PROBLEMAS PROPUESTOS.....	148
Ejercicios con Gráficos.....	149
BIBLIOGRAFIA	160

PROLOGO

Esta guía fue realizada por estudiantes de tercer semestre de Ingeniería de Sistemas de la Universidad Católica Boliviana con el fin de proporcionar una herramienta útil a todos los estudiantes que estén cursando la materia de Programación II o que tengan alguna dificultad en la programación en Java o que simplemente deseen ampliar su conocimiento en el tema.

En esta guía encontrará aproximadamente 100 ejercicios resueltos (programas en java), clasificados en diferentes secciones. Todos estos programas funcionan correctamente y fueron ejecutados utilizando la herramienta *Ready to Program*, disponible en forma gratuita en

<http://www.sju.edu/~parkinso/javatools/readytoprogram.html>

INTRODUCCIÓN

En el año 1991 nació Java cuando en *Sun*, una empresa nueva, se propusieron crear un nuevo lenguaje destinado en principio a electrodomésticos, por lo tanto, la primera versión era un lenguaje sencillo capaz de generar código de tamaño reducido.

El continuo cambio en las computadoras obligaba a cambiar los programas continuamente por lo que se pensó en desarrollar una máquina virtual independiente que posteriormente se denominaría *Java Virtual Machine* (JVM) y esto dio lugar al primer lema de Java "*Write once, Run Everywhere*".

Evidentemente, los electrodomésticos no estaban interesados en Java, así que comenzó a introducirse como lenguaje de programación para computadoras a finales de 1995, introduciéndose como una herramienta necesaria, puesto que Netscape Navigator 2.0 la introdujo como intérprete.

Se publicó *Java 1.1* a principios de 1997 y *Java 1.2* a finales de 1998 con una evolución prodigiosa de 12 paquetes iniciales, 23 paquetes en la versión 1.1 y 59 paquetes en la versión 1.2 (denominada Java 2).

La compañía *Sun* describe a *Java* como "simple, orientado a objetos, distribuido, interpretado, robusto, seguro, de arquitectura neutra, portable, de altas prestaciones, multitarea y dinámico", y lo bueno de esta definición es que todo lo que dice, que no es poco, es absolutamente cierto, aunque en algunos aspectos mejorable.

Programacion Orientada a Objetos

La principal función de este tutorial no es enseñar todos y cada uno de los conceptos y particularidades de la programación dirigida a objetos, así que simplemente se darán algunos conceptos que se deben tener claros para comenzar.

La estructura general de un programa orientado a objetos (POO), se especifica en una clase principal que contienen el programa principal, (función *main()*).

Una clase es una colección de variables y métodos que manejan variables y estructuras de datos. Se denominan normalmente datos y métodos miembro.

La POO se basa en la programación de clases. Una clase constituye un patrón o modelo para crear objetos. Todos los objetos deben pertenecer a una clase determinada y por lo tanto se dice que todos los objetos deben ser instancias de una clase, es decir, que los objetos forman un contexto de clase. Un objeto estará formado por las variables y métodos de la clase y será siempre diferente de otro objeto, aunque este pertenezca a la misma clase. Las clases también pueden incluir variables *static* que no formarán parte de los objetos de dicha clase, sino de la clase en sí, estas variables se denominan variables de clase.

La herencia es otro concepto importante de la POO, permite definir nuevas clases basándose en clases ya existentes. Una clase que *se extiende* de otra, hereda todas sus variables y todos sus métodos. La clase derivada puede añadir nuevas variables y métodos y redefinir las variables y métodos heredados. En Java una clase sólo puede heredar de otra, es lo que se denomina herencia simple.

El último de los conceptos claves que se van a puntualizar aquí es el *polimorfismo*, el polimorfismo tiene que ver con la relación que se establece

entre la llamada a un método y el código que efectivamente se asocia con dicha llamada. A esta relación se llama *vinculación o binding*. La vinculación puede ser temprana, en tiempo de compilación, o tardía, en tiempo de ejecución.

La vinculación temprana es la más eficiente y es la que se da normalmente con funciones normales o sobrecargadas. La *sobrecarga* se refiere a la posibilidad de tener dos o más funciones con el mismo nombre pero funcionalidad diferente. Es decir, dos o más funciones con el mismo nombre realizan acciones diferentes. El compilador usará una u otra dependiendo de los parámetros usados. A esto se llama sobrecarga de funciones. La vinculación tardía es la que se utiliza con funciones redefinidas.

Además de clases, Java proporciona interfaces. Un *interface* es un conjunto de declaraciones de métodos. Si una clase implementa un *interface*, debe definir todas las funciones especificadas en él. Una clase puede implementar ningún, uno, o varios *interfaces*. Un interface si soporta la herencia múltiple de varios interfaces.

Particularidades de Java

A continuación se enuncian varias particularidades de Java, que servirán al programador de este lenguaje a entender mejor la estructura para crear un programa en Java de manera exitosa.

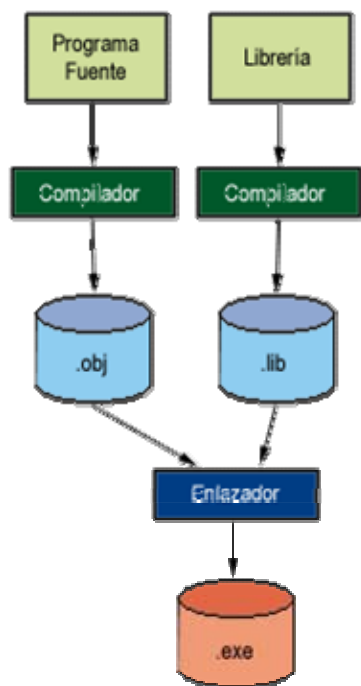
- La sintaxis es muy parecida a C++ y las palabras clave son las mismas eliminando algunas en desuso y añadiendo otras para realizar cosas características de Java.
- En Java es habitual utilizar nombres con minúsculas para variables siempre y cuando no se den algunas de las siguientes situaciones:
 - El nombre consta de varias palabras y normalmente se pone una detrás de otra con la primera letra de cada palabra en mayúsculas:
Ej. elMayor.

- o Los nombres de clases e interfaces comienzan siempre por mayúsculas: Ej. Geometría.
 - o Los nombres de objetos, métodos, variables miembro y variables locales de los métodos comienzan siempre por minúscula.
 - o Los nombres de las variables finales, o constantes, se definen siempre con mayúsculas: Ej. PI.
 - o Todas las variables en Java son referencias y la utilización de la memoria asociada a éstas y a los objetos que se manejan es una función del recolector de basura.
- Los ficheros de código deben tener la extensión *java* y tendrán el mismo nombre que la clase principal que contienen. Estos ficheros pueden contener más de una clase, pero sólo una de ellas será una clase pública y se denomina clase principal.
- Los ficheros ya compilados tienen la extensión *class* y existe siempre un fichero de este tipo por clase.
- Una aplicación está compuesta por uno o varios ficheros *class* permitiendo así modularizar cada una de las partes de la aplicación en ficheros. La ejecución de una aplicación comienza siempre por una clase que contiene la función *main()* sin añadir la extensión *class*.
- En Java todas las clases heredan, aunque no se especifique explícitamente de la clase general *Object* que hace de raíz de toda la jerarquía de clases de Java. Los métodos que define la clase *object* son de dos clases, métodos que se pueden redefinir por el programador, *clone()*, *equals()*, *toString()*, *finalize()* y métodos que no pueden ser redefinidos, son métodos *final* como *getClass()*, *notify()*, *notifyAll()* aunque los tres últimos están relacionados con las hebras, *threads*.

- Java es un lenguaje fuertemente tipificado y la conversión entre referencias de clases diferentes exige que ambas clases estén relacionadas mediante herencia o mediante la implementación de un interfaz en el caso de que la referencia sea de un tipo *interface*.
- Las clases *Java* se agrupan en *packages* a los que se pueden definir como librerías de clases. Si se declara una clase sin *package* se consideran pertenecientes a un *package* por defecto, *default* que es el directorio actual. Java proporciona un conjunto de paquetes al que se denomina *API*. El nombre de los paquetes suele ser un nombre en minúsculas y puede constar de varios nombres unidos por puntos. Todas las clases que están en el mismo paquete deben estar en el mismo directorio. Los nombres compuestos de los paquetes están relacionados con la jerarquía de directorios en que se guardan las clases. Los paquetes se utilizan para almacenar clases relacionadas, evitar conflictos de nombres y ayudar a la accesibilidad de las clases. La inclusión de una clase en un paquete se hace mediante la sentencia *package* que se incluye en la primera línea del fichero java que contiene la clase. la sentencia *import* seguida del nombre de un paquete más el nombre de una clase de ese paquete, indica que clase se utilizará en ese fichero cuando se invoque a un objeto de la clase importada.
- Al contrario de C++, Java no utiliza destructores, sino que opcionalmente el usuario puede incluir n método que se ejecutará cuando el objeto vaya a ser reciclado, es el método *finalize()*.

El interprete de Java

Java, para conseguir ser un lenguaje independiente del sistema operativo y del procesador que incorpore la máquina utilizada, es tanto interpretado como compilado. El código fuente escrito con cualquier editor se compila generando el ByteCode. Este código intermedio es de muy bajo nivel, pero sin alcanzar las instrucciones máquina propias de cada plataforma y no



tiene nada que ver con el p-code de Visual Basic.

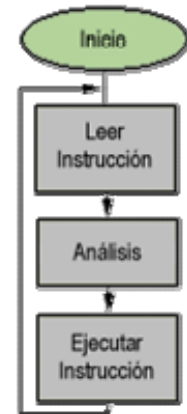
El ByteCode corresponde al 80% de las instrucciones de la aplicación. Ese mismo código es el que se puede ejecutar sobre cualquier plataforma. Para ello hace falta el runtime, que sí es completamente dependiente de la máquina y del sistema operativo que interpreta dinámicamente el ByteCode y añade el 20% de instrucciones que faltaban para su ejecución. Con este sistema es fácil crear aplicaciones multiplataforma, pero para ejecutarlas es necesario que exista el runtime correspondiente al sistema operativo utilizado.

No obstante, este panorama está cambiando a pasos agigantados, y aunque Java sigue siendo básicamente un lenguaje interpretado, la situación se acerca mucho a la de los programas compilados, sobre todo en lo que a la rapidez en la ejecución del código se refiere. Para comprender el funcionamiento de HotSpot, que es el último lanzamiento que hace Sun y que promete interpretar los ByteCodes más rápido que un programa compilado, las siguientes imágenes muestran cómo actúa el sistema runtime en los diversos casos que hasta ahora pueden darse.

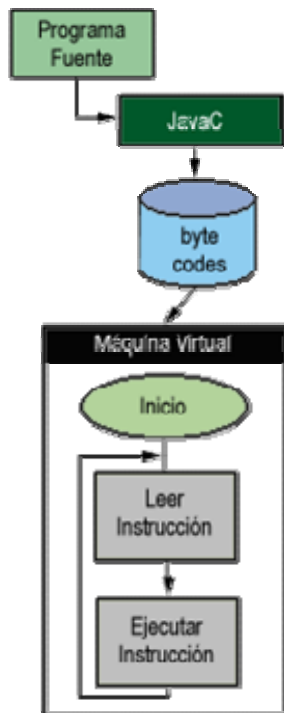
La imagen de la izquierda muestra las acciones correspondientes a un compilador tradicional. El compilador traslada las sentencias escritas en lenguaje de alto-nivel a múltiples instrucciones, que luego son enlazadas junto

con el resultado de múltiples compilaciones previas que han dado origen a librerías, y juntando todo ello, es cuando genera un programa ejecutable.

La imagen de la derecha muestra la forma de actuación de un intérprete. Básicamente es un enorme bucle, en el cual se va leyendo o recogiendo cada una de las instrucciones del programa fuente que se desea ejecutar, se analiza, se parte en trozos y se ejecuta. Luego se va a recoger la siguiente instrucción que se debe interpretar y se continúa con este proceso hasta que se terminan las



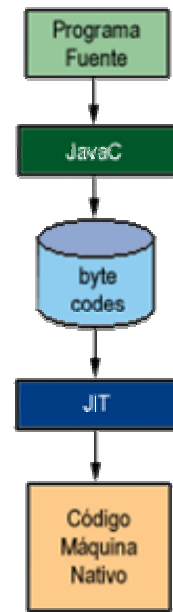
instrucciones o hasta que entre las instrucciones hay alguna que contiene la orden de detener la ejecución de las instrucciones que componen el programa fuente.



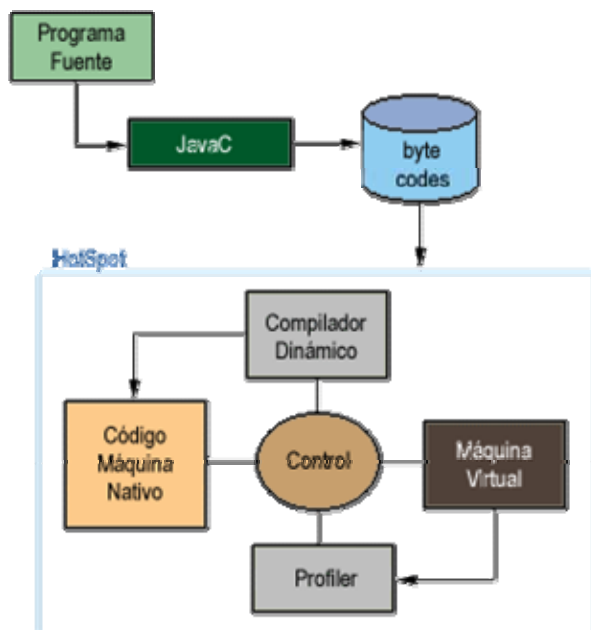
La imagen siguiente, situada a la izquierda, muestra un tipo de intérprete más eficiente que el anterior, el intérprete de ByteCodes, que fue popularizado hace más de veinte años por la Universidad de California al crear el UCSD Pascal. En este caso, el intérprete trabaja sobre instrucciones que ya han sido trasladadas a un código intermedio en un paso anterior. Así, aunque se ejecute en un bucle, se elimina la necesidad de analizar cada una de las instrucciones que componen el programa fuente,

porque ya lo han sido en el paso previo. Este es el sistema que Java utiliza, y la Máquina Virtual Java es un ejemplo de este tipo de intérprete. No obstante, sigue siendo lento, aunque se obtenga un código independiente de plataforma muy compacto, que puede ser ejecutado en cualquier ordenador que disponga de una máquina virtual capaz de interpretar los ByteCodes trasladados.

Un paso adelante en el rendimiento del código Java lo han representado los compiladores Just-In-Time, que compilan el código convirtiéndolo a código máquina antes de ejecutarlo. Es decir, un compilador JIT va trasladando los ByteCodes al código máquina de la plataforma según los va leyendo, realizando un cierto grado de optimización. El resultado es que cuando el programa se ejecute, habrá partes que no se ejecuten y que no serán compiladas, y el compilador JIT no perderá el tiempo en optimizar código que nunca se va a ejecutar. No obstante, los compiladores JIT no pueden realizar demasiadas optimizaciones, ya que hay código que ellos no ven, así que aunque siempre son capaces de optimizar la parte de código de inicialización de un programa, hay otras partes que deben ser optimizadas, según se van cargando, con lo cual, hay una cierta cantidad de tiempo que inevitablemente ha de perderse.



Y, finalmente, se presenta la última tendencia en lo que a compilación e intérpretes se refiere. Lo último en que trabaja Sun es HotSpot, una herramienta



que incluye un compilador dinámico y una máquina virtual para interpretar los ByteCodes, tal como se muestra en la figura siguiente, situada a la izquierda de la página. Cuando se cargan los ByteCodes producidos por el compilador por primera vez, éstos son interpretados en la máquina virtual. Cuando ya están en ejecución, el profiler mantiene información sobre el rendimiento y selecciona el método sobre el que

se va a realizar la compilación. Los métodos ya compilados se almacenan en

un caché en código máquina nativo. Cuando un método es invocado, esta versión en código máquina nativo es la que se utiliza, en caso de que exista; en caso contrario, los ByteCodes son reinterpretados. La función control que muestra el diagrama es como un salto indirecto a través de la memoria que apunta tanto al código máquina como al interpretado, aunque Sun no ha proporcionado muchos detalles sobre este extremo.

Diferencias y similitudes con C++

Java no soporta typedefs, defines o comandos de preprocesador. Al no existir un preprocesador, no está prevista la inclusión de ficheros de cabecera, tampoco tiene cabida el concepto de macro o constante. Sin embargo, sí se permite cierto uso de constantes enumeradas a través de la utilización de la palabra clave final. Java tampoco soporta enums, aunque soporte constantes enumeradas, como acabo de decir.

Java soporta clases, pero no soporta estructuras o uniones.

Todas las aplicaciones C++ necesitan una función de entrada llamada main() y puede haber multitud de otras funciones, tanto funciones miembros de una clase como funciones independientes. En Java no hay funciones independientes, absolutamente todas las funciones han de ser miembros de alguna clase (métodos). Funciones globales y datos globales en Java no están permitidos.

En C++ se pueden crear árboles de herencia de clases independientes unos de otros. En Java esto no es posible, en última instancia hay una clase Object, de la que hereda todo lo que el programador cree.

Todas las funciones o definiciones de métodos en Java deben estar contenidos dentro de la definición de la clase. Para un programador C++ puede parecerle que esto es semejante a las funciones inline, pero no es así. Java no permite, al menos directamente, que el programador pueda declarar funciones inline.

Tanto Java como C++ soportan funciones o métodos (estáticos) de clases, que pueden ser invocados sin necesidad de tener que instanciar ningún objeto de la clase.

En Java se introduce el concepto de interface, que no existe en C++. Una interface se utiliza para crear una clase base abstracta que contenga solamente las constantes y las declaraciones de los métodos de la clase. No se permite que haya variables miembro ni definiciones de métodos. Además, en Java también se pueden crear verdaderas clases abstractas.

Java no soporta herencia múltiple, aunque se pueden utilizar las posibilidades que ofrece el uso de interfaces para emplear las ventajas que ofrece la herencia múltiple, evitando los inconvenientes que se derivan de su uso. La herencia simple es similar en Java y en C++, aunque la forma en que se implementa es bastante diferente, especialmente en lo que respecta a la utilización de los constructores en la cadena de herencia.

Java no soporta la sentencia goto, aunque sea una palabra reservada. Sin embargo, soporta las sentencias break y continue con etiquetas, que no están soportadas por C++. Bajo ciertas circunstancias, estas sentencias etiquetadas se pueden utilizar como un goto encubierto.

Java no soporta la sobrecarga de operadores, aunque la utilice internamente, pero no está disponible para el programador. Tampoco soporta la conversión automática de tipos, excepto en las conversiones seguras.

Al contrario que C++, Java dispone de un tipo String y los objetos de este tipo no pueden modificarse. Las cadenas que se encuentren definidas entre comillas dobles son convertidas automáticamente a objetos String. Java también dispone del tipo StringBuffer, cuyos objetos sí se pueden modificar, y además se proporcionan una serie de métodos para permitir la manipulación de cadenas de este tipo.

Al contrario que C++, Java trata a los arrays como objetos reales. Disponen de un miembro, `length`, que indica la longitud del array. Se genera una excepción cuando se intenta sobrepasar el límite indicado por esta longitud. Todos los arrays son instanciados en memoria dinámica y se permite la asignación de un array a otro; sin embargo, cuando se realiza una asignación, simplemente tenemos dos referencias a un mismo array, no hay dos copias del array, por lo que si se altera un elemento de un array, también se alterará en el otro. A diferencia de C++, el tener dos "punteros" o referencias a un mismo objeto en memoria dinámica no representa necesariamente un problema (aunque sí puede provocar resultados imprevistos). En Java, la memoria dinámica es liberada automáticamente, pero esta liberación no se lleva a cabo hasta que todas las referencias a esa memoria son NULL o dejan de existir. Por tanto, a diferencia de C++, una zona de memoria dinámica nunca puede ser inválida mientras esté siendo referenciada por alguna variable.

Java no soporta punteros, al menos en el sentido que atribuye C++, es decir, no permite modificar el contenido de una zona de memoria apuntada por un puntero, o realizar operaciones aritméticas con punteros. La mayor necesidad de uso de punteros deriva de la utilización de cadenas y arrays, y esto se evita al ser objetos de primera clase en Java. Por ejemplo, la declaración imprescindible en C++, `char *puntero`, para referenciar al primer elemento de una cadena no se necesita en Java, al ser la cadena un objeto `String`.

La definición de clase es semejante en Java y C++, aunque en Java no es necesario el punto y coma (;) final. El operador de ámbito (::) necesario en C++ no se usa en Java, sino que se utiliza el punto (.) para construir referencias. Y, como no hay soporte de punteros, el operador flecha (->) tampoco está soportado en Java. En C++, las funciones y datos miembros se invocan utilizando el nombre de la clase y el nombre del miembro estático conectados por el operador de ámbito. En Java, se utiliza el punto (.) para conseguir el mismo propósito.

Al igual que C++, Java dispone de tipos primitivos como int, float, etc. Pero, al contrario de C++, los tamaños de estos tipos son iguales independientemente de la plataforma en que se estén utilizando. No hay tipos sin signo en Java, y la comprobación de tipos es mucho más restrictiva en Java que en C++. Java dispone de un tipo boolean verdadero.

Las expresiones condicionales en Java se evalúan a booleano en vez de a entero como en el caso de C++. Es decir, en Java no se permiten sentencias del tipo `if(x+y)`, porque la expresión que va dentro del paréntesis no se evalúa a booleano.

El tipo `char` en C++ es un tipo de 8 bits que mapea el conjunto completo de caracteres ASCII. En Java, el tipo `char` es de 16 bits y utiliza el set de caracteres Unicode (los caracteres del 0 al 127 del set Unicode, coinciden con el set ASCII).

Al contrario que en C++, el operador desplazamiento (`>>`) es un operador con signo, insertando el bit de signo en la posición vacía. Por ello, Java incorpora el operador `>>>`, que inserta ceros en las posiciones que van quedando vacías tras el desplazamiento.

C++ permite la instanciación de variables u objetos de cualquier tipo en tiempo de compilación sobre memoria estática o, en tiempo de ejecución, sobre memoria dinámica. Sin embargo, Java requiere que todas las variables de tipos primitivos sean instanciadas en tiempo de compilación, y todos los objetos sean instanciados en memoria dinámica en tiempo de ejecución. Java proporciona clases de envoltura para todos los tipos primitivos, excepto para `byte` y `short`, que permiten que estos tipos primitivos puedan ser instanciados en memoria dinámica como objetos en tiempo de ejecución, si fuese necesario.

C++ requiere que las clases y funciones estén declaradas antes de utilizarlas por primera vez. Esto no es necesario en Java. C++ también requiere que los miembros estáticos de una clase se redeclaren fuera de la clase. Esto tampoco es necesario en Java.

En C++, si no se indican valores de inicialización para las variables de tipos primitivos, pueden contener basura. Aunque las variables locales de tipos primitivos se pueden inicializar en la declaración, los datos miembros de tipo primitivo de una clase no se pueden inicializar en la definición de la clase. En Java, se pueden inicializar estos datos miembros de tipo primitivo en la declaración de la clase. También se pueden inicializar en el constructor. Si la inicialización no se realiza explícitamente, o falla por lo que sea, los datos son inicializados a cero (o su equivalente) automáticamente.

Al igual que ocurre en C++, Java también soporta constructores que pueden ser sobrecargados. Y, del mismo modo que sucede en C++, si no se proporciona un constructor explícitamente, el sistema proporciona un constructor por defecto.

En Java todos los objetos se pasan por referencia, eliminando la necesidad del constructor copia utilizado en C++. No hay destructores en Java. La memoria que no se utiliza es devuelta al Sistema a través del reciclador de memoria, que se ejecuta en un thread diferente al del programa principal. Esta es una de las diferencias extremadamente importantes entre C++ y Java.

Como C++, Java también soporta la sobrecarga de funciones. Sin embargo, el uso de argumentos por defecto no está soportado en Java. Al contrario que C++, Java no soporta templates, por lo que no existen funciones o clases genéricas.

El multithreading, o multihilo, es algo característico de Java, que se proporciona por defecto.

Aunque Java utiliza las mismas palabras clave que C++ para indicar el control de acceso: `private`, `public` y `protected`, la interpretación es significativamente diferente entre C++ y Java.

La implementación del manejo de excepciones en Java es más completo y bastante diferente al empleado en C++.

Al contrario que C++, Java no soporta la sobrecarga de operadores. No obstante, los operadores `+` y `+=` son sobrecargados automáticamente para concatenar cadenas, o para convertir otros tipos a cadenas.

Como en C++, las aplicaciones Java pueden hacer llamadas a funciones escritas en otros lenguajes, llamadas métodos nativos. No obstante, los applets no pueden hacer llamadas a métodos nativos.

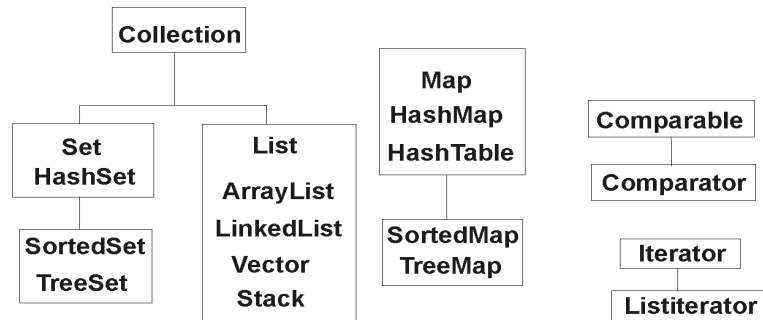
A diferencia de C++, Java dispone de un sistema interno de generación de documentación. Si se utilizan comentarios escritos de determinada forma, se puede utilizar la herramienta javadoc para generar la documentación de la aplicación Java, e incluso se pueden programar doclets para generar tipos específicos de documentación.

Clases útiles

Lo mejor para conocer las clases más útiles es ponerse a programar. Para ayudar al principio de este largo camino, se muestra la siguiente tabla:

Arrays (vectores)	<code>double[] x = new double[100];</code>
Arrays bidimensionales (matrices)	<code>int[][] y = new int[3][4];</code> <code>int [][] b = {{1,2,3},{4,5,6}};</code>
String (cadenas)	para cadenas invariables. <code>String s = new String(...);</code> // ver documentación.
Wrappers	Son las clases diseñadas para complemento de los tipos primitivos y NO son objetos. Los wrappers son <i>Byte</i> , <i>Short</i> , <i>Integer</i> , <i>Long</i> , <i>Float</i> y <i>Double</i>
java.lang.Math	Proporciona métodos <i>static</i> para realizar operaciones matemáticas. <code>Math.random()</code>
Colecciones	Una colección no es más que un conjunto de objetos que se agrupan, cualquier colección se identifica por el interfaz, <i>Collection</i> Son colecciones las clases <code>java.util.Vector</code> ,

java.util.HashSet, ... y también son colecciones los interfaces java.util.List, java.util.Map, ...



- *Collection* define métodos para tratar una colección genérica de elementos.
- *Set* Colección de elementos que no admite repetición.
- *SortedSet* es un *Set* ordenado según un criterio establecido.
- *List* admite elementos repetidos y mantiene el orden de inserción.
- *Map* Conjunto de pares, clave/valor, sin repetición de claves.
- *SortedMap* es un *Map* ordenado según un criterio establecido.
- *Iterator* Interfaz de soporte utilizado para recorrer una colección y para borrar elementos.
- *ListIterator* interfaz de soporte que permite recorrer *List* en ambos sentidos.
- *Comparable* interfaz de soporte que declara el método *compareTo()* que permite ordenar las diferentes colecciones según un orden natural.
- *Comparator* interfaz de soporte que declara el método *compare()* y se utiliza en lugar de *Comparable* cuando se desea ordenar objetos no estándar o sustituir a dicha interface.

java.awt

AWT es *Abstract Windows Toolkit* y es la parte de Java que se ocupa de construir interfaces gráficas. Para construir una interfaz, se necesitan al menos tres elementos, un contenedor, unos componentes y un modelo de eventos.

El contenedor es la ventana o parte de la ventana donde se sitúan los componentes.

Los componentes son menús, botones, barras de desplazamiento, cajas y áreas de texto, etc, etc.

El modelo de eventos es la herramienta que nos permite controlar las acciones del usuario sobre los componentes. Cada vez que el usuario realiza una acción se produce un *evento* y AWT crea un objeto de la clase de evento correspondiente derivada de

	<p><i>AWTEvent</i>. El evento será posteriormente gestionado por un objeto que debe conocer el componente que ha recibido el evento. Así se diferencian dos tipos básicos de objetos, los que reciben los eventos, o <i>event source</i> y los que manejan los eventos <i>event listener</i>.</p> <p>Los objetos <i>event source</i> deben "registrar" los objetos que habrán de gestionar sus eventos. Los objetos <i>event listeners</i> deben implementar los métodos adecuados para manejar un evento determinado y la forma más sencilla de controlar esto es implementar el interfaz <i>Listener</i> de la forma adecuada.</p> <pre>eventSourceObject.addEventListener(eventListenerObject);</pre>
Threads	<p>Los threads o hilos de ejecución permiten organizar los recursos del ordenador de forma que pueda haber varios programas actuando en paralelo. Un hilo de ejecución realizará las acciones indicadas en el método <i>run()</i>.</p> <pre>public class SimpleThread extends Thread{ public SimpleThread(String name){ super(name); } public void run(){ for(int i=0;i<10;i++) System.out.println("Este es el thread " + getname()); } }</pre>
Exceptions	<p>Una excepción es la herramienta de la que dispone el programador para controlar los posibles errores. Una excepción indica que una situación inusual se ha dado y el control del programa ha sido automáticamente transferido, <i>thrown</i>, a una sección de código especial donde se va a manejar.</p> <p>Las excepciones están organizadas en una jerarquía que parte de la clase <i>java.lang.Exception</i> y los errores tipificados están organizados a partir de la clase <i>java.lang.Error</i>. La diferencia entre excepciones y errores son que las primeras son recuperables mientras que los errores no lo son.</p> <p>El manejo de excepciones se realiza mediante <i>try/catch/finally</i>.</p> <p>Las excepciones se lanzan mediante <i>throw</i>, <i>throw new Exception()</i>.</p>

Lenguaje Java

Variables y Constantes

Tipos Primitivos de Variables	boolean, char, byte, short, int, long, float y double
Creación	Se crean igual que en C. por ejemplo: int i;
Inicialización	Se inicializan con una asignación. int i = 1;
Constantes	existen constantes en decimal, en octal y en hexadecimal. int i = 123; //decimal int i = 0123; //octal int i = 0x123; //hexadecimal

Las variables de tipos básicos y las constantes son utilizadas como variables y constantes tradicionales en C. Cuando se utiliza una variable de un tipo básico como parámetro en un método, esta variable es por valor. Las Variables que no son de tipos básicos, son objetos. Todos los objetos en Java son referencias. Una referencia es lo mismo que un puntero tradicional en C o en C++. Las instancias de Java y los punteros de C tienen una diferencia y es que una referencia de Java se pasa a un método por valor. Esto significa que dentro de un método un objeto puede ser modificado en su interior, pero el puntero que lo representa, es decir, la referencia no se podrá modificar para que apunte a otro objeto diferente aunque este sea del mismo tipo.

Strings

Una cadena en Java es un objeto del tipo String. Este tipo de objeto es diferente a los demás porque Java, a pesar de ser un objeto las trata como variables primitivas. Por ejemplo, para construirlas no es necesario hacer un new, sino que pueden ser creadas como una variable normal:

```
String cadena = "mi cadena";
```

Sentencias

La sintaxis y reglas de la construcción de sentencias son las mismas que en C o C++:

- Se agrupan por *llaves*, {,}
- El ámbito de una declaración está dentro de su bloque.

Las principales sentencias que proporciona Java son:

if-else	<p>Es la sentencia más básica para controlar el flujo de un programa.</p> <pre> if(Boolean-expression) statement if (Boolean-expression) Sentencia else Sentencia </pre>
return	Devuelve el resultado de un método.
while	<pre> while(Boolean-expression) sentencia </pre>
do-while	<pre> while(Boolean-expression) sentencia </pre>
for	<pre> for(inicialización; Boolean-expression; incremento) sentencia </pre> <p>La inicialización y el incremento pueden incluir más de una expresión unidas por el operador ",".</p> <pre> public class OperadorComma { public static void main(String[] args) { for(int i = 1, j = i + 10; i < 5;i++, j = i * 2) { System.out.println("i= " + i + " j= " + j); } } } </pre>
break y continue	<p>Break rompe el flujo normal de un bucle y pasa a la siguiente sentencia después del bucle.</p> <p>Continue rompe el flujo normal del ciclo del bucle donde se ejecute y pasa a ejecutar la comprobación del bucle de nuevo.</p>

switch	<pre>switch(integral-selector) { case integral-value1 : sentencia; break; case integral-value2 : sentencia; break; case integral-value3 : sentencia; break; case integral-value4 : sentencia; break; case integral-value5 : sentencia; break; // ... default: statement; }</pre>
--------	--

Expresiones

Una expresión puede ser una sentencia o parte de ella que al ser evaluada produce un valor. El resultado de una expresión puede ser un número, un tipo de referencia. El tipo de la expresión se conoce en tiempo de compilación y su resultado en tiempo de ejecución.

Los operadores que utiliza Java son los mismos que en C junto con el operador "instanceof" que nos sirve para descubrir de que tipo es un objeto.

```
if( variable instanceof String){...}
```

En este ejemplo se comprueba si *variable* es del tipo *String*.

Crea un método que cree e inicialice un array unidimensional del tipo Double. El tamaño del array será determinado por un parámetro de este método y los valores que se utilizan para la inicialización serán dentro de un rango determinado por dos parámetros del método. Crea otro método que muestre por pantalla el array ya inicializado. Comprueba que estos dos métodos funcionan dentro de un programa adicional.

Modifica el ejercicio anterior para utilizar un array bidimensional.

Escribe una función que realice todas las posibles comparaciones lógicas entre dos cadenas, incluyendo, equals().

Escribe un programa que genere 25 números aleatorios y comprueba si están por encima o por debajo de un valor intermedio. Almacena los mayores en un array y los menores en otro array. Muestra los resultados finales por pantalla.

EJERCICIOS RESUELTOS

Sentencias Básicas (Condicionales, Aritmeticas y Contadores)

En los siguientes ejercicios, se utilizan las sentencias condicionales if-else, además de sentencias aritméticas como % (modulo), comparativas (<,>,<=,>=,etc) y tambien se hace uso de contadores y sentinelas.

1. Dado un número entero y positivo que se introduce por teclado, determinar si es par o impar.

EXPLICACION DEL PROGRAMA: Este programa lee un numero introducido por teclado y verifica si el numero es par o impar, verificando si el modulo del numero entre 2 es 0 o no.

```
//programa realizado por Freddy Valda Sanchez
import java.io.*;
public class a1
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Introduzca un numero entero y positivo");
    int a=Integer.parseInt(in.readLine());
    if(a%2==0)
    System.out.println("El numero es par");
    else
    System.out.println("El numero es impar");
    }
}
```

2. Dado un número entero que se introduce por teclado, determinar si es positivo, negativo o nulo.

EXPLICACION DEL PROGRAMA: El usuario ingresa un numero entero y el programa determina si es positivo, negativo o cero(nulo). Utiliza simples comparaciones y sale el mensaje correspondiente.

```
//Programa realizado por Andrés Prudencio R.
import java.io.*;
public class a2
{

public static void main (String args[])throws IOException
{

BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
System.out.println("Ingrese un numero entero: ");
int d = Integer.parseInt(in.readLine());
if (d==0)
System.out.print(" el numero es nulo ");
else
{if (d<0)
System.out.print("El numero es negativo");
else
System.out.print("El numero es positivo");}
}}
```

3. Dado un número entero que se introduce por teclado, determinar si se encuentra en el intervalo cerrado 51 - 100.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite Verificar que un número ingresado por teclado se encuentre en el intervalo cerrado [51,100]

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_3
{
public static void main (String args[])throws IOException
{
BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int n;
System.out.println("Ingrese el numero a verificar: ");
n = Integer.parseInt(in.readLine());
if(n>=51&n<=100)
System.out.println("El numero se encuentra en el intervalo cerrado [51-100]");
else
System.out.println("El numero no se encuentra en el intervalo cerrado [51-100]");
}}
```


4. Dado un número entero que se introduce por teclado, determinar si es negativo o superior a 100.

EXPLICACION DEL PROGRAMA: el programa en principio define la clase aa4 que contiene la función principal main. Primero se lee un numero, que es introducido por teclado y se hacen comparaciones para verificar si es negativo, mayor a 100 o en el intervalo 0-100. Finalmente se despliega por pantalla la condición de dicho numero.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class aa4
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Introduzca un numero");
    int a=Integer.parseInt(in.readLine());
    if(a<0)
    System.out.println("El numero es negativo");
    else
    if (a>100)
    System.out.println("El numero es mayor a 100");
    else
    System.out.println("El numero esta en el intervalo de 0 a 100");
    }}
```

5. Dado un número que se introduce por teclado, si es positivo verificar si se encuentra en el intervalo abierto 60 – 90, de lo contrario emitir un mensaje de error.

EXPLICACION DEL PROGRAMA: Este programa verifica si un numero ingresado por el teclado es positivo y si se encuentra en un intervalo predeterminado utilizando sentencias if.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a5
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Introduzca un numero");
```

```
int a=Integer.parseInt(in.readLine());
if(a<61)
System.out.println("ERROR");
else
if (a>89)

System.out.println("ERROR");
else
System.out.println("El numero esta en el intervalo abierto 60-90");
}
}
```

6. Una fuente de datos registra varias edades, la edad 0 indica el final del ingreso de datos, realice un programa para determinar el promedio de las edades ingresadas y además el porcentaje de personas mayores a los 50 años.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite registrar varias edades con la información que una vez ingresada la edad 0 esta nos indica el final del ingreso de edades, calculando le promedio de las edades y además el porcentaje de edades mayores a los 50 años.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class B_1
{

    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        double m = 1, aux = 0, n = 0, o = 0;
        System.out.println ("La edad 0 indica el final de ingreso de edades");
        while (m != 0)
        {
            System.out.print ("Ingrese una edad ");
            m = Integer.parseInt (in.readLine ());
            n++;
            aux = aux + m;
            if (m > 50)
            {
                o++;
            }
        }
        double s = (o * 100) / (n - 1);
        double aux2 = aux / (n - 1);
        System.out.println ("El promedio es " + aux2 );
    }
}
```

```
System.out.println ("El porcentaje de personas mayores a los 50 a*os es de: " + s );}}
```

7. Obtener el total en bonos que paga la empresa a sus empleados, además desea conocer cuantos empleados tienen más de 20 años de antigüedad y el porcentaje que reciben estos, respecto al total en bonos que paga la empresa. (Utilizar centinela).

EXPLICACION DEL PROGRAMA: El programa pide leer el minimo de años que un trabajador debe trabajar como minimo para recibir un bono y la cantidad de años que trabajaron cada trabajador (hasta 100 trabajadores) para después mediante operaciones aritméticas, mostrar en pantalla lo que se requiere en el problema.

```
//Realizado por Andres Prudencio R
import java.io.*;

public class b2
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]e=new int[100];
    int cb,cv;
    cb=0;cv=0;
    System.out.println("Indicar el numero de anios que un empleado debera haber
trabajado como minimo para recibir el bono ");
    int a=Integer.parseInt(in.readLine());
    System.out.println("Ingresar el numero de anios trabajados para cada empleado: (0
termina la entrada de datos)");
    int i=1;
    e[0]=1;
    while (e[i-1]!=0)
    { System.out.println("Empleado "+(i)+" : ");

        e[i]=Integer.parseInt(in.readLine());
        if (e[i]>a)
        {cb++; }
        if (e[i]>20)
        {cv++; }
        i++;}
    if (cb==0)
        System.out.println("Ningun empleado recibira bono");
    else
    {System.out.println("La empresa pagara "+cb+" bonos");
```

```
System.out.println( ((cv*100)/cb)+"% de todos los empleados que recibieron el bono han  
trabajado mas de veinte años");}  
}}
```

8. Leer una cierta cantidad de estaturas (detenga el proceso mediante un centinela) y determine el promedio de estaturas por debajo de 1,60 mts. y el promedio de estaturas en general.

EXPLICACION DEL PROGRAMA: Este programa lee estaturas hasta encontrar un 0 (el sentinel), luego saca un promedio de las estaturas bajo los 1.60m y un promedio de todas las estaturas.

```
//Realizado por Ignacio Salgado Faller  
import java.io.*;  
public class b3  
{  
    public static void main(String arg[])throws IOException  
    {BufferedReader in = new BufferedReader(new InputStreamReader(System.in));  
    double []x=new double[100];  
    double a=1,pb=0,pg=0,nb=0;  
    int i =0;  
    System.out.println("Introduzca las estaturas (0 para terminar):");  
    a=(Float.valueOf (in.readLine ())).floatValue ();  
    if (a>=0)  
    {x[i]=a;  
    i++;}  
    while (a!=0)  
    {a=(Float.valueOf (in.readLine ())).floatValue ();  
    x[i] =a;  
    pg=pg+x[i];  
    if (x[i]<1.6)  
    {pb=pb+x[i];  
    nb++;}  
    i++;}  
    i=i-2;  
    System.out.println("El promedio de estaturas menores a 1.60m es: "+(pb/nb));  
    System.out.println("El promedio general de estaturas es: "+(pg/i));  
    }  
}
```

9. Contar una lista de n números que ingresan por teclado, rechazando el número 0, luego obtenga el total de números positivos que ingresaron, total de negativos y el promedio de cada uno.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite ingresar n números por teclado rechazando el número 0 y nos calculara el total de números positivos ingresados por teclado y su promedio a la vez determinara lo mismo con los números negativos

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class B_9
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        double n, m, p = 0, t = 0, s = 0, s1 = 0, prop, prone;
        System.out.println ("Ingrese la cantidad de numeros a evaluar ");
        n = (Double.valueOf (in.readLine ())).doubleValue ();
        for (int i = 0 ; i < n ; i++)
        {
            System.out.println ("Ingrese un numero");
            m = (Double.valueOf (in.readLine ())).doubleValue ();
            if (m == 0)
                System.out.println ("vuelva a ingresar un numero:");
            else
            {
                if (m > 0)
                {
                    p++;
                    s = s + m;
                } if (m < 0)
                {
                    t++;
                    s1 = s1 + m;
                }
            }
            prop = s / p;
            prone = s1 / t;
            System.out.println ("El total de numeros positivos es:" + p);
            System.out.println ("El total de numeros negativos es:" + t);
            System.out.println ("El promedio de numeros positivos es: "+prop);
            System.out.println ("El promedio de numeros negativos es:" + prone);
        }
    }
}
```

PROBLEMAS PROPUESTOS

1. Escribir un programa que pida el tipo de cambio para que dada una cantidad en Bolivianos y la convierta en Dolares y viceversa.

2. Escribir un programa que pida una hora en segundos y la saque por pantalla en el formato "hh:mm:ss", es decir horas, minutos y segundos
3. Intercambiar el contenido de dos variables.
4. Determinar si un año es bisiesto (verificar las condiciones)
5. Leer un par ordenado y determinar en qué cuadrante se encuentra.
6. Convertir grados Centígrados en grados Fahrenheit
7. Leer tres números diferentes (elaborar la validación de datos) y determinar el valor intermedio(el número que no es mayor ni menor).
8. Realizar un programa con el cual se lea la hora, los minutos y segundos para luego mostrar su equivalente en segundos.
9. Dado un número x determinar si es múltiplo de otro número y.
10. Dados tres números, determinar el mayor y menor.

Ciclos y Series

En los siguientes ejercicios, se utilizan ciclos, mediante las sentencias for, while, do- while, también se desarrollan series de números.

1. Realizar un programa que lea un numero, luego desplegar esa cantidad de *.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite desplegar (*) n veces utilizando un pequeño ciclo for.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_67
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int n;
        System.out.println ("Ingrese un numero");
        n = Integer.parseInt (in.readLine ());
        System.out.println ("La cantidad de * a desplegar son:"+n);
        for (int i = 0 ; i < n ; i++)
        {System.out.print("-*-");}}}
```

2. Dado un valor n y otro valor y, desplegar en pantalla números correlativos desde 1 hasta n, reemplazando por un * cada vez que corresponda desplegar un número múltiplo de y.

EXPLICACION DEL PROBLEMA: Se pide un numero el cual será la cantidad de términos para la serie correlativa. Posteriormente se pide otro numero cuyos múltiplos no serán mostrados, pero se mostrara un asterisco en vez de esos números. Simplemente se usa un ciclo for para analizar cada número.

```
//Realizado por Andres Prudencio R.
import java.io.*;
public class b6
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
```

```

in = new BufferedReader (new InputStreamReader (System.in));
System.out.println("Ingresar el ultimo numero de la serie ");
int n=Integer.parseInt(in.readLine());
System.out.println("Ingresar el numero cuyos multiplos no seran mostrados");
int y=Integer.parseInt(in.readLine());
for (int i=1;n>=i;i++)
{
    if (i%y==0)
    System.out.print("*"+" ");
    else
    System.out.print(i+" ");
}
}
}

```

3. Programa que lea un número n luego desplegar la tabla de multiplicar de ese número. Realizar el programa: a) utilizando for b) Utilizando while c) utilizando do while.

EXPLICACION DEL PROGRAMA: Despliega la tabla de multiplicar tres veces de un numero n usando ciclos for, while, do while usando la misma condición: $10 \geq j$ donde j es el multiplicador.

```

//Realizado por Andres Prudencio R
import java.io.*;

public class b10
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    System.out.println("Ingresar el numero cuya tabla de multiplicar sera mostrada");
    int n=Integer.parseInt(in.readLine());
    int j=1;
    for(j=1;10>=j;j++)
        {System.out.println(n+"*"+j+"="+n*j);}
    System.out.println("-----");
    j=1;
    while(10>=j)
    {
        System.out.println(n+"*"+j+"="+n*j);
        j++;
    }
    System.out.println("-----");
    j=1;
    do
    {
        System.out.println(n+"*"+j+"="+n*j);
        j++;
    }
    while(j<=10);
}
}

```



```

    }
    while(10>=j);

}}

```

4. Desplegar los números de x hasta z, donde $x \leq z$, a) utilizando for`s b) Utilizando while c) utilizando do while.

EXPLICACION DEL PROGRAMA: Este programa despliega una serie de números desde x hasta z, sigue pidiendo los valores hasta que se cumpla que $x \leq z$. Se utilizan todos los tipos pedidos en el enunciado.

```

//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class b11
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    int x=1,z=0,opc=5;
    int num=0;
    while (x>z)
    {System.out.println("Ingresar el valor de x y z (x debe ser<=z):");
    x=Integer.parseInt(in.readLine());
    z=Integer.parseInt(in.readLine());
    }
    while (opc>3||opc<1)
    {System.out.println("1. Realizar desplegamiento mediante for");
    System.out.println("2. Realizar desplegamiento mediante while");
    System.out.println("3.Realizar el desplegamiento mediante do-while");
    System.out.println("Ingresar su opcion:");
    opc=Integer.parseInt(in.readLine());
    }
    switch (opc)
    {case 1:System.out.println("Los numeros son: ");
    for (int i=x;i<=z;i++)
    {System.out.print(i+" ");}
    break;
    case 2: System.out.println("Los numeros son: ");int i=x;
    while (i<=z)
    {System.out.print(i+" ");
    i++;}
    break;
    case 3:System.out.println("Los numeros son: ");int j=x;
    do
    {System.out.print(j+" ");
    j++;}

```

```

while (j<=z);
break;
}
}
}

```

5. Desarrollar un programa que, utilizando una función muestre en pantalla N filas de números naturales impares, de los siguientes números y en la forma siguiente:

```

1
1 3
1 3 5
.....

```

N (número de filas) se debe indicar por teclado.

EXPLICACION DEL PROGRAMA: El usuario ingresa el numero de filas a mostrar, aquel numero es parámetro de la función serie la cual es llamada desde el método main de la clase a6. La función serie es una función void que imprime la serie. Nota: la función serie debe estar afuera del método main, dentro de la clase a6

```

//Programa realizado por Andrés Prudencio R.
import java.io.*;
public class a6
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in=new BufferedReader(new InputStreamReader(System.in));
        System.out.print("Ingrese el numero de filas ");
        int d=Integer.parseInt(in.readLine());
        serie(d);
        public static void serie(int d)
        {int a=1,b=2;
        for (int f=1;d>=f;f++)
        {System.out.println("");
        for (int c=1;f*2>=c;c++)
        {
            if ((c%2)!=0)
                System.out.print(c+" ");
        }
        }}}

```

6. Mediante el uso de una función, escribir un programa para sumar los primeros N números naturales. El resultado desplegar en la función principal.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite sumar n números ingresados por teclado y también nos muestra la suma total

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_7
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int m,n,c=0;

        System.out.println("La cantidad de numeros a sumar : ");
        n = Integer.parseInt(in.readLine());
        for (int i=0;i<n;i++)
        {
            System.out.println("Ingrese un numero ");
            m = Integer.parseInt(in.readLine());
            c=c+m;
        }
        System.out.println("La suma de los números ingresados es:"+c);
    }
}
```

7. Mediante una función desplegar en pantalla N números naturales, en la siguiente forma:

```
1
23
456
.....
```

EXPLICACION DEL PROGRAMA: Este programa genera la secuencia mostrada arriba utilizando for's.

```
//Programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a8
{public static void main (String args[])throws IOException
```

```
{BufferedReader in;
in=new BufferedReader (new InputStreamReader (System.in));
int c=1;
System.out.println("Ingrese el numero de filas: ");
int num = Integer.parseInt(in.readLine());
System.out.println("La cadena es: ");
for (int i=1;i<=num;i++)
{System.out.println("");
for (int j=1;j<=i;j++)
{System.out.print(c+" ");
c=c+1;}}
}
}
```

8. Dada la siguiente sucesión de números: 2, 4, 8, 6, 36, 72, 70, 4900, 9800,... mediante el uso de funciones, mostrar en pantalla los términos de esta serie y calcular la suma de N (N se indica por teclado) elementos, es decir, SUMA=2+4+8+6+.....

EXPLICACION DEL PROGRAMA: El programa calcula la suma de la serie anterior para n terminos. Para ello, genera cada uno de los valores de dicha serie hasta el limite ingresado por el usuario. La logica que sigue la serie es de elevar al cuadrado, multiplicar por dos y restar dos.

```
//Programa realizado por Freddy Valda Sanchez
import java.io.*;
public class a9
{
public static void main(String args[])throws IOException
{BufferedReader in;
int s=0,a=2;
in=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Introduzca el limite N");
int n=Integer.parseInt(in.readLine());

for(int i=0;i<n;)
{
System.out.println(a);
s=s+a;
i++;
a=op1(a);
if(i==n)
break;
else
{System.out.println(a);
s=s+a;
```

```

a=op2(a);
i++;
}
if(i==n)
break;
else
{System.out.println(a);
s=s+a;
a=op3(a);
i++;
}
}
System.out.println("La suma de la serie es: "+s);
}
public static int op1 (int b)
{return b*b;}
public static int op2 (int b)
{return b*2;}
public static int op3 (int b)
{return b-2;}
}

```

9. Desarrollar un programa que, utilizando funciones, se despliegue en pantalla N términos de la serie: 1, 3, 1, 1, 4, 1, 1, 1, 5, 1, 1, 1, 1, 6, . . .

EXPLICACION DEL PROGRAMA: El usuario ingresa el numero de términos a mostrar, aquel numero es parámetro de la función serie la cual es llamada desde el método main de la clase a10. La función serie es una función void que imprime la serie.

```

//Programa realizado por Andrés Prudencio R.
import java.io.*;
public class a10
{
    public static void main (String args[])throws IOException
    {   BufferedReader in;
        in=new BufferedReader(new InputStreamReader(System.in));

        System.out.println("Ingrese el numero de terminos ");
        int n=Integer.parseInt(in.readLine());
        serie(n);
    }
    public static void serie(int n)
    {int a=3,k=0,ct=1;
    if(n<=0)
        System.out.print("el numero de terminos debe ser mayor a cero");
    else

```

```

{if (n==2)
System.out.print("1,3,");
else
System.out.print("1,");

}
while(ct<=n-2)
{ System.out.print(a+",");
ct++;
k=a-1;
for (int j=1;k>=j;j++)
{System.out.print("1,");
ct++;
if (ct==n)
{j=k*2;}
}
a++;
}
System.out.print("...");
}}

```

10. Desarrollar un programa, utilizando funciones, para calcular la siguiente suma:

$$1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$$

Indicar el valor de x ($0 \leq x \leq 1$) por teclado y sumar términos mientras estos sean mayores a 10^{-8} .

EXPLICACION DEL PROGRAMA: el programa calcula la suma de la serie anterior, para ello se define una función con retorno que calcula el factorial de un numero, una variable sum y una variable x que es introducida por teclado. Finalmente, en un ciclo while se generan cada uno de los términos de la serie y se van sumando, después se despliega la suma de la serie.

```

//Programa realizado por Freddy Valda
import java.io.*;
public class a17

```

```

{
public static void main(String args[])throws IOException
{BufferedReader in;
double sum=0,n=1;
int k=1;
in = new BufferedReader (new InputStreamReader (System.in));
System.out.println("Ingrese el valor de x (0<=x<=1)");
double x = (Double.valueOf(in.readLine())).doubleValue();

while (n>0.00000001)
{sum=sum+n;
n=(Math.pow(x,k)/fact(k));
k=k+1;
}
System.out.println("La suma de la serie es :"+ sum);
}
public static double fact(double k)
{double s=1;
for(double i=k;i>0;i--)
{s=s*i;}
return s;}
}

```

11. Desarrollar un programa, utilizando funciones, para calcular la siguiente suma:

$$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \dots$$

Indicar el valor de x ($0 \leq x \leq 1$) por teclado y sumar términos mientras estos sean mayores a 10^{-8} .

EXPLICACION DEL PROGRAMA: Este programa genera la sumatoria de la secuencia mencionada arriba utilizando sentencias while. Utiliza varias variables para generar el exponente y el numero del cual se debe calcular el factorial, para el factorial se calcula mediante una funcion adicional.

```

//Programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a18
{

```

```

public static void main(String arg[])throws IOException
{BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
double x;
System.out.println("Introduzca el valor de x entre 0 y 1:");
x = (Float.valueOf (in.readLine ())).floatValue ();

double term=0,conta=1,aux=1,sig=1;
while (aux>0.00000001)
{
term=(sig*(Math.pow(conta,x)/facto(conta)))+term;
conta=conta+2;
sig=-sig;
aux=Math.pow(conta,x)/facto(conta);
}
System.out.println("La sumatoria es: "+aux);
}

public static double facto(double xx)
{if (xx==0)
{xx=1;
return xx;}
else
{double sum=0;
for (double i=1;i<xx;i++)
{sum=i*xx+sum;}
xx=sum;
return xx;}
}
}

```

12. Desarrollar un programa para que acepte por teclado valores para A, B y N y calcule la suma de la siguiente serie:

$$\frac{1}{A} + \frac{1}{A+B} + \frac{1}{A+2B} + \frac{1}{A+3B} + \dots + \frac{1}{A+NB}$$

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite sumar n términos de la serie ya antes propuesta y también desplegarla misma

```

// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_11
{
public static void main (String args[])throws IOException
{

```



```

BufferedReader in;

in = new BufferedReader (new InputStreamReader (System.in));
double A,B,N,C,D=0,E;
System.out.println("Ingrese el valor para A: ");
A = Integer.parseInt(in.readLine());
System.out.println("Ingrese el valor para B: ");
B = Integer.parseInt(in.readLine());
System.out.println("Ingrese el valor para N: ");
N = Integer.parseInt(in.readLine());
C=(1/A)+(1/(A+B));
for(int i=2;i<=N;i++)
{
D=(D+1/(A+(i*B)));
}
E=C+D;
System.out.println("La suma de la serie es:"+E);
}}

```

13. Realice un programa para mediante un menú de opciones calcular a) seno(x), b) coseno(x) y c) ln(x). Utilice las series de Taylor.

EXPLICACION DEL PROGRAMA: el programa utiliza funciones y un menú de opciones para calcular el seno, coseno y logaritmo neperiano de un número introducido por teclado. Existe otra función para calcular el factorial de un número. Se utilizan las series de Taylor para calcular dichas funciones matemáticas.

```

//Programa realizado por Freddy Valda
import java.io.*;
public class b4
{
public static void main(String args[])throws IOException
{double x;
int opc=0;
BufferedReader in;
in=new BufferedReader(new InputStreamReader(System.in));

do
{System.out.println("-----MENU DE OPCIONES -----");
System.out.println("1. Seno de un numero");
System.out.println("2. Coseno de un numero");
System.out.println("3. Logaritmo de un numero");
System.out.println("4. Salir");
System.out.println("Ingrese una opcion: ");
opc=Integer.parseInt(in.readLine());
switch(opc)
{case 1:sen();break;
case 2:cos();break;
case 3:log();break;
}
}
}
}

```

```
case 4:break;}
}
while(opc!=4);
}

static double fact(double e)
{
float f=1;
for (int i=1;e>=i;i++)
{f=f*i;}
return f;}

static void sen()throws IOException
{BufferedReader in;
double x;
in=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Ingrese el angulo en grados para calcular el seno: ");
x=Integer.parseInt(in.readLine());
x=(x*3.1416)/180;
double sum=0,den,num,nn,op;
for (float n=0;10>=n;n++)
{num=Math.pow(-1,n);
nn=(2*n)+1;
den=fact(nn);
op=(num*Math.pow(x,nn))/den;
sum=sum+op;}
System.out.println("El resultado es: "+sum);}

static void cos()throws IOException
{BufferedReader in;
double sum=0,den,num,nn,op;
double x;
in=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Ingrese el angulo en grados para calcular el coseno: ");
x=Integer.parseInt(in.readLine());
x=(x*3.1416)/180;

for (float n=0;10>=n;n++)
{num=Math.pow(-1,n);
nn=(2*n);
den=fact(nn);
op=(num*Math.pow(x,nn))/den;
sum=sum+op;}
System.out.println("El resultado es: "+sum);}

static void log()throws IOException
{BufferedReader in;
double x;
in=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Ingrese el numero para calcular el logaritmo neperiano: ");
x=Integer.parseInt(in.readLine());
double sum=0,suma,den,num,nn,op;
for (float n=0;100>=n;n++)
{num=Math.pow(-1,n);
nn=n+1;
```

```
den=nn;

op=(num*Math.pow(x,nn))/den;
sum=sum+op;}
suma=Math.log(x);
System.out.println("El resultado es: "+suma);}}
```

PROBLEMAS PROPUESTOS

1. Escribir un programa que pida una contraseña numerica y permita tres intentos. Si el usuario ingresa la contraseña correcta, que se muestre "Correcto!" y que ejecute un programa cualquiera, después del mensaje. En caso contrario el programa debe mostrar " Contraseña equivocada". Posteriormente terminar el programa de inmediato.
2. Escribir un programa que lea una lista de números y determine cuantos son positivos, y cuantos son negativos.
3. Obtener la serie:
 $1, -1, 2, -2, 3, -3, \dots$ para n números
4. Escribir un programa que permita obtener el valor aproximado de PI, mediante la serie:
 $4 - 4/3 + 4/5 - 4/7 + 4/9 - 4/11 + \dots$ para n términos.
5. Escribir un programa que permita adivinar un numero que se genere internamente al azar, el cual esta en el rango de 1 a 50. El usuario debe adivinar este numero en base a aproximaciones, para lo cual se dispone de 5 intentos. Ejemplo:

Salida:

Estoy pensando en un numero entre 1 y 50

Intento 1

? 25

El numero esta entre 25 y 50

Intento 2

?34

El numero esta entre 34 y 50

Intento 3

?45

El numero esta entre 34 y 45

Intento 4

?40

El numero esta entre 40 y 45

Intento 5

?42

Si es correcto, que muestre: "Felicidades, adivinaste el numero"

De lo contrario: "Se acabaron los intentos, el numero que pensé era 42".

6. Escribir un programa que dados dos números, uno real (base) y un entero positivo (exponente), saque por pantalla todas las potencias con base el numero dado y exponentes entre uno y el exponente introducido.

7. Generar las secuencia:

5 5 5 5 5

4 4 4 4

3 3 3

2 2

1

8. Generar la serie de Fibonacci

9. Generar: 1,2,6,24,120,... para n términos

10. Obtener el valor de la serie:

2,5,7,10,12,15,17,...,1800

Funciones y el uso de la clase Math

En los siguientes ejercicios, se utilizan funciones, con o sin retorno. Tambien se hace uso de la clase Math, para calcular potencias, raíces, logaritmos, numeros aleatorios , etc.

1. Preparar un programa, utilizando funciones, para que admita por teclado un entero positivo M y que obtenga todos los números primos que sean menores que o iguales a M.

EXPLICACION DEL PROGRAMA: El usuario ingresa un número. El programa mostrara todos los números primos menores o iguales al número ingresado. El programa realiza primeramente una validación de la entrada del usuario. El numero debe ser positivo, si se ingresa un numero negativo, se volverá a requerir la entrada del numero. Una vez validado, se llama a la función primos la cual mostrara la serie.

```
//Programa realizado por Andrés Prudencio R.
import java.io.*;
public class a19
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in=new BufferedReader(new InputStreamReader(System.in));

        System.out.print(" Los numeros primos deben ser menores o iguales a: ");
        int m=Integer.parseInt(in.readLine());

        if(m<0)
        {System.out.print("Debe introducir un numero POSITIVO ");
        while(m<0)
        {System.out.print("Intente de nuevo ");
        m=Integer.parseInt(in.readLine());

        }}

        primos(m);
    }

    public static void primos(int m)
    {int cd=0;
    for(int h=1;m>=h;h++)
        { for(int g=1;h>=g;g++)
```

```

    {if(h%g==0)
        cd++;
    }
    if(cd==2)
    {System.out.print(h+" ");
    cd=0;}
    else
    cd=0;
    }
}
}

```

2. Desarrollar un programa con funciones para listar en una tabla, con los encabezados correspondientes, los valores de X, Y y f(x,y), de acuerdo a la siguiente ecuación:

$$f(x,y) = \frac{x^2 + 3x + y^2}{xy - 5y - 3x + 15}, \text{ donde } x=2, 2.5, 3, 3.5; y=0, 0.1, 0.2, 0.3, 0.4.$$

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite calcular los valores de la función ya antes mencionada con los valores estáticos de las variables x e y los cuales nos muestra en una tabla.

```

// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_20
{public static void main (String args[])throws IOException
{
    BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    double c,d,e;
    System.out.println("Listado de una tabla de    X , Y    f(x,y)");
    for(double x=2;x<=3.5;x=x+0.5)
    {for(double y=0;y<=0.4;y=y+0.1)
    {
        c=(x*x)+(3*x)+(y*y);
        d=(x*y)-(5*y)-(3*x)+15;
        e=c/d;
        System.out.println(":f("+x+" , "+y+")es:"+e);
    }}}
}

```

3. Desarrollar un programa para listar en una tabla, con los encabezados correspondientes, los valores de X, Y y f(x,y), de acuerdo a la siguiente función:

$$f(x,y) = \frac{x^2 - y^2}{xy - 2y + 6x}; \text{ donde } x=0, 1, 2, 3; y=1, 2, 3, 4, 5$$

EXPLICACION DEL PROGRAMA: el programa lista en pantalla los valores de "x", " y" y de f(x,y). Para ello se definen 2 ciclos tipo for, uno que genere las x y otro que genere las y en el rango que indica el problema. Existe una función con retorno que calcula el valor de f(x,y). Finalmente, cada f(x,y) existente en el rango es desplegada por pantalla.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a21
{
    public static void main(String args[])throws IOException
    {
        double a,b;
        System.out.println("TABLA DE VALORES DE X y Y");
        System.out.println("-----");
        System.out.println("X          Y          f(X,Y)");
        for( a=0;a<=3;a++)
        {for(b=1;b<=5;b++)
        {System.out.println(a+"      "+b+"      "+func(a,b));
        }}
    }
    public static double func(double x,double y)
    {double aw,aq;
    aw=(x*x)-(y*y);
    aq=(x*y)-(2*y)+(6*x);
    return (aw/aq);}
}
```

4. Desarrollar un programa con un menú para calcular:

1. El volumen de un paralelepípedo rectangular de longitud A, altura B y ancho C.
2. El volumen de una esfera de radio r .
3. El volumen de un cilindro recto de radio r y altura h
4. El volumen de un cono circular recto de radio r y altura h .

EXPLICACION DEL PROGRAMA: Este programa calcula los volúmenes de varias figuras geometricas. Para esto utilizamos las formulas conocidad y ademas utilizamos un menú de opciones para que el usuario pueda calcular el volumen deseado.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a22
{
    public static void main(String arg[])throws IOException
    {BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    int opc=6;
    int a,b,c,v;
    double pi=3.1416,r,v1,h;
    while (opc<1||opc>5)
    {System.out.println("1. Volumen de un Paralelepipedo Rectangular");
    System.out.println("2. Volumen de una Esfera");
    System.out.println("3. Volumen de un Cilindro Recto");
    System.out.println("4. Volumen de un Cono Circular Recto");
    System.out.println("5. Salir");
    System.out.println("Seleccionar su opcion: ");
    opc = Integer.parseInt(in.readLine( ));
    }
    switch (opc)
    {case 1: System.out.println("Ingresar la base, altura y ancho: ");
    a = Integer.parseInt(in.readLine( ));
    b = Integer.parseInt(in.readLine( ));
    c = Integer.parseInt(in.readLine( ));
    v=a*b*c;System.out.println("El volumen es: "+v);break;
    case 2: System.out.println("Ingresar el radio: ");
    r= Integer.parseInt(in.readLine( ));v1=(r*r*r)*pi*(4/3);
    System.out.println("El volumen es: "+v1);break;
    case 3: System.out.println("Ingresar el radio y la altura: ");
    r= Integer.parseInt(in.readLine( ));
    h= Integer.parseInt(in.readLine( ));v1=pi*(r*r)*h;
    System.out.println("El volumen es: "+v1);break;
    case 4: System.out.println("Ingresar el radio y la altura: ");
    r= Integer.parseInt(in.readLine( ));
    h= Integer.parseInt(in.readLine( ));v1=(pi*(r*r)*h)/3;
    System.out.println("El volumen es: "+v1);break;
    }
    }}
}
```

5. Desarrollar un programa para desplegar en pantalla los siguientes números, mediante el uso de funciones:

1 2 3 4 5

5 4 3 2 1


```

1 2 3 4
4 3 2 1
1 2 3
3 2 1
1 2
2 1
1
1

```

EXPLICACION DEL PROGRAMA: El usuario ingresa un número el cual será el máximo para la serie a mostrar. En el ejemplo anterior sería 5. Al igual que en el ejercicio 19, se valida la entrada y se llama a la función anidada la cual mostrara la serie.

```

//Programa realizado por Andrés Prudencio R.
import java.io.*;
public class a23
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in=new BufferedReader(new InputStreamReader(System.in));

        System.out.print(" Numero maximo: ");
        int n=Integer.parseInt(in.readLine());

        if(n<0)
        {System.out.print("Debe introducir un numero POSITIVO ");
        while(n<0)
        {System.out.print("Intente de nuevo ");
        n=Integer.parseInt(in.readLine());}}

        anidado(n);
    }

    public static void anidado(int n)
    {for (int i=n;i>0;i--)
    {System.out.println();
    for (int y=1;y<=i;y++)
    {System.out.print(y+" ");}
    System.out.println();
    for (int j=i;j>0;j--)
    System.out.print(j+" ");
    }}}

```

6. Escribir un programa que presente un menú para convertir grados Centígrados, que se introduce por teclado, en grados Fahrenheit y viceversa, utilizando funciones. Las fórmulas de conversión son:

$$\text{Centígrados} = \frac{9.0}{5.0}(\text{grados Fahrenheit}) + 32.$$

$$\text{Fahrenheit} = \frac{5.0}{9.0}(\text{grados Centígrados} - 32).$$

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite convertir grados centígrados ingresados por teclado a grados Fahrenheit y viceversa.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_24
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        double n,f,c;
        int res=1;
        while(res!=0)
        {
            System.out.println("MENU DE OPCIONES ");
            System.out.println("1.Ingresar grados centigrados y convertirlo a fahrenheit ");
            System.out.println("2.Ingresar grados fahrenheit y convertirlo a centigrado ");
            System.out.println("3. SALIR");
            System.out.println("INGRESE LA OPCION");
            res=Integer.parseInt(in.readLine());
            switch (res)
            {
                case 1:
                    System.out.println("Ingrese la temperatura en grados centigrados");
                    n = (Double.valueOf (in.readLine ())).doubleValue ();
                    f=(1.8*n)+32;
                    System.out.println("La temperatura ingresada en grados fahrenheit es:"+f);
                    break;
                case 2:
                    System.out.println("Ingrese la temperatura en grados fahrenheit");
                    n = (Double.valueOf (in.readLine ())).doubleValue ();
                    c=(n-32)*(0.55);
                    System.out.println("La temperatura ingresada en grados centigrados es:"+c);
                case 3: System.exit(0);
                break;
            }
        }
    }
}
```

7. Escribir un programa que presente un menú para calcular funciones matemáticas, como la raíz cuadrada, logaritmo neperiano, logaritmo decimal y el valor de ex, para un número que se introduce por teclado, utilizando funciones.

EXPLICACION DEL PROGRAMA: el programa despliega un menú de opciones para calcular diversas funciones matemáticas. Existen funciones con retorno para calcular el cada una de ellas. Primero se debe ingresar el valor para posteriormente hallar la función deseada. Para dichas funciones se utiliza la clase Math.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a25
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    double x=0;
    int resp;
    do
    {
        System.out.println("MENU PARA CALCULAR FUNCIONES MATEMATICAS");
        System.out.println("-----");
        System.out.println("1. RAIZ CUADRADA");
        System.out.println("2. LOGARITMO NEPERIANO");
        System.out.println("3. LOGARITMO DECIMAL");
        System.out.println("4. VALOR DE e ELEVADO A LA X");
        System.out.println("5. INGRESAR EL VALOR");
        System.out.println("6. SALIR");
        System.out.println("INGRESE LA OPCION POR FAVOR");
        resp=Integer.parseInt(in.readLine());
        switch (resp){
            case 1:raiz(x);break;
            case 2:lognep(x);break;
            case 3:logdec(x);break;
            case 4:ealax(x);break;
            case 5:
                x = (Double.valueOf(in.readLine())).doubleValue();break;
            case 6:
                break;
        }
    }
    while(resp!=6);

    }
    public static void raiz(double x)
    {double r;
```

```
r=Math.sqrt(x);
System.out.println("RAIZ CUADRADA:"+r);}
public static void lognep(double x)
{double r;
r=Math.log(x);
System.out.println("LOGARITMO NEPERIANO:"+r);}
public static void logdec(double x)
{double r;
r=Math.log(x)/Math.log(10);
System.out.println("LOGARITMO DECIMAL:"+r);}
public static void ealax(double x)
{double r,e;
e=2.717281;
r=Math.pow(e,x);
System.out.println("VALOR DE e ELEVADO A LA X:"+r);}
}
```

8. Escribir un programa, para generar y desplegar 20 números aleatorios utilizando la función `RAND()`, en el rango de 20 a 80.

EXPLICACION DEL PROGRAMA: Este programa genera numeros aleatorios entre los limites propuestos utilizando la formula $(\text{Limite superior} - \text{Limite inferior}) + \text{Limite inferior}$ para la sentencia `random`. Luego despliega los numeros

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a26
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int num;
System.out.println("Los numeros son: ");
for (int u=1;u<=20;u++)
{num=(int)(Math.random()*60+20);
System.out.print(num+" ");}
}}
```

9. Escribir un programa, con funciones, para generar y desplegar N números aleatorios comprendidos entre 0 y 1.

EXPLICACION DEL PROGRAMA: El usuario ingresa la cantidad de números que serán mostrados, bajo esa cantidad, se generaran números al azar entre 0 y 1. En este programa se utiliza en la salida de números la función `df.format`, la cual reduce la cantidad de decimales a mostrar para los tipos de datos reales. En este caso dos. Esto se define en la sentencia `DecimalFormat df = new DecimalFormat("0.00");`. Se debe incluir la librería `java.text.DecimalFormat` antes de declarar la clase.

```
//Programa realizado por Andrés Prudencio R.
import java.io.*;
import java.text.DecimalFormat;
public class a27
{
    public static void main (String args[])throws IOException
    {   BufferedReader in;
        in=new BufferedReader(new InputStreamReader(System.in));

        System.out.print(" Cantidad de numeros: ");
        int n=Integer.parseInt(in.readLine());

        if(n<0)
        {System.out.print("Debe introducir un numero POSITIVO ");
        while(n<0)
        {System.out.print("Intente de nuevo ");
        n=Integer.parseInt(in.readLine());}}
        rr(n);
    }
    public static void rr(int n)
    {DecimalFormat df = new DecimalFormat("0.00");
    float f;
    for (int i=1;n>=i;i++)
    {if (i%40==0)
    System.out.println();
    f=(float)(Math.random()*0.9);
    System.out.print(df.format(f)+" "); }}}}
```

10. Escribir un programa, para generar y desplegar N números aleatorios comprendidos entre dos valores límites que se indican por teclado.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite desplegar n números comprendidos entre dos límites x e y que también son ingresados por teclado utilizando una forma aleatoria para desplegar los distintos números

```
// Programa realizado por Sergio W. Ossio Marin
import java.lang.Math.*;
import java.io.*;
public class A_28
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int n,x,y,i,k;

        System.out.println("Ingrese cantidad de números a desplegar: ");
        n = Integer.parseInt(in.readLine());
        System.out.println("Ingrese el limite inferior: ");
        x = Integer.parseInt(in.readLine());
        System.out.println("Ingrese el limite superior: ");
        y = Integer.parseInt(in.readLine());
        for (i=0;i<n;i++)
        {k=(int)(Math.random()*(y-10)+x);
        System.out.println("los numeros son "+k);
        }}}

```

11. Realiza un programa para desplegar **n** números aleatorios menores que **y**.
Donde **n** también es aleatorio.

EXPLICACION DEL PROGRAMA: Este programa despliega tantos números aleatorios genere el programa siempre y cuando sean menores que y, que también es aleatorio.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class b7
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    int y=(int) (Math.random()*50);
    int num=0;
    int n=(int)(Math.random()*50);
    System.out.println("Se desplegaran "+n+" numeros menores que "+y);
    for (int i=0;i<n;i++)
    {num=(int)(Math.random()*50);
    if (num>=y)
    {i--;}
    else
    {System.out.print(num+" ");}
    }
    }
}

```

12. Realizar un programa para calcular y desplegar, mediante funciones, de los primeros 15 números naturales, una tabla con valores correspondientes a las siguientes funciones matemáticas

TABLA DE VALORES

No.	Sen(x)	Cos(x)	Tan(x)	log(x)	log10(x)	Sqrt(x)
-----	--------	--------	--------	--------	----------	---------

EXPLICACION DEL PROGRAMA: el programa calcula las funciones matemáticas correspondientes a la tabla anterior, para ello se hace uso de un ciclo for del 1 al 15 y se calcula valores con la clase Math. Existen funciones al final de la clase a29 para realizar dichas operaciones. Para poder desplegar los resultados con 4 decimales se define un formato decimal y se lo utiliza al momento de desplegar el número.

```
//Programa realizado por Freddy Valda
import java.io.*;
import java.text.DecimalFormat;
public class a29
{
    public static void main(String args[])throws IOException
    {
        double x;
        DecimalFormat df = new DecimalFormat("0.0000");
        System.out.println("TABLA DE VALORES");
        System.out.println("-----");
        System.out.println("No          Sen(x)    Cos(X)    Tan(X)    log(x)    log10(x)
sqrt(x)");
        for(x=1;x<=15;x++)
        {System.out.println(df.format(x)+"    "+df.format(seno(x))+
"+df.format(coseno(x))+    "+df.format(tangente(x))+    "+df.format(lognep(x))+
"+df.format(logdec(x))+    "+df.format(raiz(x)));
        }
    }

    public static double seno(double x)
    {double r;
    r=Math.sin(x);
    return r;
    }

    public static double coseno(double x)
    {double r;
    r=Math.cos(x);
    return r;
    }
}
```

```

public static double tangente(double x)
{double r;
r=Math.tan(x);
return r;}

public static double raiz(double x)
{double r;
r=Math.sqrt(x);
return r;}

public static double lognep(double x)
{double r;
r=Math.log(x);
return r;}

public static double logdec(double x)
{double r;
r=Math.log(x)/Math.log(10);
return r;}
}

```

13. Realizar un programa para calcular y desplegar de 15 números reales con dos decimales, que se generan aleatoriamente, una tabla con valores correspondientes a las siguientes funciones:

TABLA DE VALORES

No.	Ceil(x)	Floor(x)	Pow(x, 2)	Pow(10,x)
-----	---------	----------	-----------	-----------

EXPLICACION DEL PROGRAMA: Este programa genera la tabla mostrada arriba, utilizando una libreria para que los numeros generados solo muestren 2 decimales en vez de todos los decimals que genera random. Se utilizaron varias funciones void para calcular cada dato requerido. Por ejemplo, si genera el numero 2.3, la funcion ceil mostrara su valor entero inmediato inferior (es decir, 2), la funcion floor mostrara su valor entero inmediato superior (es decir, 3). Pow(2,x)ara el numero dos a la x-esima potencia ($2^{2.3}$) y pow (10,x) elevara x a la decima potencia (2.3^{10})


```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
import java.math.*;
import java.text.DecimalFormat;
public class a30
{
    public static void main(String args[])throws IOException
    {
        double y;
        int x;
        DecimalFormat df = new DecimalFormat("0.00");
        System.out.println("TABLA DE VALORES");
        System.out.println("-----");
        System.out.println("No.    Ceil(x)    Floor(X)    Pow(x,2)    Pow(10,x)");
        System.out.println("-----");
        for(x=1;x<=15;x++)
        {y=(double)(Math.random()*100);
        System.out.println(x+" "+df.format(ceil(y))+ " "+df.format(floor(y))+ " "+df.format(pow(y))+ " "+df.format(pow2(y)));
        }
    }

    public static double ceil(double x)
    {long d;
    d=(long)x;
    return d;
    }

    public static double floor(double x)
    {long j;
    j=(long)x;
    if (x%j==0)
    {return x;}
    else
    {x=x+1;
    j=(long)x;
    return j;}
    }

    public static double pow(double x)
    {double bb;
    bb= Math.pow(x,2);
    return bb;
    }

    public static double pow2(double x)
    {double ss;
    ss= Math.pow(10,x);
    return ss;
    }
}
```

PROBLEMAS PROPUESTOS

1. Usando funciones y la sentencia switch, escribir un programa que pida los datos necesarios y calcule el área y el perímetro para cada figura indicada.
a) Un cuadrado b) Un rectángulo c) Un triángulo d) Un círculo

2. Elabore un programa que calcule el area de un triangulo mediante la formula:

$$Area = \sqrt{p(p-a)(p-b)(p-c)}$$

lados del triangulo. Para que el triangulo exista, debe cumplirse donde p es el semiperimetro, $p=(a+b+c)/2$, siendo a, b, c los tres que los lados sean todos positivos, y además que la suma de dos lados cualesquiera sea mayor que el otro lado.

3. Diseñar una función que calcule el promedio de varios números introducidos por el teclado. Hacer dos versiones, una para un número fijo de valores y otra para un número cualquiera.de valores.
4. Escribir una función que intercambie el valor de dos variables, es decir si $X=5$ e $Y=7$ tras aplicar la función, por ejemplo haciendo "intercambiar(X,Y)" se tiene que $X=7$ e $Y=5$.
5. Diseñar una función que calcule la potencia enésima de un número, es decir que calcule X^n para X real y n entero.
6. Escriba una función con retorno que determine si un numero es cubo perfecto (un cubo perfecto es aquel numero que la suma de sus digitos elevados al cubo reproduce el mismo numero. Ej: $1^3 + 5^3 + 3^3 = 153$, en caso de serlo retorne 1 o 0 si no lo es.

7. Escriba una función VOID que obtenga la suma de los n primeros pares primos.
8. Leer un número entero y determinar si el mismo es un numero capicúa emplear una función con retorno (1 si es capicúa, 0 si no lo es).
9. Escribir una función con retorno que encuentre la raíz cuadrada de un número empleando el método de Newton.
10. Escribir una función con retorno para determinar si 3 números leídos desde el teclado forman un triangulo (retornar 1 si así lo fueran y 0 de no ser así), luego mediante una segunda función con retorno determinar si forman un triangulo equilátero (1), isósceles(2) o escaleno (3). Los valores que aparecen entre paréntesis son los valores a retornar.

Arreglos

Los siguientes programas, fueron realizados utilizando arreglos o vectores. Mediante el uso de este tipo de variables se resolvieron estos problemas.

1. Escribir un programa que llene un vector con una lista de números del 1 al 20, luego despliegue este vector indicando a la derecha de cada uno si es divisible por 3 o no.

EXPLICACION DEL PROGRAMA: En este programa no existe la entrada de datos. Genera un listado de números en los que se determina si son o no múltiplos de 3. Para determinar esto, se analiza el resto de la división de cada numero del vector entre 3, si este es cero, el numero es múltiplo.

```
//Programa realizado por Andrés Prudencio R.
import java.io.*;
public class a31
{
    public static void main(String args[])
    {
        int[]vector=new int[20];
        int n=20;
        for(int i=0;i<n;i++)//llenado del vector
        {
            vector[i]=i+1;
        }

        for(int i=0;i<n;i++)
        {
            if (vector[i]%3==0)
                System.out.println(vector[i]+" es Divisible por 3");
            else
                System.out.println(vector[i]+" No es divisible por 3");
        }
    }
}
```

2. Escribir un programa que llene un vector con la siguiente secuencia numérica: 1, 5, 3, 7, 5, 9, 7, ..., 23. La secuencia debe detenerse al llegar al 23. Luego desplegar el vector en pantalla.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite generar la serie ya antes mencionada y poderla llenar en un vector y a su vez mostrarla.

```
// Programa realizado por Sergio W. Ossio Marin
class A_32
{public static void main (String args [])
{
    int [] vector = new int [100];
    int s = 1;
    vector [0] = 1;
    for (int i = 1 ; i <= 20 ; i++)
    {
        if (i % 2 != 0)
        {
            s = s + 4;
            vector [i] = s;
        }
        if (i % 2 == 0)
        {
            s = s - 2;
            vector [i] = s;
        }
    }
    for (int j = 0 ; j < 20 ; j++)
        System.out.print (" " + vector [j]);
}}
```

3. Hacer un programa que lea 10 valores enteros en un array desde el teclado y calcule y muestre: la suma, el valor medio, el mayor y el menor

EXPLICACION DEL PROGRAMA: El programa lee 10 valores desde teclado y calcula la suma de estos, el valor medio, el mayor y el menor, para esto se utiliza un ciclo para llenar una arreglo previamente declarado, posteriormente mediante comparaciones simples y luego de declarar 2 variables que serán el mayor y el menor y utilizando un ciclo for, se halla el mayor y menor valores.

```
//Programa realizado por Freddy Valda
//Arreglo
import java.io.*;
public class a33
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int[]vector=new int[15];
    int suma=0,prom,aux1,aux2;
    System.out.println("INGRESE LOS 10 VALORES DEL ARREGLO ");
    for(int i=1;i<=10;i++)
    {vector[i]=Integer.parseInt(in.readLine());
    suma=suma+vector[i];
    }
    prom=suma/10;
    aux1=vector[1];
    aux2=vector[1];
    for(int j=1;j<=10;j++)
    {if(vector[j]>aux1)
    aux1=vector[j];
    else
    ;
    if(vector[j]<aux2)
    aux2=vector[j];
    }

    System.out.println("LA SUMA DEL ARREGLO ES: "+suma);
    System.out.println("EL VALOR MEDIO DEL ARREGLO ES: "+prom);
    System.out.println("EL MAYOR VALOR ES: "+aux1);
    System.out.println("EL MENOR VALOR ES: "+aux2);
    }
}
```

4. Diseñar e implemente un programa para intercalar los elementos de dos vectores A y B, cuyos elementos están ordenados de menor a mayor y que dé como resultado otro vector C ordenado ascendentemente sin elementos repetidos. Desplegar en pantalla los tres vectores.

EXPLICACION DEL PROGRAMA: Este programa permite al usuario crear dos vectores, luego estos son ordenados ascendentemente mediante el metodo shell sort y los muestra. Luego el programa crea un tercer vector donde se guardan los elementos no repetidos de cada uno de los dos vectores, lo ordena y lo muestra.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
```

```

public class a34
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]a=new int[100];
int[]b=new int[100];
int[]c=new int[200];
int k,aux,cc=0,cia=0,cib=0,pg=0;
System.out.println("Introduzca la longitud del primer y segundo vector: ");
int n=Integer.parseInt(in.readLine());
int m=Integer.parseInt(in.readLine());
System.out.println("Introduzca los datos del primer vector(1-100): ");
for(int i=0;i<n;i++)
{a[i]=Integer.parseInt(in.readLine());}
System.out.println("Introduzca los datos del segundo vector(1-100): ");
for(int f=0;f<m;f++)
{b[f]=Integer.parseInt(in.readLine());}
k=n/2;
while (k>0)
{ for(int x=0;x<n;x++)
{if((k+x)<=(n-1))
{if (a[x]>a[x+k])
{aux=a[x+k];
a[x+k]=a[x];
a[x]=aux;
cc=cc+1;}}}

if (cc==0)
k=(int)(k/2);
cc=0;}
k=m/2;
while (k>0)
{ for(int x=0;x<m;x++)
{if((k+x)<=(m-1))
{if (b[x]>b[x+k])
{aux=b[x+k];
b[x+k]=b[x];
b[x]=aux;
cc=cc+1;}}}

if (cc==0)
k=(int)(k/2);
cc=0;}
pg=0;
for (int nb=0;nb<n+m;nb++)
{c[nb]=0;}
for (int i=0;i<n;i++)
{for (int nb=0;nb<n+m;nb++)
{if (a[i]==c[nb])
{cia++;}}
if (cia==0)
{c[pg]=a[i];
cia=0;

```

```

pg=pg+1;}
else
{cia=0;}}
for (int i=0;i<m;i++)
{for (int nb=0;nb<n+m;nb++)
{if (b[i]==c[nb])
{cib++;}}
if (cib==0)
{c[pg]=b[i];
cib=0;
pg=pg+1;}
else
{cib=0;}}
k=(m+n)/2;
while (k>0)
{ for(int x=0;x<m+n;x++)
{if((k+x)<=(n+m-2))
{if (c[x]>c[x+k])
{aux=c[x+k];
c[x+k]=c[x];
c[x]=aux;
cc=cc+1;}}}

if (cc==0)
k=(int)(k/2);
cc=0;}
System.out.println("Los vectores son: ");
for(int f=0;f<n;f++)
{System.out.print(a[f]+" ");}
System.out.println("");
for(int i=0;i<m;i++)
{System.out.print(b[i]+" ");}
System.out.println("");
for(int i=0;i<pg;i++)
{System.out.print(c[i]+" ");}
}
}

```

5. Escribir un programa para que calcule unión, intersección y diferencia de dos vectores de enteros A y B de tamaño máximo 10. El programa debe permitir la inicialización de los vectores desde el teclado.

EXPLICACION DEL PROGRAMA: Este programa no utiliza funciones, esta dividido en cuatro secciones. La primera es la entrada de datos de los vectores con ciclos for. La segunda es la obtención del vector unión. Un vector en el cual se iran grabando todos los datos de ambos vectores. La tercera parte es mostrar los vectores y su unión. La cuarta parte muestra la

intersección grabada también en un vector(aib) y por ultimo se muestran las diferencias a-b y b-a siendo a y b los vectores conteniendo los valores ingresados.

```
//Programa realizado por Andres Prudencio R.
import java.io.*;
public class a35
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]a=new int[10];
int[]b=new int[10];
int[]c=new int[20];//vector union
int[]aib=new int[10];
int k,aux,cc=0,cia=0,cib=0,pg=0,n,m;
//entrada de datos
System.out.print("Introduzca el tamaño de los vectores: ");
n=Integer.parseInt(in.readLine());
m=n;
System.out.println();
System.out.println("Introduzca los datos del primer vector(1-10): ");
for(int i=0;i<n;i++)
{a[i]=Integer.parseInt(in.readLine());}
System.out.println("Introduzca los datos del segundo vector(1-10): ");
for(int f=0;f<m;f++)
{b[f]=Integer.parseInt(in.readLine());}
//obtencion del vector union
k=n/2;
while (k>0)
{ for(int x=0;x<n;x++)
{if((k+x)<=(n-1))
{if (a[x]>a[x+k])
{aux=a[x+k];
a[x+k]=a[x];
a[x]=aux;
cc=cc+1;}}}
if (cc==0)
k=(int)(k/2);
cc=0;}
k=m/2;
while (k>0)
{ for(int x=0;x<m;x++)
{if((k+x)<=(m-1))
{if (b[x]>b[x+k])
{aux=b[x+k];
b[x+k]=b[x];
b[x]=aux;
cc=cc+1;}}}
if (cc==0)
k=(int)(k/2);
cc=0;}
pg=0;
```

```

for (int nb=0;nb<n+m;nb++)
{c[nb]=0;}
for (int i=0;i<n;i++)
{for (int nb=0;nb<n+m;nb++)
{if (a[i]==c[nb])
{cia++;}}
if (cia==0)
{c[pg]=a[i];
cia=0;
pg=pg+1;}
else
{cia=0;}}
for (int i=0;i<m;i++)
{for (int nb=0;nb<n+m;nb++)
{if (b[i]==c[nb])
{cib++;}}
if (cib==0)
{c[pg]=b[i];
cib=0;
pg=pg+1;}
else
{cib=0;}}
k=(m+n)/2;
while (k>0)
{ for(int x=0;x<m+n;x++)
{if((k+x)<=(n+m-2))
{if (c[x]>c[x+k])
{aux=c[x+k];
c[x+k]=c[x];
c[x]=aux;
cc=cc+1;}}}
if (cc==0)
k=(int)(k/2);
cc=0;}
//mostrar vectores
System.out.println("Los vectores son: ");
for(int f=0;f<n;f++)
{System.out.print(a[f]+" ");}
System.out.println("");
for(int i=0;i<m;i++)
{System.out.print(b[i]+" ");}
System.out.println();
System.out.println("Union:");
System.out.println();

for(int i=0;i<pg;i++)
{System.out.print(c[i]+" ");}
int i,p=0;
System.out.println();
//obtiene y muestra la interseccion
for (i=0;i<n;i++)
{for(int j=0;j<n;j++)
{if (a[i]==b[j])
{aib[p]=b[j];
p++;}}//contador de elementos en la interseccion

```

```

    }
}
if (p==0)
System.out.print("No existen elementos comunes");
else
{System.out.print("Interseccion de A y B ");
for (i=0;i<p;i++)
System.out.print(aib[i]+"");}
System.out.println();
//Diferencia A-B las diferencias se basan en las interseccion
System.out.print("A-B: ");
int y=0;
for (i=0;i<n;i++)
{for(int j=0;j<p;j++)
{if (a[i]==aib[j])
{y++;}}
if (y==0)
System.out.print(a[i]+" ");
y=0;
}
System.out.println();
//Diferencia B-A
System.out.print("B-A: ");
y=0;
for (i=0;i<n;i++)
{for(int j=0;j<p;j++)
{if (b[i]==aib[j])
{y++;}}
if (y==0)
System.out.print(b[i]+" ");
y=0;
}}}

```

6. Realice un programa que lea dos vectores desde teclado y produzca como salida el producto vectorial de los mismos.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite llenar dos vectores, realizar el producto vectorial entre ellos y en un tercer vector guardar y mostrar es mismo.

```

// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_36
{

public static void main (String args[])throws IOException
{
BufferedReader in;

```

```

in = new BufferedReader (new InputStreamReader (System.in));
int n,m;

System.out.println("introduzca el tama^o de los vectores: ");
n = Integer.parseInt(in.readLine());
int [] vector1 = new int [20];
int [] vector2 = new int [20];
int [] vector3 = new int [20];
for(int i=0;i<n;i++)
{System.out.println("introduzca valores del vector 1: ");
m = Integer.parseInt(in.readLine());
vector1[i]=m;}
for(int j=0;j<n;j++)
{System.out.println("introduzca valores del vector 2: ");
m = Integer.parseInt(in.readLine());
vector2[j]=m;}
for(int k=0;k<n;k++)
{
vector3[k]=vector1[k]*vector2[k];
}
System.out.println("El producto vectorial de los dos vectores ingresados es");
for(int h=0;h<n;h++)
System.out.print(" "+vector3[h]);
}}

```

7. Realice un programa que lea dos vectores desde teclado y produzca como salida la suma de los mismos

EXPLICACION DEL PROGRAMA: el programa calcula la suma de 2 vectores introducidos por teclado, para ello guarda la suma en un nuevo vector c y finalmente despliega dicho vector. Se utilizan cuatro ciclos for, uno para ingresar el primer arreglo, otro para el segundo, otro para calcular la suma de dichos vectores, finalmente otro para desplegar el vector que contiene la suma.

```

//Programa realizado por Freddy Valda
import java.io.*;
public class a37
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]a=new int[100];
int[]b=new int[100];
int[]c=new int[100];
int n;

```

```

System.out.println("INGRESE EL TAMANO DE LOS ARREGLOS (IGUAL TAMANO)");
n = Integer.parseInt(in.readLine());
System.out.println("INGRESE EL ARREGLO 1");
for(int q=1;q<=n;q++)
{a[q]=Integer.parseInt(in.readLine());
}
System.out.println("INGRESE EL ARREGLO 2");
for(int w=1;w<=n;w++)
{b[w]=Integer.parseInt(in.readLine());
}

for(int i=1;i<=n;i++)
{c[i]=a[i]+b[i];
}
System.out.println( " LA SUMA DE LOS ARREGLOS ES: ");
for(int j=1;j<=n;j++)
{System.out.print(c[j]+" ");
}
}
}
}

```

8. Realice un programa que lea un vector desde teclado y produzca como salida el producto escalar del mismo.

EXPLICACION DEL PROGRAMA: Este programa multiplica los elementos del vector ingresado y nos muestra el resultado a través de una sentencia for

```

//programa realizado por Ignacio salgado Faller
import java.io.*;
public class a38
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]a=new int[100];
long sum=0;
System.out.println("Introduzca la longitud del vector: ");
int n=Integer.parseInt(in.readLine());
System.out.println("Introduzca los datos del vector(1-100): ");
for(int i=0;i<n;i++)
{a[i]=Integer.parseInt(in.readLine());}
for(int i=0;i<n;i++)
{sum=sum+(a[i]*a[i]);}
System.out.println("El producto escalar es: "+sum);
}
}

```

9. Se tienen N temperaturas almacenados en un vector, se desea calcular su media, la desviación estándar y cuáles de las temperaturas son inferiores a la media.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite guardar en un vector n temperaturas ingresadas por teclado y a su vez poder calcular su media, su desviación estándar y además verificar y mostrar todas aquellas temperaturas que se encuentren por debajo de su media.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_40
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        double g=0, f, t, p, e,n, m, s = 0, aux;

        System.out.println ("introduzca el numero de temperaturas a registrar: ");
        n = Integer.parseInt (in.readLine ());
        double [] vector = new double [20];
        for (int i = 0 ; i < n ; i++)
        {
            System.out.println ("introduzca las temperaturas: ");
            m = (Double.valueOf (in.readLine ())).doubleValue ();
            vector [i] = m;
            s = s + m;
        }
        aux = s/n;
        System.out.println ("La temperatura media es: " + aux);
        System.out.println ("Las temperaturas por debajo de la temperatura media son: ");
        for (int k = 0 ; k < n ; k++)
        {
            if (vector [k] < aux)
                System.out.print (" " + vector [k]);
            g = g+Math.pow((vector [k] - aux),2);
        }

        p =g/n;
        e = Math.sqrt (p);
        System.out.println ();
        System.out.println ("La desviacion es: " + e);
    }
}
```

10. Escribir un programa para generar un vector de 20 elementos con números aleatorios reales(con dos decimales) y luego hallar el máximo y mínimo elemento.

EXPLICACION DEL PROGRAMA: Este programa genera un vector aleatorio, el cual es llenado con números reales. Luego halla el mínimo y el máximo valor a través de comparaciones utilizando if-else.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
import java.math.*;
import java.text.DecimalFormat;
public class a41
{
    public static void main (String args[])throws IOException
    {
        double[]vector=new double[20];
        DecimalFormat df = new DecimalFormat("0.00");
        double a=vector[0];
        double b=200;
        for (int i=0;i<20;i++)
        {vector[i]=(double)(Math.random()*100);
        if (vector[i]>a)
        {a=vector[i];}
        if (vector[i]<b)
        {b=vector[i];}
        }
        System.out.println("El vector es: ");
        for (int o=0;o<20;o++)
        {System.out.print(df.format(vector[o])+" ");}
        System.out.println("El mayor es: "+df.format(a));
        System.out.println("El menor es: "+df.format(b));
    }
}
```

11. Elabore un programa que inserte un elemento en un arreglo ordenado. Pensar en un método distinto al secuencial.

EXPLICACION DEL PROGRAMA: Este programa inserta un elemento a un arreglo introducido por el usuario. El mismo es ordenado antes y después de ingresar el nuevo elemento y mostrado en ambas ocasiones.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a42
{
    public static void main(String args[])throws IOException
```

```

{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]a=new int[100],dig;
int sum=0,k,aux,cc=0;
System.out.println("Introduzca la longitud del vector: ");
int n=Integer.parseInt(in.readLine());
System.out.println("Introduzca los datos del vector(1-100): ");
for(int i=0;i<n;i++)
{a[i]=Integer.parseInt(in.readLine());}
System.out.println("Su vector ordenado es: ");
k=n/2;
while (k>0)
{ for(int x=0;x<n;x++)
{if((k+x)<=(n-1))
{if (a[x]>a[x+k])
{aux=a[x+k];
a[x+k]=a[x];
a[x]=aux;
cc=cc+1;}}}

if (cc==0)
k=(int)(k/2);
cc=0;}
for(int i=0;i<n;i++)
{System.out.print(a[i]+" ");}
System.out.println("");
System.out.println("Que numero desea ingresar? ");
a[n]=Integer.parseInt(in.readLine());
System.out.println("Su nuevo vector es: ");
k=n/2;
while (k>0)
{ for(int x=0;x<n;x++)
{if((k+x)<=(n))
{if (a[x]>a[x+k])
{aux=a[x+k];
a[x+k]=a[x];
a[x]=aux;
cc=cc+1;}}}

if (cc==0)
k=(int)(k/2);
cc=0;}
for(int i=0;i<=n;i++)
{System.out.print(a[i]+" ");}
}
}

```

12. Realizar un programa que permita calcular el producto escalar de dos vectores de dimensión 2000 en los cuales al menos 99% de sus elementos son cero.

EXPLICACIÓN DEL PROGRAMA: Este programa genera un vector de tamaño 2000 en el que 99% de sus elementos son ceros. Primero se establecen los límites y después el llenado de datos en cada vector para proceder al producto escalar.

```
// Programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a43
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));

int n=2000,nca=0,pos=0,ncb=0,xx=2000,yy=2000;
int[]a=new int[n];
int[]b=new int[n];
int[]auxa=new int[n];
int[]auxb=new int[n];

for(int i=0;i<n;i++)
{auxa[i]=0;
auxb[i]=0;
a[i]=0;
b[i]=0;
}
nca=(int)(Math.random()*20+1980);
ncb=(int)(Math.random()*20+1980);
xx=xx-nca;
yy=yy-ncb;

for (int u=1;u<=xx;u++)
{pos=(int)(Math.random()*1999);
if (auxa[pos]==0)
{a[pos]=(int)(Math.random()*100);
auxa[pos]=1;}
else
{u=u-1;}}

for (int g=1;g<=yy;g++)
{pos=(int)(Math.random()*1999);
if (auxb[pos]==0)
{b[pos]=(int)(Math.random()*100);
auxb[pos]=1;}
else
{g=g-1;}}

int pc=0;
for (int i=0;i<n;i++)
{
pc=pc+(a[i]*b[i]);
}
System.out.print("Producto escalar: "+pc); }}
```

13. Realice un programa que elimine elementos duplicados de una lista y por cada elemento indique la cantidad de veces que se repite en la lista.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite eliminar los elementos repetidos de un arreglo y a su vez nos informa sobre cuantas veces se repitió el elemento.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_44
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int n, m;

        System.out.println ("introduzca la cantidad de numeros a verificar: ");
        n = Integer.parseInt (in.readLine ());
        int [] vector = new int [20];
        for (int i = 0 ; i < n ; i++)
        {
            System.out.println ("introduzca los valores: ");
            m = Integer.parseInt (in.readLine ());
            vector [i] = m;
        }
        System.out.print ("El vector es:");
        System.out.println();
        for (int j = 0 ; j < n ; j++)
        {
            System.out.print (" " + vector [j]);
        }
        for (int k = 0 ; k < n ; k++)
        {
            int c = 0;
            for (int l = k + 1 ; l < n ; l++)
            {
                if (vector [k] == vector [l])
                {
                    c++;
                    vector [l] = 0;
                }
            }
            if (vector [k] != 0)
            {
                System.out.println ();
                System.out.println ("El numero " + vector [k] + " " + "Se repite " + c +
"veces");
            }
        }
        System.out.println (" ");
        System.out.print ("El vector sin elementos repetidos es:");
    }
}
```

```

for (int q = 0 ; q < n ; q++)
{
    System.out.print (" " + vector [q]);}}

```

14. Desarrollar un programa que utilice funciones para generar dos vectores A y B de 20 valores cada uno, tal que:

A={1,2,2,3,3,3,4,4,4,4,,.....}

B={0,1,1,2,3,5,8,.....}

- Mediante una función calcular $C(i)=A(i)+B(i)$
- Mediante otra desplegar en pantalla los vectores A,B,C, incluyendo los encabezados que usted crea conveniente

EXPLICACION DEL PROGRAMA: el programa genera las series propuestas mediante funciones, la segunda es la serie de Fibonacci. Utilizando funciones calcula en un nuevo vector la suma de las series y finalmente despliega cada vector

```

//Programa realizado por Andres Prudencio R.
import java.io.*;
public class a45
{
    public static void main (String args[])
    {
        int n=20;
        int[]a=new int[n];
        int[]b=new int[n];
        int[]c=new int[n];
        llena_a(n,a);
        llena_b(n,b);
        calcula(n,a,b,c);
        muestra(n,a,b,c);
    }
    public static void llena_a(int n,int a[])
    { int h=1,f=0;
      while(f<n)
      {
          for (int j=1;h>=j;j++)
          { if(f<n)

              a[j]=h;
              f++;}
              h++;
      }
    }
}

```

```

    }

    public static void llena_b(int n,int b[])
    {int e=0,r=1,aux;
      for (int i=0;i<n;i++)
      {b[i]=e;
       aux=e;
       e=e+r;
       r=aux;}}

    public static void calcula(int n,int a[],int b[],int c[])
    {for (int i=0;i<n;i++)
      c[i]=a[i]+b[i];}
    public static void muestra(int n,int a[],int b[],int c[])

    {System.out.println("Vector A:");
    for (int o=0;o<n;o++)
      {System.out.print(a[o]+" ");}
    System.out.println();
    System.out.println("Vector B:");
    for (int o=0;o<n;o++)
      {System.out.print(b[o]+" ");}
    System.out.println();
    System.out.println("Vector C:");
    for (int o=0;o<n;o++)
      {System.out.print(c[o]+" ");}
    }}

```

15. Escribir un programa para que se llene un vector con N números aleatorios reales, otra función despliegue en pantalla el contenido del vector, con otra función se busque el menor y mayor elemento, mediante otra función se sume los valores del vector.

EXPLICACION DEL PROGRAMA: Este programa llena un vector con números aleatorios mediante una sentencia for y luego realiza comparaciones para mostrar el menos y el mayor valor.

```

//programa realizado por Ignacio Salgado Faller
import java.io.*;
import java.text.DecimalFormat;
public class a46
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    double[]a=new double[100];

```

```

double sum=0;
System.out.println("Introduzca la longitud del vector: ");
int n=Integer.parseInt(in.readLine());
    DecimalFormat df = new DecimalFormat("0.00");
    double ax=0;
    double b=200;
    for (int i=0;i<n;i++)
    {a[i]=(double)(Math.random()*100);
    sum=sum+a[i];
    if (a[i]>ax)
    {ax=a[i];}
    if (a[i]<b)
    {b=a[i];}}
    System.out.println("El vector es: ");
    for (int o=0;o<n;o++)
    {System.out.print(df.format(a[o])+" ");}
    System.out.println("");
    System.out.println("El mayor es: "+df.format(ax));
    System.out.println("El menor es: "+df.format(b));
    System.out.println("La suma es: "+df.format(sum));
    }}

```

16. Escribir un programa para que mediante una función se llene el vector A con N números aleatorios enteros, luego mediante otra función crear un vector B con los elementos de A que ocupan las posiciones pares, mediante una función despliegue en pantalla los dos vectores.

EXPLICACION DEL PROGRAMA: El programa primeramente determina los tamaños de ambos vectores para que las funciones no impriman ceros excedentes. Posteriormente en el método main se llaman a las funciones en las cuales se introducen los datos para cada vector y posteriormente se los muestran. En la función llena_b, las posiciones del vector b van de dos en dos para guardarlas en el vector b con un contador.

```

//Programa realizado por Andres Prudencio R.
import java.io.*;
public class a47
{
    public static void main (String args[])throws IOException
    { BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));

    System.out.print("Ingresar el tamaño del vector A: ");
    int n=Integer.parseInt(in.readLine());
    int nb;

```

```

        if(n%2==0)
        {nb=(n/2);}
    else
    {nb=(n/2)+1;}
    int[]a=new int[n];
    int[]b=new int[n];
    // funciones:
    llena_a(n,a);
    llena_b(n,a,b);
    muestra(n,nb,a,b);
    }
    public static void llena_a(int n,int a[])
    {for (int i=0;i<n;i++)
        a[i]=(int)(Math.random()*89+10);}
    public static void llena_b(int n,int a[],int b[])
    {int d=0;b[0]=a[0];
    int i=2;
    int e=1;
    while(n>i)
    {b[e]=a[i];
    e++;
    i=i+2;}
    }
    public static void muestra(int n,int nb,int a[],int b[])

    {for (int o=0;o<n;o++)
        {System.out.print(a[o]+" ");}
    System.out.println();
    for (int o=0;o<nb;o++)
        {System.out.print(b[o]+" ");} }
}

```

17. Dada una sucesión de números comprendidos entre 1 y 100, almacenados en un vector, escriba un programa que calcule todos los divisores de cada número.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite llenar un vector de forma aleatoria n números comprendidos entre el 1 y 100 y además nos calcula los divisores de dichos números.

```

// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_48
{

    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
    }
}

```

```

int n, m;

System.out.println ("Ingrese el tam*o del vector: ");
n = Integer.parseInt (in.readLine ());
int [] vector = new int [20];
for (int i = 0 ; i < n ; i++)
{
    int k = (int) (Math.random () * 100 - 1);
    vector [i] = k;
}
System.out.println ("El Vector es ");
for (int j = 0 ; j < n ; j++)
{
    System.out.print (" " + vector [j]);
}
System.out.println (" ");
for (int d = 0 ; d < n ; d++)
{
    System.out.println ("<Los divisores del> " + vector [d] + " son:");
    for (int j = 1 ; j <= vector [d] ; j++)
    {
        if (vector [d] % j == 0)
            System.out.print (" " + j + ",");
    }
    System.out.println();}}}

```

18. Hacer un programa que lea diez valores enteros en un array y los muestre en pantalla. Después que los ordene de menor a mayor y los vuelva a mostrar. Y finalmente que los ordene de mayor a menor y los muestre por tercera vez. Para ordenar la lista usar una función que implemente el método de la burbuja y que tenga como parámetro de entrada el tipo de ordenación, de mayor a menor o de menor a mayor. Para el array usar una variable global.

EXPLICACION DEL PROGRAMA: El programa lee 10 elementos y los ordena de menor a mayor y viceversa mediante una función que recibe 1 para ordenar de mayor a menor y 2 para ordenar de menor a mayor. Finalmente despliega el arreglo y el mismo ordenado de las 2 formas.

```

//Programa realizado por Freddy Valda
import java.io.*;
public class a68
{
    public static void main(String args[])throws IOException

```

```
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]a=new int[15];
int[]b=new int[15];
System.out.println("INGRESE LOS 10 VALORES DEL ARREGLO ");
for(int i=1;i<=10;i++)
{a[i]=Integer.parseInt(in.readLine());
}
b=a;
System.out.println("EL ARREGLOS ES: ");
mostrar(a);
System.out.println("");
System.out.println("EL ARREGLO ORDENADO DE MAYOR A MENOR ES: ");
ordena(a,1);
System.out.println("");
System.out.println("EL ARREGLO ORDENADO DE MENOR A MAYOR ES: ");

ordena(b,2);

}

static void mostrar(int a[])
{for(int i=1;i<=10;i++)
{System.out.print(" "+a[i]);
}}

static void ordena(int a[],int k)
{int aux;
aux=a[1];
if(k==1)
//ORDENA DE MAYOR A MENOR
{
for(int i=1;i<=10;++i)
{for(int j=i;j<=10;++j)
{
if(a[i]<a[j])
{
aux=a[i];
a[i]=a[j];
a[j]=aux;
}
}
}
}

//ORDENA DE MENOR A MAYOR
else
{
for(int yy=1;yy<=10;yy++)
{
for(int ii=yy;ii<=10;ii++)
{if(a[ii]<a[yy])
{aux=a[ii];
a[ii]=a[yy];
a[yy]=aux;
}
```



```

a[yy]=aux;}}}}
mostrar(a);
}
}

```

19. a) Escribir una función que realice una búsqueda secuencial (lineal) de un elemento dentro de un arreglo de enteros.
- b) Idem pero para Búsqueda Binaria.

EXPLICACION DEL PROGRAMA: Este programa nos ofrece dos métodos de búsqueda: la secuencial y la binaria. Cada método tiene sus propias características.

```

//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a89
{
    public static void main (String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    int[]x=new int[100];
    int n;
    int opc=3,bus=0,ii=0,is=0,im=0;
    System.out.println("Introduzca la longitud del vector: ");
    n=Integer.parseInt(in.readLine());
    System.out.println("Introduzca los datos del arreglo: ");
    for(int i=0;i<n;i++)
    {x[i]=Integer.parseInt(in.readLine());}
    System.out.println("Ingresar el dato a buscar: ");
    bus=Integer.parseInt(in.readLine());
    while (opc>2||opc<1)
    {System.out.println("1. Buscar un dato realizando una busqueda secuencial: ");
    System.out.println("2. Buscar un dato realizando una busqueda binaria: ");
    System.out.println("Ingresar opcion: ");
    opc=Integer.parseInt(in.readLine());
    }
    switch (opc)
    {case 1: for (int i=0;i<n;i++)
        {if (x[i]==bus)
        {System.out.println ("Se encontro el numero "+bus+" en la posicion "+(i+1));
        return;}}
    System.out.println ("No se encontro el elemento");
    case 2: is=n-1;
        while (ii<=is)
        {im =(int)(ii+is)/2;
        if(bus>x[im])
        {ii=im+1;}
        }
    }
    }

```

```
        else
        {if(bus<x[im])
        {is=im-1;}
        else
        {break;}}}
    if (ii>is)
    {System.out.println ("Se encontro el numero "+bus);}
    else
    {System.out.println ("No se encontro el elemento");}
}}
```

20. Obtenga la suma de los elementos de un array lineal que contiene números enteros.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite calcular la suma de los elementos de un arreglo ingresados por teclado.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class B_5
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int n,m,s = 0;

        System.out.println ("introduzca la cantidad de numeros a evaluar: ");
        n = Integer.parseInt (in.readLine ());
        int [] vector = new int [20];
        for (int i = 0 ; i < n ; i++)
        {
            System.out.println ("introduzca los valores: ");
            m = Integer.parseInt(in.readLine ());
            vector [i] = m;
            s = s + m;
        }System.out.println ("El Vector ingresado es:");
        for (int i = 0 ; i < n ; i++)
        {System.out.print(vector[i]+" ");}
        System.out.println ();
        System.out.println ("La suma de los elementos del vector es: " + s);
    }
}
```

21. Realice un programa para leer un arreglo ingresando solamente números múltiplos de 3 que ingresan por teclado.

EXPLICACION DEL PROGRAMA: el programa lee el tamaño de un arreglo y pide los valores. Los guarda en el arreglo solo si son múltiplos de 3 determinando si son o no múltiplos de 3 utilizando la sentencia % que es mod

```
//Programa realizado por Freddy Valda
import java.io.*;
public class b12
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    int a,m,p=0;
    int []c=new int [50];
    System.out.println("Ingrese la cantidad de numeros a guardar en el array:");
    a=Integer.parseInt(in.readLine());
    for(int i=0;i<a;)
    {
        System.out.println("Ingrese un valor,multiplo de 3");
        m=Integer.parseInt(in.readLine());
        if(m%3==0)
        {p++;
        c[i]=m;
        i++;
        }
        else
        System.out.println("valor ingresado no es multiplo de 3");
    }
    System.out.println("el arreglo de multiplos de 3 es:");
    for(int j=0;j<p;j++)
    {System.out.print(c[j]+" ");
    }
    }
}
```

22. Obtenga la sumatoria y la media de los elementos de un arreglo.

EXPLICACIÓN DEL PROGRAMA: Este programa nos determina la suma y la media de n elementos ingresados por teclado y asu ves guardados en un arreglo.

```
// Programa realizado por Sergio W. Ossio Marin
```

```

import java.io.*;
public class B_13
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int aux,n,m,s = 0;

        System.out.println ("introduzca la cantidad de numeros a evaluar: ");
        n = Integer.parseInt (in.readLine ());
        int [] vector = new int [20];
        for (int i = 0 ; i < n ; i++)
        {
            System.out.println ("introduzca los valores: ");
            m = Integer.parseInt(in.readLine ());
            vector [i] = m;
            s = s + m;
        }aux=s/n;

        System.out.println ("El Vector ingresado es:");
        for (int i = 0 ; i < n ; i++)
        {System.out.print(vector[i]+" ");}
        System.out.println ();
        System.out.println ("La suma de los elementos del vector es: " + s);
        System.out.println ("La media de los elementos del vector es: " + aux);
    }
}

```

Aplicación de vectores – Programa estadístico

23. Realizar un POO donde aplique funciones de acuerdo al requerimiento. La clase debe llamarse estadística:

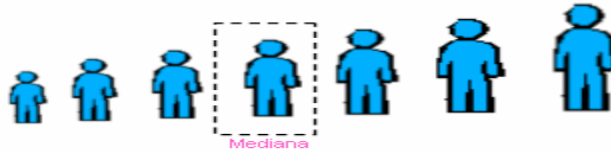
- A Lectura de un arreglo de n elementos.
- B Sumatoria de los elementos del arreglo
- C Listado de los elementos del arreglo
- D Encontrar el elemento máximo del arreglo.
- E Ordenar el arreglo.
- F Encontrar el elemento mínimo del arreglo
- G Encontrar el rango (elemento máximo – elemento mínimo)
- H Obtener la media de los elementos del arreglo

Media aritmética:

$$\bar{x} = \frac{\sum_{i=1}^n x_i}{n}$$

I La mediana, es el valor que separa por la mitad los elementos del arreglo ordenado de menor a mayor, de tal forma que el 50% de estas son menores que la mediana y el otro 50% son mayores. Si el número de datos es impar la

mediana será el valor central, si es par se toma como mediana la media aritmética de los valores centrales.



J Moda es el valor de la variable que más veces se repite, es decir aquella cuya frecuencia absoluta es mayor, no tiene porqué ser única.

K

Varianza:

$$s^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n}$$

L

Desviación típica:

$$s = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n}}$$

M

Coef. Variación:

$$CV = \frac{s}{\bar{x}}$$

N Verificar si los elementos del arreglo tienen una distribución simétrica. Es simétrica, cuando su mediana, moda y su media aritmética coinciden.

EXPLICACION DEL PROBLEMA: El programa primeramente pide el tamaño de un vector y sus datos. Este vector será analizado, sin embargo, se puede leer otro vector una vez que el menú de opciones se muestre en pantalla. Este es mostrado mediante while y switch. El usuario puede escoger cualquier opción, no necesita ejecutar todas las 13 primeras opciones para obtener el resultado ya que si una opción tiene una función que requiere de otras, estas serán llamadas automáticamente.

OBSERVACIONES: No se uso sobrecarga de operadores ni de funciones

```
//Realizado por Andres Prudencio R
import java.io.*;
public class estadistica
{
public static void main(String args[])throws IOException
{BufferedReader in;
```

```
in = new BufferedReader (new InputStreamReader (System.in));
System.out.print("TAMANIO DEL VECTOR: ");
int n=Integer.parseInt(in.readLine());
int[]vector=new int[100];
for(int i=0;i<n;i++)
{
    System.out.println("INGRESE EL ELEMENTO  "+(i+1));
    vector[i]=Integer.parseInt(in.readLine());
}
int re=1,max=0,min=0,r=0;

float media=0,moda=0,mediana=0,sum=0;
double varza=0,dt=0,CV=0;

while(re!=16)
{
    System.out.println("---MENU DE OPCIONES---");
    System.out.println("1. LECTURA DE OTOR ARREGLO ");
    System.out.println("2. SUMATORIA DE LOS ELEMENTOS");
    System.out.println("3. LISTADO DE UN ARREGLO");
    System.out.println("4. ENCONTRAR EL ELEMENTO MAXIMO ");
    System.out.println("5. ORDENAR EL ARREGLO DE FORMA ASCENDENTE");
    System.out.println("MOSTRAR:");
    System.out.println("6. EL ELEMENTO MINIMO ");
    System.out.println("7. EL RANGO ");
    System.out.println("8. LA MEDIA(el vector sera ordenado de forma ascendente ");
    System.out.println("9. LA MEDIANA ");
    System.out.println("10. LA MODA ");
    System.out.println("11. LA VARIANZA ");
    System.out.println("12. LA DESVIACION TIPICA ");
    System.out.println("13. EL COEFICIENTE DE VARIACION");
    System.out.println("14. DETERMINAR SI EL VECTOR ES SIMETRICO");
    System.out.println("==> Cualquier otro numero para salir");
    System.out.println("INGRESE LA OPCION");
    re=Integer.parseInt(in.readLine());

    switch (re){
        case 1:
            lectura(n,vector);
            break;
        case 2: System.out.print("La suma de los elementos es: "+sumatoria(n,vector));
            sum=sumatoria(n,vector);break;
        case 3: listado(n,vector);break;
        case 4: System.out.print("El elemento maximo es: "+maximo(n,vector));
            max=maximo(n,vector);
            System.out.println();
            break;
        case 5: ordena(n,vector);listado(n,vector);break;
        case 6: System.out.println("El elemento minimo es: "+minimo(n,vector));
            min=minimo(n,vector);
            break;
        case 7:
            minimo(n,vector);maximo(n,vector);
            r=rango(max,min);System.out.println("El rango de este vector es "+r);
```

```
break;
case 8:
sum=sumatoria(n,vector);
media=omedia(n,sum);System.out.println("La media es "+media);
break;
case 9:
ordena(n,vector);
mediana=omediana(n,vector);
System.out.println("La mediana es "+mediana);
break;
case 10:
moda=omoda(n,vector);
if (moda!=0)
System.out.println("La moda es "+moda);
else
sum=sumatoria(n,vector);
moda=omedia(n,sum)*-1;
break;
case 11:
sum=sumatoria(n,vector);
media=omedia(n,sum);
System.out.println("Varianza: "+varianza(n,vector,media));
break;
case 12:
sum=sumatoria(n,vector);
media=omedia(n,sum);
varza=varianza(n,vector,media);
System.out.println("Desviacion Tipica:"+dtipica(varza));
break;
case 13:
sum=sumatoria(n,vector);
media=omedia(n,sum);
varza=varianza(n,vector,media);
dt=dtipica(varza);
System.out.println("Coeficiente de Variacion: "+cv(media,dt));
break;
case 14:
sum=sumatoria(n,vector);
media=omedia(n,sum);
mediana=omediana(n,vector);
moda=omoda(n,vector);
if (media==mediana&&mediana==moda)
System.out.println("El vector es simetrico");
else
System.out.println("El vector no es simetrico");
break;
default:System.exit(0);
break;
}
}}
//FUNCIONES:
public static void lectura(int n,int vector[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
System.out.println("Ingresa el tamano del vector");
```

```
n=Integer.parseInt(in.readLine());
for(int i=0;i<n;i++)
{
    System.out.println("INGRESE EL ELEMENTO "+(i+1));
    vector[i]=Integer.parseInt(in.readLine());
}
}

public static int sumatoria(int n,int vector[])
{int sum=0;
for (int i=0;n>i;i++)
{sum=sum+vector[i];}
return sum;
}

public static void listado(int n,int vector[])
{
for (int i=0;n>i;i++)
{System.out.print(vector[i]+" ");}
System.out.println();
}

public static int maximo(int n,int vector[])
{int max=vector[0];
for (int i=0;i<n;i++)
{if (vector[i]>max)
max=vector[i];}
return max;
}

public static void ordena(int n,int vector[])
{int aux;
for(int i=0;i<n;++i)
for(int j=0;j<n;++j)
{
    if(vector[i]<vector[j])
    {
        aux=vector[i];
        vector[i]=vector[j];
        vector[j]=aux;
    }
}
}

public static int minimo(int n,int vector[])
{int min=vector[0];
for (int i=0;i<n;i++)
{if (vector[i]<min)
min=vector[i];}
return min;
}

public static int rango(int max,int min)
{int r=max-min;
return r;}

public static float omedia(int n,float sum)
{float media;
```



```
media=(float)(sum/n);
return media;
}
public static float omediana(int n,int vector[])
{ float mediana=0,s,nn=n;
  int a1=0,a2=0,a3=0,a4=0;

  if(n%2!=0)
  {s=n/2;
   for(int u=0;u<n;u++)
   {if(u==s)
    mediana=vector[u];}
   return mediana;
  }
  else
  {if(n%2==0)
   a1=(n/2)-1;
   a2=n/2;
   for(int e=0;e<n;e++)
   { if(e==a1)
    a3=vector[e];
    if(e==a2)
    a4=vector[e];
   }
   mediana=(a3+a4)/2;
  }
  return mediana;
}

public static float omoda(int n,int vector[])
{float moda=0;
int maxim=0,minim=0,t=0;
for (int i=0;i<n;i++)
{minim=0;
  for(int j=0;j<n;j++)
  {if(i!=j)
   {if(vector[i]==vector[j])
    {minim++;}}}
  if (minim>maxim)
  {maxim=minim;
  t=vector[i];
  }}
  minim=0;
  if (maxim==minim)
  {
  System.out.println("No hay moda");
  return 0;}
  else
  {
  for(int k=0;k<n;k++)
  {
  if (t==vector[k])
  {
  moda=vector[k];
```

```
}}  
  
return moda;  
}  
}  
  
public static float varianza(int n,int vector[],float media)  
{  
    float sum=0,va,sums;  
    float d=n;  
    for (int i=0;i<d;i++)  
    {sums=vector[i]-media;  
    sum=sum+(sums*sums);  
    }  
    va=sum/d;  
    return va;  
}  
public static double dtipica(double varza)  
{double dt=Math.sqrt(varza);  
return dt;  
}  
public static double cv(float media,double dt)  
{double CV=dt/media;  
return CV;  
}  
}
```

PROBLEMAS PROPUESTOS

1. escriba un programa que permita encontrar la sumatoria de los números impares comprendidos desde 50 a 1000 guardados en un arreglo
2. elabore un programa que permita introducir 20 elementos de tipo entero en un arreglo, el programa mostrara impreso el arreglo en orden inverso
3. elabore un programa que permita encontrar el primer y segundo mayor de un arreglo de 15 elementos

4. hacer un programa que lea diez valores enteros en un array y los muestre en pantalla. después que los ordene de menor a mayor y los vuelva a mostrar. y finalmente que los ordene de mayor a menor y los muestre por tercera vez. para ordenar la lista usar una función que implemente el método de la burbuja
5. elabore un programa que encuentre al número mayor y menor de un arreglo y luego muestre en qué posición se encontraban estos números originalmente.
6. elabore un programa que imprima en orden inverso una cadena de caracteres
7. elabore un programa que permita introducir un arreglo de 10 elementos, el programa mostrara un histograma de esos datos (el histograma se interpretara con la salida de n asteriscos donde n es el valor de cada elemento del arreglo) ej: el arreglo es 2,3,4 el histograma será 2->**
3->***
4->****
8. elabore un programa que permita introducir un arreglo de 25 elementos de tipo entero. luego pedir al usuario que introduzca un número. el programa mostrara el numero de veces que se repite dicho valor en el arreglo
9. elabore un programa que permita encontrar el primer y segundo menor de un arreglo de 15 elementos
10. elabore un programa que permita introducir un arreglo de 8 elementos de tipo entero. el programa mostrara un arreglo en donde muestre un 1 para los primos y un 0 para los no primos

Matrices

Los siguientes programas utilizan matrices para su solución. Son programas orientados a la utilización y manejo de matrices. Una matriz es una tabla o arreglo rectangular de números o, más generalmente, una tabla consistente en cantidades abstractas que pueden sumarse y multiplicarse.

1. Escribir un programa que transponga una matriz sin emplear una matriz auxiliar.

EXPLICACION DEL PROGRAMA: el programa transpone una matriz de orden n,m sin usar una matriz auxiliar. Para llevar a cabo dicho propósito al ingresar la matriz se guardan las filas en vez de columnas y viceversa, finalmente se muestra la matriz transpuesta

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a49
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int[][]a=new int[50][50];
    int n,m;
    System.out.println("INGRESE EL NUMERO DE FILAS DE LA MATRIZ");
    n =Integer.parseInt(in.readLine());
    System.out.println("INGRESE EL NUMERO DE COLUMNAS DE LA MATRIZ");
    m =Integer.parseInt(in.readLine());
    System.out.println("INGRESE LA MATRIZ");
    for(int f=1;f<=n;f++)
    {System.out.println("INGRESE LA FILA "+f);
    for(int c=1;c<=m;c++)
    {
        a[c][f]=Integer.parseInt(in.readLine());
    }
    }

    System.out.println( " LA MATRIZ TRANSPUESTA ES: ");

    for(int q=1;q<=m;q++)
    {
        for(int w=1;w<=n;w++)
        {System.out.print(" "+a[q][w]);
        }
    }
}
```

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

2. Escribir un programa que copie un arreglo A de dimensión $N1 = N \times N$ en una matriz de M de orden $N \times N$, por filas

EXPLICACION DEL PROGRAMA: Este programa permite que el usuario ingrese el orden de una matriz y llene así un vector con $n \times n$ datos. Una vez leídos los datos, estos se colocan en la matriz y esta se despliega

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a50
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int[]a=new int[100];
    int cont=0;
    int [][]mat=new int[100][100];
    System.out.println("Introduzca el orden de la matriz: ");
    int n=Integer.parseInt(in.readLine());
    System.out.println("Introduzca los "+n*n+" datos del vector: ");
    for(int i=0;i<n*n;i++)
    {a[i]=Integer.parseInt(in.readLine());}
    for (int j=0;j<n;j++)
    {for (int k=0;k<n;k++)
    {mat[j][k]=a[cont];
    cont++;}
    }
    System.out.println("Su matriz es: ");
    for (int j=0;j<n;j++)
    {System.out.println("");
    for (int k=0;k<n;k++)
    {System.out.print(mat[j][k]+" " );}}
    }
}
```

3. Escribir un programa que muestre en pantalla los elementos de una matriz al recorrer esta en espiral partiendo desde el elemento 1,1 en sentido de las agujas del reloj.

EXPLICACION DEL PROGRAMA: el programa llena y muestra una matriz con valores desde el 1 hasta el max*max. max es la variable que determina el tamaño de la matriz.

```
//Realizado por Andrés Prudencio R.
import java.io.*;
public class a51
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int MAX;
        System.out.println ("Ingrese el orden de la matriz cuadrada");
        MAX = Integer.parseInt (in.readLine ());
        int [] [] a = new int [MAX] [MAX];

        int i, j, k = 1;
        for (int d = 1 ; d < MAX / 2 + 1 ; d++)
        {
            i = j = d - 1;
            for (; j < MAX - d ; j++)
                a [i] [j] = k++;
            for (; i < MAX - d ; i++)
                a [i] [j] = k++;
            for (; j >= d ; j--)
                a [i] [j] = k++;
            for (; i >= d ; i--)
                a [i] [j] = k++;
        }
        a [MAX / 2] [MAX / 2] = MAX * MAX;

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

4. Escribir un programa que verifique que una matriz M de orden N*N (cuadrada) es una matriz simétrica.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite verificar si una matriz ingresada de tamaño n*n (cuadrada) sea simétrica o no.

```
// Programa realizado por Sergio W. Ossio Marin
```

```
import java.io.*;
public class A_52
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int n, h = 0;
        int matriz [] [] = new int [10] [10];
        System.out.println ("Ingrese el numero de filas y columnas: ");
        n = Integer.parseInt (in.readLine ());
        for (int f = 0 ; f < n ; f++)
        {
            for (int c = 0 ; c < n ; c++)
            {
                System.out.println ("Introduzca un valor");
                matriz [f] [c] = Integer.parseInt (in.readLine ());
            }
        }
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                System.out.print (matriz [i] [j] + " ");
            }
            System.out.println ();
        }
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                if (i!= j)
                {
                    if (matriz [i] [j] != matriz [j] [i])
                        h++;
                }
            }
        }
        if (h != 0)
            System.out.println ("No es simetrica");
        else
            System.out.println ("Es simetrica");
    }
}
```

5. Escribir un programa para que mediante funciones se llene una matriz de N filas por M columnas con números aleatorios reales, mostrar en pantalla la matriz, luego se sumen las diagonales de esta matriz.

EXPLICACION DEL PROGRAMA: el programa pide primero el número de filas y de columnas de la matriz, luego mediante un ciclo for llena la matriz.

Finalmente se utilizan 2 variables sum1 y sum2 para guardar la suma de la diagonal principal y secundario respectivamente, para mostrar finalmente la matriz y las sumas de sus diagonales.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a53
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    float[][]a=new float[100][100];
    int n,m;
    float sum1=0,sum2=0;
    System.out.println("INGRESE EL NUMERO DE FILAS DE LA MATRIZ");
    n =Integer.parseInt(in.readLine());
    System.out.println("INGRESE EL NUMERO DE COLUMNAS DE LA MATRIZ");
    m =Integer.parseInt(in.readLine());
    llenar(a,n,m);

    System.out.println( " LA MATRIZ ES: ");

    for(int q=1;q<=n;q++)
    {
        for(int w=1;w<=m;w++)
        {System.out.print(" "+a[q][w]);
        }
        System.out.println("");
    }

    for(int qq=1;qq<=n;qq++)
    {for(int ww=1;ww<=m;ww++)
    {
        if (qq==ww)
        sum1=sum1+a[qq][ww];
        else
        ;
        if((qq+ww)==(n+1))
        sum2=sum2+a[qq][ww];
        else
        ;
    }
    }
    System.out.println("LA SUMA DE LA DIAGONAL PRINCIPAL ES:"+sum1);
    System.out.println("LA SUMA DE LA DIAGONAL SECUNDARIA ES:"+sum2);
    }
    public static float[][] llenar(float a[][],int n,int m)
    {for(int f=1;f<=n;f++)
    {for(int c=1;c<=m;c++)
    {a[f][c]=(float)Math.random()*100;}}
    return a;
    }
}
```


6. Escribir un programa para que mediante funciones se llene una matriz con los siguientes números:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 1 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 1 & 1 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 1 & 1 & 1 & 2 & 3 & 4 & 5 \\ 1 & 1 & 1 & 1 & 1 & 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

luego desplegar en pantalla esta matriz.

EXPLICACION DEL PROGRAMA: Este programa genera la matriz mencionada arriba permitiendo al usuario ingresar el orden de la misma, utilizando sentencias for e if-else para comprobar que la fila sea mayor o igual a la columna para llenarla con 1's.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a54
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int cont;
int [][]mat=new int[100][100];
System.out.println("Introduzca el orden de la matriz: ");
int n=Integer.parseInt(in.readLine());
for (int j=0;j<n;j++)
{cont=2;
for (int k=0;k<n;k++)
{if (j>=k)
{mat[j][k]=1;}
else
{mat[j][k]=cont;
cont++;}}
}
System.out.println("Su matriz es: ");
for (int j=0;j<n;j++)
{System.out.println("");
for (int k=0;k<n;k++)
```

```

    {System.out.print(mat[j][k]+" ");}
  }
}

```

7. Realice un programa que lea dos matrices de tamaño NxM desde teclado, y devuelva la multiplicación de dichas matrices.

EXPLICACION DEL PROGRAMA: En este programa, primeramente se validan los tamaños de las matrices que se desean multiplicar, puesto que el número de columnas de la primera debe igual al número de filas de la segunda. Posteriormente se leen datos y se multiplican las matrices guardando los resultados en la matriz ca para luego mostrarla.

```

//Programa realizado por Andres Prudencio R.
import java.io.*;
public class a55
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int[][]a=new int[100][100];
    int[][]b=new int[100][100];
    int[][]ca=new int[100][100];
    int f=0,c=0,resp,aux,x,y,suma=0;

    int fi1=0,fi2=1,co1=2,co2=0;
    while(co1!=fi2)
    {System.out.println("Ingrese las filas y columnas de las MATRIZ 1");
    System.out.println("recuerde que las columnas de la primera deben iguales a las
    filas de la segunda");
    fi1 =Integer.parseInt(in.readLine());
    co1 =Integer.parseInt(in.readLine());
    System.out.println("Ingrese las filas y columnas de las MATRIZ 2");
    System.out.println("recuerde que las columnas de la primera deben iguales a las
    filas de la segunda");fi2 =Integer.parseInt(in.readLine());
    co2 =Integer.parseInt(in.readLine());}
    System.out.println("LEER MATRIZ 1");
    for(int iga=1;iga<=fi1;iga++)
    {System.out.println("Ingrese la fila: "+iga);
    for(int jga=1;jga<=co1;jga++)
    {a[iga][jga]=Integer.parseInt(in.readLine());
    }}
    System.out.println("LEER MATRIZ 2");
    for(int iha=1;iha<=fi2;iha++)
    {System.out.println("Ingrese la fila: "+iha);
    for(int jha=1;jha<=co2;jha++)

```

```

{b[iha][jha]=Integer.parseInt(in.readLine());
}}
int suum=0;
for(int i1=1;i1<=fi1;i1++)
{
for(int j1=1;j1<=co2;j1++)
{
suum=0;
for(int kk=1;kk<=co1;kk++)
{sum=(sum+a[i1][kk]*b[kk][j1]);}
ca[i1][j1]=sum;
}}
System.out.println();
for(int i=1;i<=fi1;i++)
{
for(int j=1;j<=co1;j++)
{System.out.print(" "+a[i][j]);
}
System.out.println("");
}
System.out.println(" * ");

for(int i=1;i<=fi1;i++)
{
for(int j=1;j<=co1;j++)
{System.out.print(" "+b[i][j]);
}
System.out.println();}
System.out.println();

for(int q1=1;q1<=fi1;q1++)
{
for(int w1=1;w1<=co2;w1++)
{System.out.print(" "+ca[q1][w1]);
}
System.out.println("");
}
}
}

```

8. Realizar un programa para transponer los elementos de una matriz rectangular.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite transponer los elementos de una matriz rectangular utilizando una serie de ciclos for y comparaciones.

```

// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_56
{

```

```
public static void main (String args []) throws IOException
{
    BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int f, c;
    int matriz1 [] [] = new int [10] [10];
    int matriz2 [] [] = new int [10] [10];
    System.out.println ("Introduzca el numero de filas");
    f = Integer.parseInt (in.readLine ());
    System.out.println ("Introduzca el numero de columnas");
    c = Integer.parseInt (in.readLine ());

    for (int i = 0 ; i < f ; i++)
    {
        for (int j = 0 ; j < c ; j++)
        {
            System.out.println ("Introduzca un valor");
            matriz1 [i] [j] = Integer.parseInt (in.readLine ());
        }
    }
    System.out.println ();
    System.out.println ("La matriz es");
    for (int i = 0 ; i < f ; i++)
    {
        for (int j = 0 ; j < c ; j++)
        {
            System.out.print (matriz1 [i] [j] + " ");
        }
        System.out.println ();
    }
    for (int i = 0 ; i < c ; i++)
    {
        for (int j = 0 ; j < f ; j++)
        {
            matriz2 [i] [j] = matriz1 [j] [i];
        }
    }
    System.out.println ();
    System.out.println ("La matriz transpuesta");
    for (int i = 0 ; i < c ; i++)
    {
        for (int j = 0 ; j < f ; j++)
        {
            System.out.print (matriz2 [i] [j] + " ");
        }
        System.out.println ();
    }
}
```

9. Realizar un programa para intercambiar dos filas (columnas) de una matriz.

EXPLICACION DEL PROGRAMA: el programa intercambia 2 filas o 2 columnas de una matriz, para ello utiliza un menú de opciones y pide las filas o las columnas a intercambiar, finalmente muestra la nueva matriz.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a57
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int[][]a=new int[100][100];
    int n,m;
    int f1,f2,res,aux1,aux2;
    System.out.println("INGRESE EL NUMERO DE FILAS DE LA MATRIZ");
    n =Integer.parseInt(in.readLine());
    System.out.println("INGRESE EL NUMERO DE COLUMNAS DE LA MATRIZ");
    m =Integer.parseInt(in.readLine());

    System.out.println("INGRESE LA MATRIZ");
    for(int f=1;f<=n;f++)
    {System.out.println("INGRESE LA FILA "+f);
    for(int c=1;c<=m;c++)
    {
        a[f][c]=Integer.parseInt(in.readLine());
    }
    }

    System.out.println( " LA MATRIZ INGRESADA ES: ");
    for(int q=1;q<=n;q++)
    {
        for(int w=1;w<=m;w++)
        {System.out.print(" "+a[q][w]);
        }
        System.out.println("");
    }
    System.out.println( "1. INTERCAMBIAR 2 FILAS ");
    System.out.println( "2. INTERCAMBIAR 2 COLUMNAS ");
    System.out.println("INGRESE LA OPCION");
    res=Integer.parseInt(in.readLine());
    switch (res){
    case 1:
        System.out.println( "INGRESE LAS 2 FILAS A CAMBIAR ");
        f1=Integer.parseInt(in.readLine());
        f2=Integer.parseInt(in.readLine());
        for(int ww=1;ww<=m;ww++)
        {aux1=a[f1][ww];
        a[f1][ww]=a[f2][ww];
        a[f2][ww]=aux1;
        }

        break;
    case 2:
        System.out.println( "INGRESE LAS 2 COLUMNAS A CAMBIAR ");
```

```

f1=Integer.parseInt(in.readLine());
f2=Integer.parseInt(in.readLine());
for(int qq=1;qq<=n;qq++)
{aux1=a[qq][f1];
a[qq][f1]=a[qq][f2];
a[qq][f2]=aux1;
}
break;
}
System.out.println( " LA NUEVA MATRIZ ES: ");
for(int r=1;r<=n;r++)
{
for(int e=1;e<=m;e++)
{System.out.print(" "+a[r][e]);
}
System.out.println("");
}
}
}
}

```

10. Desarrollar un programa para cargar una matriz de NxM con números aleatorios reales con dos decimales, luego hallar el máximo (mínimo) valor de esa matriz.

EXPLICACION DEL PROGRAMA: Este programa llena una matriz con números generados aleatoriamente (reales) y luego halla y despliega los máximos y mínimos valores.

```

//Realizado por Ignacio Salgado Faller
import java.io.*;
import java.math.*;
import java.text.DecimalFormat;
public class a58
{
    public static void main (String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
        double[][]mat=new double[100][100];
        int n,m;
        DecimalFormat df = new DecimalFormat("0.00");
        System.out.println("Numero de filas: ");
        n=Integer.parseInt(in.readLine());
        System.out.println("Numero de columnas: ");
        m=Integer.parseInt(in.readLine());
        double a=0;
        double b=200;
        for (int i=0;i<n;i++)
        {for (int u=0;u<m;u++)
        {mat[i][u]=(double)(Math.random()*89+10);

```

```

        if (mat[i][u]>a)
        {a=mat[i][u];}
        if (mat[i][u]<b)
        {b=mat[i][u];}
    }}
    System.out.println("La matriz es: ");
    for (int i=0;i<n;i++)
    {System.out.println("");
        for (int u=0;u<m;u++)
        {System.out.print(df.format(mat[i][u])+" ");}}
    System.out.println("");
    System.out.println("El mayor es: "+df.format(a));
    System.out.println("El menor es: "+df.format(b));
    }}

```

11. Desarrollar un programa para cargar una matriz de NxM con números aleatorios reales con dos decimales, luego formar un vector fila(columna) cuyos elementos sean iguales a la sumatoria de los elementos de las columnas (filas) de la matriz.

EXPLICACION DEL PROGRAMA: Primeramente se determina el tamaño de la matriz. Debemos considerar que el numero de columnas será igual al tamaño del vector, es por eso, que en //crea vector se puede utilizar la misma variable j para llenar los datos en vector. De ese modo solo se utiliza un sumador(sum) y el resultado se guarda en el vector en la posición j.

```

//Programa realizado por Andres Prudencio R.
import java.io.*;
import java.math.*;
import java.text.DecimalFormat;
public class a59
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in=new BufferedReader(new InputStreamReader(System.in));

        float[][]a=new float[100][100];
        int n,m;

        DecimalFormat df = new DecimalFormat("0.00");
        System.out.print("Ingresar el tamaño de la matriz(n x m).n= ");
        n=Integer.parseInt(in.readLine());
        //System.out.println();
        System.out.print("m= ");
        m=Integer.parseInt(in.readLine());
        float[]vf=new float[m];
    }
}

```

```

        for (int i=0;i<n;i++)
            {for(int j=0;j<m;j++)
            {a[i][j]=(float)(Math.random()*99+0);}}
        for (int i=0;i<n;i++)
            {System.out.println();
            for(int j=0;j<m;j++)
            {System.out.print(df.format(a[i][j])+" ");}}
//crea vector
float sum=0;
for (int j=0;j<m;j++)
    {for(int i=0;i<n;i++)
    {sum=sum+a[i][j];}
    vf[j]=sum;
    sum=0;
    }
System.out.println();

for (int o=0;m>o;o++)
System.out.print(df.format(vf[o])+" ");
System.out.print(" <==Vector Fila");
}}

```

12. Desarrollar un programa para cargar una matriz de NxM con números aleatorios reales con dos decimales, luego llevar los elementos de la matriz por filas (columnas) a un vector.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite cargar una matriz n*m de forma aleatoria con números reales con dos decimales exactamente y luego trasladarla a un vector todos sus elementos.

```

// Programa realizado por Sergio W. Ossio Marin
import java.text.DecimalFormat;
import java.io.*;
public class A_60
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        DecimalFormat df = new DecimalFormat ("0.00");
        double [][] a = new double [100] [100];
        double [] vector = new double [100];
        double k, f, c;
        System.out.println ("Ingrese el numero de filas");
        f = Integer.parseInt (in.readLine ());
        System.out.println ("Ingrese el numero de columnas");
        c = Integer.parseInt (in.readLine ());
        k = Double.valueOf (df.format (Math.random () * 100)).doubleValue ();
    }
}

```



```

for (int i = 0 ; i < f ; i++)
{
    for (int j = 0 ; j < c ; j++)
    {
        k = Double.valueOf (df.format (Math.random () * 100)).doubleValue ();
        a [i] [j] = k;
    }
}
for (int i = 0 ; i < f ; i++)
{
    for (int j = 0 ; j < c ; j++)
    {
        System.out.print (a [i] [j] + " ");
    }
    System.out.println ();
}
System.out.println ();
int h = 0;
for (int i = 0 ; i < f ; i++)
{
    for (int j = 0 ; j < c ; j++)
    {
        vector [h] = a [i] [j];
        h++;
    }
}
for (int j = 0 ; j < f * c ; j++)
{
    System.out.print (vector [j] + " ");
}
}
}

```

13. Desarrollar un programa para cargar un vector de K elementos con números aleatorios reales con dos decimales, luego llevar los elementos del vector a una matriz por filas, donde $K=N \times M$.

EXPLICACION DEL PROGRAMA: el programa primero llena un vector del tamaño ingresado por el usuario con valores aleatorios y lo convierte en una matriz del tamaño que ingresa el usuario.

```

//Programa realizado por Freddy Valda
import java.text.DecimalFormat;
import java.io.*;
public class a61
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    DecimalFormat df = new DecimalFormat("0.00");

```

```

double[][]a=new double[100][100];
double []kk=new double[100];
int k,n,m,sum=1;
System.out.println("INGRESE EL TAMANO DEL VECTOR");
k =Integer.parseInt(in.readLine());
for(int y=1;y<=k;y++)
{kk[y]=Double.valueOf(df.format(Math.random()*100)).doubleValue();

}
System.out.println("INGRESE EL NUMERO DE FILAS N, RECUERDE K=N*M");
n =Integer.parseInt(in.readLine());
System.out.println("INGRESE EL NUMERO DE COLUMNAS M RECUERDE K=N*M");
m =Integer.parseInt(in.readLine());

for(int f=1;f<=n;f++)
{
for(int c=1;c<=m;c++)
{a[f][c]=kk[sum];
sum++;
}}

System.out.println(" LA MATRIZ ES: ");
for(int q=1;q<=n;q++)
{
for(int w=1;w<=m;w++)
{System.out.print(" "+a[q][w]);
}
System.out.println("");
}
}
}
}

```

14. Realice un procedimiento que dado un arreglo A de $(n*(n+1))/2$ elementos almacene sus elementos en una matriz de $n \times n$ como muestra en el ejemplo:

Si $n=4$ y el arreglo fuese

12 24 15 13 16 9 1 2 0 3

La matriz debe quedar así:

12	0	0	0
24	15	0	0
13	16	9	0
1	2	0	3

EXPLICACION DEL PROGRAMA: Este programa permite al usuario ingresar los datos de un vector, los cuales son guardados posteriormente en una matriz de tamaño $n \times n$ de la forma en la que se muestra arriba.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a62
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int[]a=new int[100];
int cont=0,fl=0;
int [][]mat=new int[100][100];
System.out.println("Introduzca el orden de la matriz: ");
int n=Integer.parseInt(in.readLine());
System.out.println("Introduzca los "+(n*(n+1))/2+" datos del vector: ");
for(int i=0;i<(n*(n+1))/2;i++)
{a[i]=Integer.parseInt(in.readLine());}
for (int j=0;j<n;j++)
{for (int k=0;k<n;k++)
{if (j>=k)
{mat[j][k]=a[cont];
cont++;}
else
{mat[j][k]=fl;}}
}
System.out.println("Su matriz es: ");
for (int j=0;j<n;j++)
{System.out.println("");
for (int k=0;k<n;k++)
{System.out.print(mat[j][k]+" " );}}
}
}
```

15. Realizar un programa para que reciba, desde teclado, una matriz NxN de números enteros, luego calcule y escriba la suma de todas las filas y todas las columnas.

EXPLICACION DEL PROGRAMA: Se hizo uso de sumadores y cada resultado se guardo en una posición específica de la misma matriz dependiendo de la columna o fila en el vector. Posteriormente se procedio a mostrar los resultados.

```
//Realizado por Andres Prudencio R.
import java.io.*;
public class a63
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
System.out.println("Ingrese el orden de la matriz");
```

```

int n=Integer.parseInt(in.readLine());
System.out.println("Ingrese los "+((n)*(n))+" datos");
n=n+1;//n=n+1 porque se requiere una matriz de n+1 x n+1 para que en las ultimas
columnas y filas se realice las sumas
//para una matriz de orden n
int[][]a=new int[n+1][n+1];
int p,i,j,d=n-1;
//lee datos
for(i=0;i<n-1;i++)
{ for (j=0;j<n-1;j++)
{
a[i][j]=Integer.parseInt(in.readLine());
}
}
//realiza las sumas de filas y columnas
int sum=0,sumc=0;
System.out.println("");
for (p=0;p<n-1;p++)
{sum=0;
sumc=0;
for(i=0;i<n-1;i++)
{
sum=sum+a[i][p] ;
sumc=sumc+a[p][i];
}

a[p][d]=sum;
a[d][p]=sumc;
}

int sum1=0,sumi=0;

for(i=0;i<n-1;i++)
for (j=0;j<n-1;j++)

//muestra resultados
System.out.print("Matriz:");

for(i=0;i<n-1;i++)
{System.out.println("");
for (j=0;j<n-1;j++)
{if (j==n-1)
System.out.print(" = "+a[i][j]+" ");
else
System.out.print(a[i][j]+" ");
}
}
System.out.println("");
System.out.println("-----");

System.out.println("Suma de filas");
for (j=0;n-1>j;j++)
{System.out.print(a[n-1][j]+" ");
System.out.println("");}

```

```
System.out.println("");
System.out.println("Suma de columnas");
for (j=0;n-1>j;j++)
{System.out.print(a[j][n-1]+" ");}
System.out.println("");
}}
```

16. Escribir un programa que acepte por teclado una matriz de M orden NxN y un arreglo V de orden N que devuelva el número de veces que V aparece como fila de M.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite verificar si un vector de tamaño n se repite y cuantas veces se repite si así fuera en una matriz de orden n*m.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_64
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        int matriz [] [] = new int [20] [20];
        int [] vector = new int [20];
        int n;
        System.out.println ("Introduzca el tama*o de la matriz y el vector");
        n = Integer.parseInt (in.readLine ());
        for (int i = 0 ; i < n ; i++)
        {
            System.out.println ("Introduzca un valor");
            vector [i] = Integer.parseInt (in.readLine ());
        }
        System.out.println ("El vector es");

        for (int i = 0 ; i < n ; i++)
        {
            System.out.print (vector [i] + " ");
        }
        System.out.println ();
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                System.out.println ("Introduzca un valor");
                matriz [i] [j] = Integer.parseInt (in.readLine ());
            }
        }
        System.out.println ("La matriz es");
        for (int i = 0 ; i < n ; i++)
```

```

    {
        for (int j = 0 ; j < n ; j++)
        {
            System.out.print (matriz [i] [j] + " ");
        }
        System.out.println ();
    }
    int c, d = 0;
    for (int i = 0 ; i < n ; i++)
    {
        c = 0;
        for (int j = 0 ; j < n ; j++)
        {
            if (matriz [i] [j] == vector [j])
            {
                c++;
            }
        }
        if (c == n)
        {
            d++;
        }
    }
    System.out.println ("El vector se repite " + d + " veces");
}

```

17. Escriba un programa con una función ES_DIAG que reciba como parámetro una matriz M de orden NxN que devuelva TRUE si M es una matriz diagonal.

EXPLICACION DEL PROGRAMA: el programa recibe primero una matriz cuadrada y mediante una función booleana retorna true si es diagonal, si no retorna false.

```

//Programa realizado por Freddy Valda
import java.io.*;
public class a65
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int[][]a=new int[100][100];
    int n;
    System.out.println("INGRESE EL TAMANO DE LA MATRIZ (n*n)");
    n =Integer.parseInt(in.readLine());

    System.out.println("INGRESE LA MATRIZ");
    for(int f=1;f<=n;f++)
    {System.out.println("INGRESE LA FILA "+f);
    for(int c=1;c<=n;c++)

```

```
{
a[f][c]=Integer.parseInt(in.readLine());
}}

System.out.println( " LA MATRIZ INGRESADA ES: ");
for(int q=1;q<=n;q++)
{
for(int w=1;w<=n;w++)
{System.out.print(" "+a[q][w]);
}
System.out.println("");
}
if(ES_DIAG(a,n)==true)
System.out.println( "LA MATRIZ ES DIAGONAL ");
else
System.out.println( "LA MATRIZ NO ES DIAGONAL ");
}

static boolean ES_DIAG (int a[],int n)
{
int aux=0;
for(int q=1;q<=n;q++)
{
for(int w=1;w<=n;w++)
{if(q!=w)
if(a[q][w]!=0)
aux=aux+1;
}
}
if(aux==0)
return true;
else
return false;
}
}
```

18. Realice una función que dada una matriz de tamaño $M \times M$, un número que indique una de sus diagonales y un valor entero, devuelva si todos los elementos de la diagonal son iguales a dicho valor. Suponga que la diagonal principal es la número cero, la inferior a esta es -1 y así sucesivamente en orden decreciente; y las diagonales superiores a la principal siguen un orden ascendente a partir de 1. Utilice esta función para hacer un programa que calcule si la matriz es nula y si es identidad.

EXPLICACION DEL PROGRAMA: Este programa nos permite comprobar, a través de un menú de opciones, las diferentes instrucciones del problema.

La diagonal principal es la cero, la siguiente (en orden ascendente) es la 1, etc. Y nos permite ver si la matriz ingresada por el usuario es la identidad, la nula, o ninguna, además de decirnos si todos los elementos de una diagonal son iguales.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a66a
{
public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int [][]mat=new int[100][100];
int opc=4,dd,dig;
System.out.println("Introduzca el orden de la matriz: ");
int n=Integer.parseInt(in.readLine());
System.out.println("Introduzca los datos de la matriz: ");
for (int j=0;j<n;j++)
{for (int k=0;k<n;k++)
{mat[j][k]=Integer.parseInt(in.readLine());}
}
System.out.println("La matriz ingresada es: ");
for (int j=0;j<n;j++)
{System.out.println("");
for (int k=0;k<n;k++)
{System.out.print(mat[j][k]+" ");}}
System.out.println("");
while (opc>3||opc<1)
{System.out.println("1. Verificar si los elementos de la diagonal son iguales: ");
System.out.println("2. Verificar si la matriz es nula: ");
System.out.println("3. Verificar si la matriz es la matriz identidad: ");
System.out.println("Seleccione su opcion: ");
opc=Integer.parseInt(in.readLine());}
switch(opc)
{case 1: System.out.println("Introduzca la diagonal y el digito a verificar: ");
dd=Integer.parseInt(in.readLine());
dig=Integer.parseInt(in.readLine());
diagig(mat,dd,dig,n);break;
case 2: matnul(mat,n);break;
case 3: matide(mat,n);break;
}
}
public static void diagig(int y[],int g, int u,int kj)
{int aux,aux2=0,conti=0,g1=0;
if (g>=0)
{if(g==(kj-1))
{conti++;}
else
{g1=g;
aux=u;
while (g<kj)
{if (aux==y[aux2][g])
```



```
{conti++;}
aux2++;
g++;
}}
if (conti==(kj-g1))
{System.out.println("Todos los datos de la diagonal son iguales al valor");}
else
{System.out.println("No son todos iguales al valor");}
}
else
{g=-g;
g1=g;
aux=u;
while (g<kj)
{if (aux==y[g][aux2])
{conti++;}
aux2++;
g++;
}
if (conti==(kj-g1))
{System.out.println("Todos los datos de la diagonal son iguales al valor");}
else
{System.out.println("No son todos iguales al valor");}
}
}
public static void matnul(int y[],int gg)
{int dab=gg-1,conti=0,aux=0,aux2=0,g1=0;
for (int ii=0;ii<gg;ii++)
{aux2=0;
if(ii==0)
{conti++;}
else
{g1=ii;
while (g1<gg)
{if (aux==y[aux2][g1])
{conti++;}
aux2++;
g1++;
}}}
for (int uu=0;uu<dab;uu++)
{g1=uu;
aux2=0;
while (g1<gg)
{if (aux==y[g1][aux2])
{conti++;}
aux2++;
g1++;
}}
if (conti==(gg*gg))
{System.out.println("La matriz es nula");}
else
{System.out.println("La matriz no es nula");}
}
public static void matide(int y[],int gg)
{int dab=gg-1,conti=0,aux=0,aux2=0,g1=0,diago=0,diago1=1;
```

```

for (int ii=0;ii<gg;ii++)
{aux2=0;
if(y[ii][ii]==diago1)
{diago++;}
if(ii==0)
{conti++;}
else
{g1=ii;
while (g1<gg)
{if (aux==y[aux2][g1])
{conti++;}
aux2++;
g1++;
}}}
for (int uu=0;uu<dab;uu++)
{g1=uu;
aux2=0;
while (g1<gg)
{if (aux==y[g1][aux2])
{conti++;}
aux2++;
g1++;
}}
if (conti==((gg*gg)-gg)&&diago==gg)
{System.out.println("La matriz es la identidad");}
else
{System.out.println("La matriz no es la identidad");}
}
}

```

19. Hacer un programa que lea 25 valores enteros en una tabla de 5 por 5, y que después muestre la tabla y las sumas de cada fila y de cada columna. Procura que la salida sea clara, no te limites a los números obtenidos.

EXPLICACION DEL PROGRAMA: El programa utiliza una matriz de 6x6 en vez de una de 5 x 5 para los 25 datos, porque en las ultimas columnas y filas se guardan los resultados de las sumas. Si se quisiera, se podría aumentar el rango en los ciclos para mostrar la matriz, y los resultados se mostrarían en pantalla junto a sus respectivas columnas y filas. De todos modos se opto por mostrar los resultados de forma separada.

```

//Realizado por Andres Prudencio R.
import java.io.*;
public class a66b
{

```

```

public static void main(String args[])throws IOException
{BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
int n=6;//6 porque se requiere una matriz de 6x6 para que en las ultimas columnas y
filas se realice las sumas
//para 25 datos, sera sqrt(25)+1
System.out.println("Ingrese los "+((n-1)*(n-1))+ " datos");
//int n=Integer.parseInt(in.readLine());

int[][]a=new int[n+1][n+1];
int p,i,j,d=n-1;
//lee datos
for(i=0;i<n-1;i++)
{ for (j=0;j<n-1;j++)
{
a[i][j]=Integer.parseInt(in.readLine());
}
}
//realiza las sumas de filas y columnas
int sum=0,sumc=0;
System.out.println("");
for (p=0;p<n-1;p++)
{sum=0;
sumc=0;
for(i=0;i<n-1;i++)
{
sum=sum+a[i][p] ;
sumc=sumc+a[p][i];
}

a[p][d]=sum;
a[d][p]=sumc;
}

//muestra resultados
System.out.print("Matriz:");

for(i=0;i<n-1;i++)
{System.out.println("");
for (j=0;j<n-1;j++)
{if (j==n-1)
System.out.print(" = "+a[i][j]+" ");
else
System.out.print(a[i][j]+" ");
}
}
System.out.println("");
System.out.println("-----");

System.out.println("Suma de filas");
for (j=0;n-1>j;j++)
{System.out.print(a[n-1][j]+" ");
System.out.println("");}
System.out.println("");
System.out.println("Suma de columnas");

```

```

for (j=0;n-1>j;j++)
{System.out.print(a[j][n-1]+" ");}
System.out.println("");
}

```

Aplicación de Matrices – Calculos Matriciales

20. Realizar un P00 para trabajar con matrices, sobrecargado operadores y funciones(si es posible o simulando estas) donde se realice:

- a) Lectura de la matriz
- b) Listado de la matriz
- c) Ordenar todos los elementos de la matriz
- d) Ordenar la fila X de la matriz
- e) Ordenar la columna Y de la matriz
- f) Obtener la traza de la matriz
- g) Obtener el rango de la matriz.
- h) Obtener el elemento máximo de la matriz
- i) Suma de dos matrices
- i) Multiplicación de matrices.
- k) Obtener la matriz identidad en un constructor.
- 1) Multiplicar una matriz por un escalar.

EXPLICACION DEL PROGRAMA: el programa calcula todas las anteriores opciones, para ello despliega un menú de opciones y realiza cada calculo utilizando la sentencia switch para cada caso.

```

//Programa realizado por Freddy Valda
import java.io.*;
public class bb3
{
    public bb3(int a[],int n)
    {for(int i=1;i<=n;i++)
    {for(int j=1;j<=n;j++)
    {if(i==j)
    a[i][j]=1;
    else
    a[i][j]=0;
    }}}

    public static void main(String args[])throws IOException
    {BufferedReader in;
    in = new BufferedReader (new InputStreamReader (System.in));
    int[][]a=new int[100][100];
    int[][]b=new int[100][100];
    int[][]ca=new int[100][100];
    int f=0,c=0,resp,aux,x,y,suma=0;

    do
    {

```

```

System.out.println(" Menu de opciones: ");
System.out.println(" 1. LECTURA DE LA MATRIZ ");
System.out.println(" 2. LISTADO DE LA MATRIZ ");
System.out.println(" 3. ORDENAR TODOS LOS ELEMENTOS DE LA MATRIZ ");
System.out.println(" 4. ORDENAR LA FILA X DE LA MATRIZ ");
System.out.println(" 5. ORDENAR LA COLUMNA Y DE LA MATRIZ ");
System.out.println(" 6. OBTENER LA TRAZA DE LA MATRIZ");
System.out.println(" 7. OBTENER EL RANGO DE LA MATRIZ ");
System.out.println(" 8. OBTENER EL ELEMENTO MAXIMO DE LA MATRIZ ");
System.out.println(" 9. SUMA DE DOS MATRICES ");
System.out.println(" 10. MULTIPLICACION DE MATRICES ");
System.out.println(" 11. OBTENER LA MATRIZ IDENTIDAD EN UN CONSTRUCTOR
");
System.out.println(" 12. MULTIPLICAR UNA MATRIZ POR UN ESCALAR ");
System.out.println(" 13. SALIR  ");
System.out.println(" Ingrese la opcion: ");
resp =Integer.parseInt(in.readLine());

switch (resp)
{case 1:
System.out.println("Ingrese las filas y columnas");
f =Integer.parseInt(in.readLine());
c =Integer.parseInt(in.readLine());
for(int i=1;i<=f;i++)
{System.out.println("Ingrese la fila: "+i);
for(int jq=1;jq<=c;jq++)
{a[i][jq]=Integer.parseInt(in.readLine());
}}
break;
case 2:
for(int qq=1;qq<=f;qq++)
{
for(int ww=1;ww<=c;ww++)
{System.out.print(" "+a[qq][ww]);
}
System.out.println("");
}
break;
case 3:
for(int ii=1;ii<=f;ii++)
{for(int j=1;j<=c;j++)
{
for(int q=1;q<=f;q++){
for(int w=1;w<=c;w++)
{if (a[ii][j]<a[q][w])
{aux=a[ii][j];
a[ii][j]=a[q][w];
a[q][w]=aux;}
}}}}
break;
case 4:
System.out.println("Ingrese la fila a ordenar ");
x =Integer.parseInt(in.readLine());
for(int ja=1;ja<=c;ja++)
{

```

```
for(int qa=1;qa<=c;qa++)
{if (a[x][ja]<a[x][qa])
{aux=a[x][ja];
a[x][ja]=a[x][qa];
a[x][qa]=aux;}
}
}

break;
case 5:
System.out.println("Ingrese la columna a ordenar ");
y=Integer.parseInt(in.readLine());
for(int ia=1;ia<=f;ia++)
{
for(int wa=1;wa<=f;wa++)
{if (a[ia][y]<a[wa][y])
{aux=a[ia][y];
a[ia][y]=a[wa][y];
a[wa][y]=aux;}
}
}

break;
case 6:
suma=0;
System.out.print("La traza es: ");
for(int ib=1;ib<=f;ib++)
{for(int jb=1;jb<=c;jb++)
{if(ib==jb)
suma=suma+a[ib][jb];
}}
System.out.print(" "+suma);

break;
case 7:
int suma0=0;
for(int id=1;id<=f;id++)
{for(int jd=1;jd<=c;jd++)
{if(id==jd)
suma0=suma0+a[id][jd];
}}
int suma1=0,rango;
for(int ic=1;ic<=f;ic++)
{for(int jc=1;jc<=c;jc++)
{if((ic+jc)==(f+1))
suma1=suma1+a[ic][jc];
}}
rango=suma0-suma1;
System.out.print("EL RANGO ES: "+rango);

break;
case 8:
int max;
max=a[0][0];
for(int ie=1;ie<=f;ie++)
```

```

{for(int je=1;je<=c;je++)
{if(a[ie][je]>max)
max=a[ie][je];
}}
System.out.println("VALOR MAXIMO: "+max);
break;
case 9:
int fi,co;
System.out.println("Ingrese las filas y columnas de las 2 matrices(deben ser iguales
para que se puedan sumar)");
fi =Integer.parseInt(in.readLine());
co =Integer.parseInt(in.readLine());
System.out.println("MATRIZ 1");
for(int ig=1;ig<=fi;ig++)
{System.out.println("Ingrese la fila: "+ig);
for(int jg=1;jg<=co;jg++)
{a[ig][jg]=Integer.parseInt(in.readLine());
}}
System.out.println("MATRIZ 2");
for(int ih=1;ih<=fi;ih++)
{System.out.println("Ingrese la fila: "+ih);
for(int jh=1;jh<=co;jh++)
{b[ih][jh]=Integer.parseInt(in.readLine());
}}

for(int iq=1;iq<=fi;iq++)
{
for(int jq=1;jq<=co;jq++)
{ca[iq][jq]=a[iq][jq]+b[iq][jq];
}}

for(int qqa=1;qqa<=fi;qqa++)
{
for(int wwa=1;wwa<=co;wwa++)
{System.out.print(" "+ca[qqa][wwa]);
}
System.out.println("");
}

break;
case 10:
int fi1,fi2,co1,co2;
System.out.println("Ingrese las filas y columnas de las matriz 1 (recuerde que las
columnas de la primera deben iguales a las filas de la segunda)");
fi1 =Integer.parseInt(in.readLine());
co1 =Integer.parseInt(in.readLine());
System.out.println("Ingrese las filas y columnas de las matriz 2 (recuerde que las
columnas de la primera deben iguales a las filas de la segunda)");
fi2 =Integer.parseInt(in.readLine());
co2 =Integer.parseInt(in.readLine());
System.out.println("MATRIZ 1");
for(int iga=1;iga<=fi1;iga++)
{System.out.println("Ingrese la fila: "+iga);
for(int jga=1;jga<=co1;jga++)
{a[iga][jga]=Integer.parseInt(in.readLine());
}
}

```

```

}}
System.out.println("MATRIZ 2");
for(int iha=1;iha<=fi2;iha++)
{System.out.println("Ingrese la fila: "+iha);
for(int jha=1;jha<=co2;jha++)
{b[iha][jha]=Integer.parseInt(in.readLine());
}}
int suum=0;
for(int i1=1;i1<=fi1;i1++)
{
for(int j1=1;j1<=co2;j1++)
{
suum=0;
for(int kk=1;kk<=co1;kk++)
{suum=(suum+a[i1][kk]*b[kk][j1]);}
ca[i1][j1]=suum;
}}

for(int q1=1;q1<=fi1;q1++)
{
for(int w1=1;w1<=co2;w1++)
{System.out.print(" "+ca[q1][w1]);
}
System.out.println("");
}

break;
case 11:
int [][]xx=new int[100][100];
int dd;
System.out.println("Ingrese el tamaño de la identidad");
dd=Integer.parseInt(in.readLine());
bb3 identidad=new bb3(xx,dd);
for(int ai=1;ai<=dd;ai++)
{for(int aj=1;aj<=dd;aj++)
{System.out.print(" "+xx[ai][aj]);
}
System.out.println("");}

break;
case 12:
System.out.println("Ingrese el escalar a multiplicar");
int esc=Integer.parseInt(in.readLine());
for(int qz=1;qz<=f;qz++)
{
for(int wz=1;wz<=c;wz++)
{a[qz][wz]=(a[qz][wz]*esc);
}
}

for(int qc=1;qc<=f;qc++)
{
for(int wc=1;wc<=c;wc++)
{System.out.print(" "+a[qc][wc]);
}
}

```



```

System.out.println("");
}

break;
case 13:
break;}

}
while(resp != 13);
System.out.println("FIN DEL PROGRAMA");
}
}

```

21. Realizar un programa en Java para ordenar un vector de tamaño n por el método Shell, n y los elementos son introducidos por el usuario, suponer $1 < n < 25$.

```

import java.io.*;
/**
 * @author Hernan Payrumani M.
 */
public class Shell3 {
    public static void main(String[] args) throws IOException {
        BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
        int[] a = new int[25];
        System.out.println("Ingresa la dimension del vector [1,25];");
        int n = Integer.parseInt(in.readLine());
        for(int i=0; i<n; i++){
            System.out.print("a["+i+1+"]: ");
            a[i]=Integer.parseInt(in.readLine());
        }
        shellSort(a,n);
        for(int i=0;i<n;i++)
            System.out.print(a[i]+" ");
    }
    public static void shellSort(int a[],int n){
        for(int gap=n/2;gap>0;
            gap=gap==2?1:(int)(gap/2.2)){
            for(int i=gap;i<n;i++){
                int temp=a[i];
                int j;
                for(j=i;j>=gap&&temp<a[j-gap];
                    j-=gap){
                    a[j]=a[j-gap];
                }
                a[j]=temp;
            }
        }
    }
}

```

22. Realizar un programa en Java que por medio de un menú el usuario elija el método que desee para ordenar un vector de tamaño n que será introducido por el mismo

```
import java.io.*;
/**
 * @author Hernan Payrumani M.
 */
public class SortMenu {
    public static void main(String[] args) throws IOException {
        BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
        int[] a = new int[50];
        try{
            System.out.println("\t\t\tMenu");
            System.out.println("\t\t1.Metodo de la Burbuja");
            System.out.println("\t\t2.Metodo de Insercion");
            System.out.println("\t\t3.Metodo de Seleccion");
            System.out.println("\t\t4.Metodo Shell");
            int op = Integer.parseInt(in.readLine());
            switch(op){
            case 1:
                System.out.println("Ingrese la dimension del vetor [1,50];");
                int n = Integer.parseInt(in.readLine());
                for(int i=0; i<n; i++){
                    System.out.print("a["+(i+1)+"]: ");
                    a[i]=Integer.parseInt(in.readLine());
                }
                buble(a,n);
                mostrar(a,n);
                break;
            case 2:
                System.out.println("Ingrese la dimension del vetor [1,50];");
                n = Integer.parseInt(in.readLine());
                for(int i=0; i<n; i++){
                    System.out.print("a["+(i+1)+"]: ");
                    a[i]=Integer.parseInt(in.readLine());
                }
                insercion(a,n);
                mostrar(a,n);
                break;
            case 3:
                System.out.println("Ingrese la dimension del vetor [1,50];");
                n = Integer.parseInt(in.readLine());
                for(int i=0; i<n; i++){
                    System.out.print("a["+(i+1)+"]: ");
                    a[i]=Integer.parseInt(in.readLine());
                }
                seleccion(a,n);
                mostrar(a,n);
                break;
            case 4:
                System.out.println("Ingrese la dimension del vetor [1,50];");
```

```

        n = Integer.parseInt(in.readLine());
        for(int i=0; i<n; i++){
            System.out.print("a["+(i+1)+"]: ");
            a[i]=Integer.parseInt(in.readLine());
        }
        shell(a,n);
        mostrar(a,n);
        break;
    }
}
catch( Exception ex ){
    System.out.println("\n\n");
    System.out.println( "--o--o--o--o--o--o--o--o--");
    System.out.println( "- Error! Numero incorrecto -");
    System.out.println( "--o--o-A-o--o--o--o--o--o--");
    System.out.println("\n\n");
}
}
public static void bubble(int a[],int n){
    for(int i=1;i<n;i++){
        for(int j=0;j<n-i;j++){
            if(a[j]>a[j+1]){
                int aux=a[j];
                a[j]=a[j+1];
                a[j+1]=aux;
            }
        }
    }
}
public static void insercion(int a[],int n){
    for(int i=1;i<n;i++){
        int aux=a[i];
        int j=i-1;
        while(j>=0&&aux<a[j]){
            a[j+1]=a[j];
            j--;
        }
        a[j+1]=aux;
    }
}
public static void seleccion(int a[],int n){
    for(int i=0;i<n;i++) {
        int menor=a[i];
        int pos=i;
        for(int j=i+1;j< n;j++){
            if(a[j]<menor){
                menor=a[j];
                pos=j;
            }
        }
        a[pos]=a[i];
        a[i]=menor;
    }
}
public static void shell(int a[],int n){
    for(int gap=n/2;gap>0;

```

```

        gap=gap==2?1:(int)(gap/2.2)){
            for(int i=gap;i<n;i++){
                int temp=a[i];
                int j;
                for(j=i;j>=gap&&temp<a[j-gap];
                    j-=gap){
                    a[j]=a[j-gap];
                }
                a[j]=temp;
            }
        }
    }
}

public static void mostrar(int a[],int n){
    for(int i=0;i<n;i++)
        System.out.print(a[i]+" ");
}
}

```

Aplicación de matrices – Inversion de Matrices por Faddeeva

22. Nuevamente, Programa para encontrar la matriz inversa con el método de Faddeeva. (Preguntar el método al Ing. Guillermo Espinoza, profesor de Cs. Exactas).

EXPLICACION DEL PROBLEMA: Este programa nos permite encontrar la inversa de una matriz $n \times n$ por el metodo de faddeeva, el cual es un metodo muy dinamico ya que este implica todo un procedimiento el cual intentaremos explicar brevemente: En primer lugar leer una matriz A de tamaño $n \times n$ luego esta matriz es copiada a una matriz A_n , calculamos la traza de esta matriz , esta traza será multiplicadora de la matriz identidad. Este producto lo usaremos para que pueda ser restada con la matriz original A. Esta resta a la vez será guardada en otra matriz que será multiplicada por la matriz original, este ultimo producto también será guardado en una ultima matriz la cual pasara por todo el ciclo hasta ahora mencionado.

```

// Programa realizado por Sergio W. Ossio Marin y Andres Prudencio R.
import java.io.*;
public class faddeeva
{
    public static void main (String args []) throws IOException
    {

```

```

BufferedReader in;
in = new BufferedReader (new InputStreamReader (System.in));
double [][] id = new double [100] [100];
double [][] A = new double [100] [100];
double [][] B = new double [100] [100];
double [][] AI = new double [100] [100];
double [][] AUX = new double [100] [100];

double [][] An = new double [100] [100];
double [][] BB = new double [100] [100];
int n, f, c;
double qq=0;
System.out.println ("Ingrese el orden de la matriz cuadrada");
n = Integer.parseInt (in.readLine ());
//lectura de la matriz
iden (id, n);
for (int i = 0 ; i < n ; i++)
{
    System.out.println ("Ingrese la fila: " + (i + 1));
    for (int j = 0 ; j < n ; j++)
    {
        A [i] [j] = Integer.parseInt (in.readLine ());
    }
}
iden (id, n);
copy (A, An, n);
int cq=0;
int h = 1;
for (int i = 1 ; i <= n ; i++)
{
    q = traza (An, n);
    q = q / h;
    cq++;
    if (cq == n)
    {qq = 1 / q;};
    //System.out.println();
    //System.out.print(q);
    //System.out.println();

    cxp (id, q, n, AUX);
    OB_B (An, B, n, AUX);
    if (i == (n-1))
    {copy (B, BB, n);}

    MULT (B, n, A, An);
    //System.out.println();
    //show(An,n);
    // System.out.println();
    h++;
}

cxp(BB,qq,n,AI);
System.out.println ("Matriz Inversa: ");
show(AI,n);

```

```

    }
    public static void iden (double id [] [], int n)
    {
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                if (i == j)
                    id [i] [j] = 1;
                else
                    id [i] [j] = 0;
            }
        }
    }
    public static double traza (double l [] [], int n)
    {
        double sum = 0;
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                if (i == j)
                    sum += l [i] [j];
            }
        }
        return sum;
    }

    public static void cxp (double O [] [], double q, int n, double P [] [])
    {
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                P [i] [j] = q * O [i] [j];
            }
        }
    }
    public static void OB_B (double A [] [], double O [] [], int n, double U [] [])
    {
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                O [i] [j] = A [i] [j] - U [i] [j];
            }
        }
    }
    public static void MULT (double O [] [], int n, double A [] [], double y [] [])
    {
        double suum = 0;
        for (int i = 0 ; i < n ; i++)
        {
            for (int j = 0 ; j < n ; j++)
            {
                suum = 0;
                for (int k = 0 ; k < n ; k++)
                {
                    suum = (suum + O [i] [k] * A [k] [j]);
                }
                y [i] [j] = suum;
            }
        }
    }

```

```

    }}
    public static void show (double O [] [], int n)
    {
        for (int q = 0 ; q < n ; q++)
        {
            for (int w = 0 ; w < n ; w++)
            {

                System.out.print (" " + O [q] [w]);
            }
            System.out.println ("");
        }
    }
    public static void copy (double O [] [], double An [] [], int n)
    {
        for (int q = 0 ; q < n ; q++)
        {
            for (int w = 0 ; w < n ; w++)
            {
                An [q] [w] = O [q][w];
            }
        }
    }
}

```

PROBLEMAS PROPUESTOS

1. Dadas 2 matrices de orden $n \times n$, obtener la suma de las mismas
2. Generar la siguiente matriz de orden $n \times n$:

```

11111
10001
10001
10001
10001
11111

```

3. Generar la siguiente matriz:

```

1 2 4 7
3 5 8 11
6 9 12 14
10 13 15 16

```

4. Generar la siguiente matriz de orden $n \times n$:

```

*1111
1*001
10*01
100*1

```

1111*

5. Realice un programa para invertir una matriz mediante un método distinto al de Faddevva

6. Realice un programa para generar la siguiente matriz de orden $n \times n$:

```
00000
01110
01010
01110
00000
```

7. Realice un programa que genere una matriz en la que se guarden los datos de las ventas de 6 vendedores de la siguiente manera:

	Cantidad Vendida	Vendedor 1	Cantidad vendida
Vendedor 2			
Toyota			
Peugeot			
Mazda			
Ford			
Mitsubishi			

Luego mostrar la cantidad de ventas por marca de cada vendedor y mostrar al que mas haya vendido y al que menos haya vendido

8. Crear un programa que genere aleatoriamente una elección presidencial de la siguiente manera:

	MNR	ADN	MIR	MAS
La Paz					
Cochabamba					
Santa Cruz					
Beni					
Tarija					

.....

Una vez llenada la tabla, mostrar que partido tuvo mas votos en cada departamento y que partido gana a nivel nacional y con que porcentaje de los votos.

9. Realice un programa para generar la siguiente matriz de orden $n \times n$:

```
00n00
00n00
nnnnn
00n00
00n00
```

Reemplazando las n's por el orden de la matriz

10. Realice un programa que reemplace todos los lugares en los que la fila y la columna sumadas sumen el orden de la matriz por el orden de la matriz. Se debe tomar como posición inicial.

Recursividad

Los siguientes programas fueron resueltos haciendo uso de funciones recursivas. Las funciones recursivas son aquellas que se invocan a si mismas.

1. Escribir una función recursiva que halle la suma de los primeros "n" números naturales.

EXPLICACION DEL PROGRAMA: Este programa simplemente realiza una función recursiva tipo void para hallar la suma requerida.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a69
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    int sum=0;
    System.out.println("Introduzca ula cantidad de naturales: ");
    int n=Integer.parseInt(in.readLine());
    enter(n,sum);
    }
    public static void enter(int y, int w)
    {if (y<0)
    {System.out.println("La suma es: "+w)}
    else
    {w=w+y;
    enter(y-1,w);}
    }
}
```

2. Escribir una función recursiva que devuelva la cantidad de dígitos de un número entero.

EXPLICACION DEL PROGRAMA: El programa usa una función recursiva donde la condición básica es que si el la parte entera de un numero dividido entre 10 es 0 entonces debe retornar 1, haciendo que la misma función actue como contador de digitos al sumarse esta con un uno.

```
//Realizado por Andres Prudencio R.
import java.io.*;
public class a70
{
    public static void main (String args[])throws IOException
    {

        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        System.out.println("Ingrese un numero entero positivo: ");
        int c=0;
        int d=Integer.parseInt(in.readLine());
        if (d>=0)
            System.out.println("Cantidad de digitos: "+cdig(d,c));
        else
            System.out.println("Debe ingresar un numero positivo o nulo");
        }

        public static int cdig(int d,int c)
        {
            if((d/10)==0)
            {
                return 1;}
            else
            {d=d/10;
            return 1+cdig(d,c);
            }
        }
    }
}
```

3. Escribir una función recursiva que permita calcular el factorial de un número entero. ¿Conviene realmente la utilización de la versión recursiva, por sobre la iterativa? Justificar debidamente la respuesta?.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite calcular el factorial de un número de una forma elegante “Recursiva”.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
```

```
public class A_71
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Introduzca el numero el cual quiere calcular su factorial");
    int n=Integer.parseInt(in.readLine());
    System.out.println(+fact(n));
    }
    static int fact(int n)
    {if ( n== 0 ) {
        return 1;}
    else
    {return n*fact(n-1);}}}
```

4. Escribir una función recursiva que calcule w^k mediante multiplicaciones sucesivas, siendo k un número natural.

EXPLICACION DEL PROGRAMA: el programa calcula un numero elevado a una potencia utilizando una función recursiva. Primero pide el numero y la potencia y finalmente despliega el resultado llamando a la función `pot(a,b)`.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a72
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Introduzca un numero y la potencia (NATURALES)");
    int a=Integer.parseInt(in.readLine());
    int b=Integer.parseInt(in.readLine());
    System.out.println(a+" ^ "+b+" = "+pot(a,b));
    }
    static int pot(int nmero,int potencia){
        if ( potencia == 0 ) {
            return 1;
        } else {
            return nmero*pot(nmero,potencia-1);
        }
    }
}
```

5. Escribir un procedimiento recursivo que calcule $z*v$, mediante sumas sucesivas, con z, v enteros.

EXPLICACION DEL PROGRAMA: Aquí se utiliza una función recursiva tipo void para calcular $z*v$ mediante sumas sucesivas.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a73
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    int sum=0;
    System.out.println("Introduzca el primer numero: ");
    int z=Integer.parseInt(in.readLine());
    System.out.println("Introduzca el segundo numero: ");
    int v=Integer.parseInt(in.readLine());
    multi(z,v,sum);
    }
    public static void multi(int y, int w, int u)
    {if (y<=0)
    {System.out.println("La multiplicacion es: "+u);}
    else
    {u=u+w;
    multi (y-1,w,u);}
    }
}
```

6. Proponer un procedimiento recursivo tal que dado un arreglo de números reales permita calcular el mínimo elemento del vector y su posición.

EXPLICACIÓN DEL PROGRAMA: El programa usa una función recursiva la cual imprime el elemento menor de un vector y su posición usando la variable c como control de posición para analizar el vector. La condición básica es que c sea -1 , para imprimir los resultados. La debido a que c esta en constante disminución (de $n-1$ a -1) se va guardando la posición del elemento minimo en la variable p para después imprimirla. El análisis del vector es el mismo: asignar como elemento minimo el elemento 1 del vector y compararlo con este. Si algún elemento es menor al 1, entonces aquel es asignado como minimo.

```
//Realizado por Andres Prudencio R.
//AP
```

```

import java.io.*;
import java.math.*;
import java.text.DecimalFormat;
public class a74
{

    public static void main (String args[])throws IOException
    {
        DecimalFormat df = new DecimalFormat("0.0");
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        System.out.println("Ingrese el tamaño del vector: ");

        int n=Integer.parseInt(in.readLine());
        float []v=new float[100];

        int c=n-1;
        for (int i=0;n>i;i++)
            v[i]=(float)(Math.random()*10);
        float min=v[0];
        System.out.println();
        System.out.println("Vector: ");
        int p=0;
        for (int i=0;n>i;i++)
            System.out.print(df.format(v[i])+" ");

        System.out.println();
        vmin(v,c,min,p);
    }

    public static float vmin(float v[],int c,float min,int p)
    {DecimalFormat df = new DecimalFormat("0.0");
    if (c==-1)
        {System.out.println("Elemento minimo "+df.format(min)+" Posicion: "+(p+1));
        return 0; }
    else
        {if(min>v[c])
            {min=v[c];
            p=c;}
        return vmin(v,c-1,min,p);
    }}}

```

7. Proponer un procedimiento recursivo tal que dado un arreglo de números reales permita calcular el promedio de sus elementos.

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite calcular el promedio de una serie de números guardados en un arreglo de una forma elegante "Recursiva".

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_75
{
    public static void main(String args[])throws IOException

    {BufferedReader in;
    int n;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Ingrese el tamaño del arreglo");
    n=Integer.parseInt(in.readLine());
    int []a=new int[100];
    int sum,medio;
    System.out.println("Ingrese los "+n+" elementos");
    for(int y=1;y<=n;y++)
    {a[y]=Integer.parseInt(in.readLine());}

    System.out.println("ARREGLO ");
    for(int yy=1;yy<=n;yy++)
    {System.out.print(" "+a[yy]);}
    System.out.println("");

    System.out.println("");
    System.out.println("La suma de los elementos es: "+suma(a,n));
    System.out.println("El valor medio de los elementos es: "+(suma(a,n)/n));
    }
    static int suma(int a[],int n)
    {if(n == 1)
    return a[n];
    else
    {return a[n]+suma(a,n-1);}
    }}
}
```

8. Escribir una función recursiva que dado un número entero positivo calcule su imagen especular. Por ejemplo: $f(345)=543$.

EXPLICACION DEL PROGRAMA: el programa calcula un número que es el número ingresado invertido mediante una función recursiva con retorno, utilizando el método pow de la clase math

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a76
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Introduzca el número");
}
```

```

int n=Integer.parseInt(in.readLine());

int c,s=0;
c=dig(n);

System.out.println("La imagen especlar es "+inv(n,c-1,s));
}

static int dig(int n)
{if(n==0)
{return 0;}
else
{return (dig(n/10)+1);}}

static int inv(int n,int c,int s)
{if(n==0)
{return 0;}
else
{
return ((n%10)*(int)Math.pow(10,c)+inv(n/10,c-1,s));
}
}
}

```

9. Escribir un procedimiento recursivo que imprima el contenido de las posiciones pares de un arreglo de enteros.

EXPLICACION DEL PROGRAMA: Este programa, a través de una función recursiva void, lee primero un vector de n términos y hace que un contador que va de 0 hasta n sea sometido a una prueba para ver si es par (contador%2). Si la posición es par, el numero es mostrado

```

//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a77
{
public static void main(String args[])throws IOException
{BufferedReader in;
in=new BufferedReader(new InputStreamReader(System.in));
int sum=0;
int []v=new int[100];
System.out.println("Ingrese la cantidad de datos: ");
int n=Integer.parseInt(in.readLine());
System.out.println("Introduzca los datos del arreglo: ");
for(int i=0;i<n;i++)
{v[i]=Integer.parseInt(in.readLine());}
System.out.println("Las posiciones pares son: ");
posi(n,v,sum);
}
}

```



```

}
public static void posi(int y, int w[], int u)
{if (u>=y)
 {y=0;}
 else
 {System.out.print(w[u]+" ");
 posi (y,w,u+2);}
 }
}

```

10. Calcular $C(n,k)$, los coeficientes binomiales recursivamente, siendo:

- a) $C(n,0)=C(n,n)=1$ si $n \geq 0$
- b) $C(n,k)=C(n-1,k)+C(n-1,k-1)$ si $n > k > 0$

EXPLICACION DEL PROGRAMA: Basicamente se introdujo las condiciones del problema a una función recursiva con la condición básica a) y con el retorno alternativo de b).

```

//Realizado por Andres Prudencio R.
import java.io.*;
public class a78
{
    public static void main (String arg []) throws IOException
    {
        BufferedReader in = new BufferedReader (new InputStreamReader
(System.in));
        float n, k;
        System.out.println ("Introduzca el valor de n y k: C(n,k)  n>=0, n>k>0");
        n = Integer.parseInt (in.readLine ());
        k = Integer.parseInt (in.readLine ());

        while (0 > n || n < k)
        {
            System.out.println ("Introduzca valores para n y k.  n>=0, n>k>0");
            n = Integer.parseInt (in.readLine ());
            k = Integer.parseInt (in.readLine ());
        }
        System.out.println(cc(n,k));
    }
    public static float cc (float n, float k)
    {
        float c = 0;
        if (n == k || k == 0)
            return 1;
        else
        {
            return cc (n - 1, k) + cc (n - 1, k - 1);
        }
    }
}

```

11. Diseñar un procedimiento recursivo para determinar el Máximo Común Divisor (mcd) de dos números enteros positivos m, n usando la siguiente indicación:

$$mcd(m,n) = \begin{cases} n & \text{si } n \leq m \text{ y } m \bmod n = 0 \\ mcd(n,m) & \text{si } m < n \\ mcd(n, m \bmod n) & \text{en otro caso} \end{cases}$$

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite calcular el MCD de dos números ingresados por teclado de una forma elegante "Recursiva".

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_79
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    int n,m;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Ingresa los dos numeros");
    n=Integer.parseInt(in.readLine());
    m=Integer.parseInt(in.readLine());
    System.out.println(+mcd(n,m));
    }
    static int mcd(int n,int m)
    {if ( n==m )
    {return n;}
    else
    if(n>m)
    return mcd(n-m,m);
    else
    return mcd(n,m-n);
    }}
```

12. Escribir una función recursiva para sumar los productos de los elementos correspondientes de dos vector A y B de n elementos cada uno y que se han llenado con números aleatorios.

EXPLICACION DEL PROGRAMA: el programa calcula la suma de los productos de 2 vectores. Se llenaron los vectores con la función random de la clase math. Finalmente se despliegan todos los vectores y el vector de la suma de sus productos.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a80
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Ingrese el tamaño de los arreglos");
    int n=Integer.parseInt(in.readLine());
    int []a=new int[100];
    int []b=new int[100];

    for(int y=1;y<=n;y++)
    {a[y]=(int)(Math.random()*100);}
    for(int f=1;f<=n;f++)
    {b[f]=(int)(Math.random()*100);}
    }

    System.out.println("ARREGLO 1");
    for(int yy=1;yy<=n;yy++)
    {System.out.print(" "+a[yy]);}
    System.out.println("");
    System.out.println("ARREGLO 2");
    for(int ff=1;ff<=n;ff++)
    {System.out.print(" "+b[ff]);
    }
    System.out.println("");
    System.out.println("La suma de los productos es: "+prod(a,b,n));
    }
    static int prod(int a[],int b[],int n)
    {if(n == 1)
    return a[n]*b[n];
    else
    {return a[n]*b[n]+prod(a,b,n-1);}
    }
}
```

13. Escribir un programa que utilice funciones recursivas para calcular la suma siguiente:

$$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \dots \text{donde } 0 \leq x \leq 1.$$

Sumar términos mientras estos sean mayores a 10^{-8} . El valor de x se introduce por teclado.

EXPLICACION: Este programa utiliza el mismo principio del ejercicio 18, pero se utiliza una función recursiva void para realizar los cálculos en vez de la sentencia while.

```
//programa realizado por Ignacio Salgado Faller
import java.io.*;
public class a81
{
    public static void main(String arg[])throws IOException
    {BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    double x;
    System.out.println("Introduzca el valor de x entre 0 y 1:");
    x = (Float.valueOf (in.readLine ())).floatValue ();
    double term=0,conta=1,aux=1,sig=1;
    sumat(x,aux,sig,term,conta);
    }

    public static void sumat(double o,double p,double q, double r, double s)
    {if (p<=0.00000001)
    {System.out.println("La sumatoria es: "+p);}
    else
    {r=(q*(Math.pow(s,o)/facto(s)))+r;
    s=s+2;
    q=-q;
    p=Math.pow(s,o)/facto(s);
    sumat(o,p,q,r,s);
    }
    }

    public static double facto(double xx)
    {if (xx==0)
    {xx=1;
    return xx;}
    else
    {double sum=0;
    for (double i=1;i<xx;i++)
    {sum=i*xx+sum;}
    xx=sum;
    return xx;}
    }}
}
```

14. Escribir un programa que utilice funciones recursivas para calcular la suma siguiente:

$$1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \dots \text{donde } 0 \leq x \leq 1.$$

EXPLICACION DEL PROGRAMA: el procedimiento es exactamente el mismo que se uso en ejercicio 81, excepto que el contador (conta) debe inicializarse en 0, de modo que en la función sumat s aumente de 2 en dos, asi, devolviendo los denominadores pares. En este programa se añadió la validacion de datos.

```
//Realizado por Ignacio Salgado
//Modificaciones: Andres Prudencio R.
import java.io.*;
public class a82
{
    public static void main(String arg[])throws IOException
    {BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
    double x;
    System.out.println("Introduzca el valor de x entre 0 y 1:");
    x = (Float.valueOf (in.readLine ())).floatValue ();
    while (0>x||x>1)
    {System.out.println("Introduzca el valor de x entre 0 y 1:");
    x = (Float.valueOf (in.readLine ())).floatValue ();}

    double term=0,conta=0,aux=1,sig=1;
    sumat(x,aux,sig,term,conta);}

    public static void sumat(double o,double p,double q, double r, double s)
    {if (p<=0.00000001)
    {System.out.println("La sumatoria es: "+p);}
    else
    {r=(q*(Math.pow(s,o)/facto(s)))+r;
    s=s+2;
    q=-q;
    p=Math.pow(s,o)/facto(s);
    sumat(o,p,q,r,s);
    }
    }

    public static double facto(double xx)
    {if (xx==0)
    {xx=1;
    return xx;}
    else
    {double sum=0;
    for (double i=1;i<xx;i++)
    {sum=i*xx+sum;}
    xx=sum;
    return xx;}
    }
}
```

15. Escribir un función recursiva para sumar n elementos de la siguiente serie:

$$1^x + 2^x + 3^x + + n^x .$$

EXPLICACIÓN DEL PROGRAMA: Este programa nos permite calcular la suma de n elementos de la serie ya antes mencionada de una forma elegante "recursiva".

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
public class A_83
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    double n,x;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Ingresa el numero de terminos");
    n=Integer.parseInt(in.readLine());
    System.out.println("Ingresa el tamao de la potencia");
    x=Integer.parseInt(in.readLine());
    System.out.println(+serie(n,x));
    }
    public static double serie(double n,double x)
    {if(n == 1)
    return 1;
    else
    {return Math.pow(n,x)+serie(n-1,x);
    }
    }}
}
```

16. Hacer un programa con funciones recursivas que lea diez valores enteros en un array desde el teclado y calcule y muestre: la suma, el valor medio, el mayor y el menor.

EXPLICACION DEL PROGRAMA: el programa utiliza funciones recursivas para calcular la suma, valor medio, mayor y menor valor de un arreglo de tamaño 10 introducido por teclado. Cada cálculo tiene su propia función recursiva.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a84
{
    public static void main(String args[])throws IOException
```

```
{BufferedReader in;
in=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Ingrese el tamaño de los arreglos");
int []a=new int[100];
int n=10;
int max,min,sum,medio;
System.out.println("Ingrese los 10 elementos");
for(int y=1;y<=10;y++)
{a[y]=Integer.parseInt(in.readLine());}

System.out.println("ARREGLO ");
for(int yy=1;yy<=10;yy++)
{System.out.print(" "+a[yy]);}
System.out.println("");

System.out.println("");
System.out.println("La suma de los elementos es: "+suma(a,n));
System.out.println("El valor medio de los elementos es: "+(suma(a,n)/10));
System.out.println("El mayor valor es: "+ may(a,n));
System.out.println("El menor valor es: "+ men(a,n));
}

static int suma(int a[],int n)
{if(n == 1)
return a[n];
else
{return a[n]+suma(a,n-1);}
}

static int may(int a[],int n)
{if(n == 1)
return a[n];
else
{if(a[n]>may(a,n-1))
return a[n] ;
else
return may(a,n-1);
}}

static int men(int a[],int n)
{if(n == 1)
return a[n];
else
{if(a[n]<men(a,n-1))
return a[n] ;
else
return men(a,n-1);
}}}
```

17. Mediante una función recursiva llenar un vector con números aleatorios, luego separar de la misma a los elementos pares, y conformar con ellos otra lista.

EXPLICACION DEL PROGRAMA: Este programa utiliza una función recursiva void para llenar un vector con números aleatorios, luego guarda los pares en un vector y los impares en otro y los muestra.

```
/programa realizado por Ignacio Salgado Faller
import java.io.*;
import java.math.*;
import java.text.DecimalFormat;
public class a85
{
    public static void main (String args[])throws IOException
    {
        { BufferedReader in;
        in=new BufferedReader(new InputStreamReader(System.in));
        int[]vector=new int[100];
        int[]vec=new int[100];
        DecimalFormat df = new DecimalFormat("0.00");
        int a=vector[0];
        int b=200;
        System.out.println("Cuantos datos desea generar?");
        int j=Integer.parseInt(in.readLine());
        int cont=0;
        int conta=0;
        recur(vector,vec,j,cont,conta);
        }

        public static void recur(int vv[],int vv2[],int jj,int co,int co2)
        {if (co>=jj)
        {muestra(vv,vv2,jj,co2);}
        else
        {vv[co]=(int)(Math.random()*100);
        if ((vv[co]%2)==0)
        {vv2[co2]=vv[co];
        co2++;
        co++;
        recur(vv,vv2,jj,co,co2);}
        else
        {co++;
        recur(vv,vv2,jj,co,co2);}
        }
        }

        public static void muestra(int uu[],int yy[],int ww,int zz)
        {System.out.println("Su vector original es: ");
        for (int uy=0;uy<ww;uy++)
        {System.out.print(uu[uy]+" ");}
        System.out.println("");
        System.out.println("Su vector de pares es: ");
```



```

        if(zz==0)
        {System.out.println("No existen pares en el vector original");}
        else
        {for (int ux=0;ux<zz;ux++)
        {System.out.print(yy[ux]+" ");}}
        }
    }
}

```

18. Escribir una función recursiva para generar un vector con números aleatorios, luego ubicar el máximo y mínimo elemento que haya en el vector, sin ordenar los elementos del mismo.

EXPLICACION DEL PROGRAMA: Se uso la misma metodología del ejercicio 74, excepto a ciertas modificaciones: se aumenta un if a la función recursiva para gragar en la variable max el elemento máximo y la eliminación de las variables que indicaban la posición del vector.

```

//Realizado por Andres Prudencio
import java.io.*;
public class a86
{
    public static void main (String args[])throws IOException
    {
        BufferedReader in;
        in = new BufferedReader (new InputStreamReader (System.in));
        System.out.println("Ingrese el tamaño del vector: ");

        int n=Integer.parseInt(in.readLine());
        int []v=new int[100];

        int c=n-1;
        for (int i=0;n>i;i++)
        v[i]=(int)(Math.random()*100+1);
        int min=v[0],max=v[0];
        System.out.println();
        System.out.println("Vector: ");
        int p=0;
        for (int i=0;n>i;i++)
        System.out.print(v[i]+" ");

        System.out.println();
        vminmax(v,c,min,max);
    }

    public static int vminmax(int v[],int c,int min,int max)
    {
        if (c== -1)
        {System.out.println("Elemento maximo: "+max+" Elemento minimo: "+min);

```

```
return 0; }
else
{
    if(min>v[c])
        {min=v[c];}
    if(max<v[c])
        {max=v[c];}
    return vminmax(v,c-1,min,max);
}
}
}
```

19. Mediante funciones recursivas generar dos vectores con números aleatorios, ordenarlos y luego multiplicar los elementos correspondientes de los dos vectores y almacenar en un tercer vector. Mostrar en pantalla los tres vectores.

EXPLICACION DEL PROGRAMA: Este programa nos permite poder dos vectores con números aleatorios mediante un función random luego mediante otra función ordenamos estos vectores y finalmente obtenemos un tercer vector el cual contiene el producto vectorial de ambos vectores oredenados y a su ves también ordena este ultimo vector.

```
// Programa realizado por Sergio W. Ossio Marin
import java.io.*;
import java.math.*;
public class A_87
{
    public static void main (String args []) throws IOException
    {
        BufferedReader in;
        int n;
        in = new BufferedReader (new InputStreamReader (System.in));
        System.out.println ("Ingresa el tamaño de los vectores");
        n = Integer.parseInt (in.readLine ());
        int [] vector1 = new int [20];
        int [] vector2 = new int [20];
        int [] vector3 = new int [20];
        for (int i = 0 ; i < n ; i++)
        {
            vector1 [i] = (int) (Math.random () * 98 + 1);
        }
        System.out.println ("El vector 1 es: ");
        for (int o = 0 ; o < n ; o++)
        {
```

```
System.out.print (vector1 [o] + " ");
}
for (int i = 0 ; i < 20 ; i++)
{
    vector2 [i] = (int) (Math.random () * 98 + 1);
}
System.out.println ();
System.out.println ("El vector 2 es: ");
for (int o = 0 ; o < n ; o++)
{
    System.out.print (vector2 [o] + " ");
}
ordena (n, vector1);
ordena (n, vector2);
mostrar (n, vector1);
mostrar (n, vector2);
mul (n, vector3, vector1, vector2);
}
public static void ordena (int n, int vectork [])
{
    int aux;
    for (int i = 0 ; i < n ; ++i)
        for (int j = 0 ; j < n ; ++j)
        {
            if (vectork [i] < vectork [j])
            {
                aux = vectork [i];
                vectork [i] = vectork [j];
                vectork [j] = aux;
            }
        }
}
public static void mostrar (int n, int vectork [])
{
    System.out.println ();
    System.out.println ("El vector ordenado es: ");
    for (int o = 0 ; o < n ; o++)
    {
        System.out.print (vectork [o] + " ");
    }
}
public static void mul (int n, int vectork [], int vectore [], int vectord [])
{
    System.out.println ();

    for (int o = 0 ; o < n ; o++)
    {
        vectork [o] = vectore [o] * vectord [o];
    }
    System.out.println ();
    System.out.println ("El vector ordenado del producto vectorial entre los dos
vectores ordenados es: ");
    for (int o = 0 ; o < n ; o++)
    {
        System.out.print (vectork [o] + " ");
    }
}
```

20. Escribir un procedimiento recursivo que calcule el producto escalar de dos vectores que recibe como parámetros.

EXPLICACION DEL PROGRAMA: el programa calcula el producto escalar de 2 vectores mediante una función recursiva que tiene como parámetros los 2 vectores, para ello en la función recursiva se van sumando cada producto de cada elemento de los 2 vectores.

```
//Programa realizado por Freddy Valda
import java.io.*;
public class a88
{
    public static void main(String args[])throws IOException
    {BufferedReader in;
    in=new BufferedReader(new InputStreamReader(System.in));
    System.out.println("Ingrese el tamaño de los arreglos");
    int n=Integer.parseInt(in.readLine());
    int []a=new int[100];
    int []b=new int[100];

    System.out.println("INGRESE EL ARREGLO 1");
    for(int y=1;y<=n;y++)
    {a[y]=Integer.parseInt(in.readLine());}
    System.out.println("INGRESE EL ARREGLO 2");
    for(int f=1;f<=n;f++)
    {b[f]=Integer.parseInt(in.readLine());}
    }

    System.out.println("ARREGLO 1");
    for(int yy=1;yy<=n;yy++)
    {System.out.print(" "+a[yy]);}
    System.out.println("");
    System.out.println("ARREGLO 2");
    for(int ff=1;ff<=n;ff++)
    {System.out.print(" "+b[ff]);}
    }
    System.out.println("");
    System.out.println("El producto escalar es: "+prod(a,b,n));
    }
    static int prod(int a[],int b[],int n)
    {if(n == 1)
    return a[n]*b[n];
    else
    {return a[n]*b[n]+prod(a,b,n-1);}
    }
}
```

PROBLEMAS PROPUESTOS

1. Mediante un programa recursivo determine si un numero n, ingresado por teclado es primo o no.
2. Diseñe e implemente un método recursivo que nos permita obtener el determinante de una matriz cuadrada de dimensión n
3. Implemente, de forma recursiva, una función que le dé la vuelta a una cadena de caracteres
4. Mediante un programa recursivo calcule la división de 2 números mediante restas sucesivas
5. Diseña un algoritmo recursivo que lea una secuencia de caracteres de longitud arbitraria terminada en un punto, e imprima la suma de todos los dígitos junto con la sucesión en orden inverso de entrada.
6. Escribe un algoritmo recursivo que ordene un array de la siguiente forma:
 - a) Sea k el índice del elemento mitad del array.
 - b) Ordena los elementos hasta a[k], incluyéndolo.
 - c) Ordena los elementos siguientes.
 - d) Mezcla los dos subarrays en un único array ordenado.Este método de clasificación interna se denomina "Mergesort".
7. Diseña un algoritmo recursivo que, dado un array de enteros, devuelva el k-ésimo elemento más pequeño del array,
es decir, si el array contiene los elementos (4,7,3,6,8,1,9,2) y k=5, el algoritmo debería devolver 6.
8. Implementa el algoritmo de búsqueda binaria de forma recursiva.
9. Mediante un programa recursivo, determinar si un numero es capicúa o no.
10. Generar el triangulo de Pascal para n filas utilizando funciones recursivas

Ejercicios con Gráficos

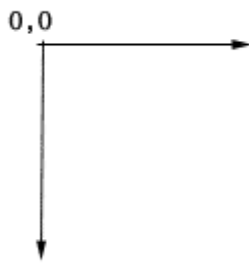
Vamos a presentar métodos para dibujar varias figuras geométricas. Como estos métodos funcionan solamente cuando son invocados por una instancia válida de la clase **Graphics**, su ámbito de aplicación se restringe a los componentes que se utilicen en los métodos *paint()* y *update()*. Normalmente los métodos de dibujo de primitivas gráficas funcionan por pares: un método pinta la figura normal y el otro pinta la figura rellena.

Ejercicio 1 Crear un applet que dibuje una línea recta que empiece en la coordenada (10, 10) del applet y llega a la coordenada (200, 10).

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*; // Librería grafica.
import java.applet.Applet; // Librería para applet

public class Linea extends Applet
// Se indica que la clase Línea pertenece a Applet.
{
    public void paint( Graphics g )
    // Se declara que g es una instancia de la clase Graphics.
    {
        g.drawLine( 10, 10, 200, 10 ); // Método para trazar líneas
    }
}
```

El applet anterior lo que ase es dibujar una línea desde la coordenada (10, 10) de la pantalla y traza una línea hasta la coordenada (200, 10). El primer numero de cada coordenada nos indica el espacio que abra entre el comienzo izquierdo del applet con la coordenada indicada y el segundo numero nos indica el espacio entre el punto y la parte superior del applet como vemos en el grafico.



Cada flecha empieza en la coordenada (0, 0) del applet la que se desplaza hacia la izquierda tendrá una segunda coordenada de la forma (x, 0), donde "x" será el tamaño de la flecha.

El método que se usa en este caso es **g.drawLine(10, 10, 200, 10)**, donde "g" es una instancia de la clase **Graphics**.

Ejercicio 2 Crear un applet que tenga una línea recta que llegue de la coordenada (0, 0) a la coordenada que se encuentre a la mitad del applet.

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;
public class linea2 extends Applet
{int mancho,malto;

    public void init() {
        Dimension d = size(); // se crea una instancia de Dimensión.
        mancho = d.width;    // devuelve el ancho del applet.
        malto = d.height;    // devuelve la altura del applet.
        repaint( );
    }
    public void paint( Graphics g )
    {
        g.drawLine( 0,0,mancho/2,malto/2 ); // Método para trazar líneas.
    }
}
```

En el anterior applet se crea una línea recta de tamaño variable de acuerdo al tamaño del applet, con ayuda de la instancia "d" se

encontró la altura y el ancho del applet de forma que dividiendo entre dos se pueda encontrar el centro del applet.

La clase que se declaro fue línea2 y se declara que pertenece a Applet

El método que se usa en este caso es **g.drawLine(0,0,mancho/2,malto/2)**, donde "g" es una instancia de la clase **Graphics**.

Ejercicio 3 Crear una equis que se intersécate en el centro del applet.

```
/ realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;
public class linea3 extends Applet
{int mancho,malto;

    public void init() {
        Dimension d = size(); // se crea una instancia de Dimensión.
        mancho = d.width;    // devuelve el ancho del applet.
        malto = d.height;    // devuelve la altura del applet.
        repaint( );
    }
    public void paint( Graphics g )
    {
        g.drawLine( 0,0,mancho-1,malto-1 ); // Método para trazar líneas
        g.drawLine( mancho-1,0,0,malto-1 ); // Método para trazar líneas
    }
}
```

Como se vio anteriormente se puede crear líneas de cualquier punto a otro de forma sencilla, como se ve en el ejercicio se puede utilizar el método **g.drawLine** mas de una vez. Además "g" es una instancia de la clase **Graphics** que se puede usar en mas métodos.

Ejercicio 4 Crear un applet que contenga un rectángulo con 200 píxeles de alto y 300 de ancho que su esquina superior izquierda este en la coordenada (50, 50).


```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;

public class Rectangulo extends Applet
{
    public void paint( Graphics g )
    {
        g.drawRect ( 50,50,300,200 ); // Método para trazar rectángulos
    }
}
```

Este ejercicio no es diferente a los que realizamos con líneas lo único que cambia es el método que se utiliza, en este caso se utiliza el método **g.drawRect(50,50,300,200)**, como en los otros ejercicios se declaro que "g" es una instancia de la clase Graphics, en el paréntesis se coloca primero la coordenada de la punta superior izquierda del rectángulo que en este caso seria (50, 50) luego se coloca el ancho y luego la altura en píxeles en este caso el ancho es 300 y la altura 200.

Ejercicio 5 Crear un applet que contenga a un rectángulo relleno con 200 píxeles de alto y 300 de ancho que su esquina superior izquierda este en la coordenada (50, 50).

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;

import java.awt.*;
import java.applet.Applet;

public class Rectangulo1 extends Applet
{
    public void paint( Graphics g )
    {
        g.fillRect ( 50,50,300,200 ); // Método para trazar rectángulos rellenos
    }
}
```

Este ejercicio es igual al anterior solo que este esta relleno, para crear rectángulos rellenos se utiliza el método **g.fillRect** y las coordenadas son iguales a **g.drawRect**.

Ejercicio 6 Crear un applet que contenga a un rectángulo relleno con 200 píxeles de alto y 300 de ancho que su esquina superior izquierda este en la coordenada (50, 50), además el color de su relleno debe ser aleatorio.

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;

public class Rectcol extends Applet { //Se declara la clase Rectcol.
    Color color;
    public void init(){
        color = new Color( aleatorio(255),aleatorio(255),aleatorio(255) );
        //Se declara un color aleatorio.
        repaint();
    }

    public void paint( Graphics g ) {
        g.setColor ( color );
        g.fillRect( 50,50,300,200 );
    }

    private int aleatorio( int rango )
        //Esta es la función que aleatorio
    {
        double retornoMath;
        retornoMath = Math.random();
        return( (int)( retornoMath * rango ) );
    }
}
```

En este ejercicio lo nuevo es el uso de un color y la forma de hacer números aleatorios, el color se obtiene con el método **g.setColor (color)** donde color es un vector con tres números que señalan que color se debe mostrar.

La función creada aleatorio solo devuelve números enteros aleatorios con ayuda del método **random()** que pertenece a la clase **Math**.

Ejercicio 7 Crear un applet que contenga un rectángulo como margen del applet y dentro el 5 rectángulos ubicados al azar con colores al azar

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;

public class Rec extends Applet
{
    int apAncho,apAlto;
    int x,y,ancho,alto;
    Color color;

    public void init() {
        Dimension d = size();
        apAncho = d.width;
        apAlto = d.height;
    }

    public void paint( Graphics g )
    {
        g.setColor( Color.black );
        g.drawRect( 0,0,apAncho-1,apAlto-1 );

        for( int i=0; i < 5; i++ )
        {
            x = aleatorio( apAncho );
            y = aleatorio( apAlto );
            ancho = aleatorio( apAncho - x );
            alto = aleatorio( apAlto - y );
            color = new Color( aleatorio(255),aleatorio(255),
                               aleatorio(255) );

            g.setColor( color );
            g.fillRect( x,y,ancho-1,alto-1 );
        }
    }

    private int aleatorio( int rango ) {
        double retornoMath;
        retornoMath = Math.random();
        return( (int)( retornoMath * rango ) );
    }
}
```

Como se ha visto anteriormente el margen se puede ubicar con el método **g.drawRect**, para crear 5 rectángulos en su interior se uso un **for** que va del 1 al 5 y gracias a la función aleatorio que devuelve números al azar podemos darles el tamaño y posición de forma aleatoria sin que se salgan del margen trazado.

Ejercicio 8 Crear la bandera de Bolivia en base a rectángulos de tamaño 300 de ancho y 200 de alto.

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;

public class Bandera extends Applet { //Se declara la clase Bandera.
    public void paint( Graphics g ) {
        g.setColor ( Color.red );      //Se declara color Rojo.
        g.fillRect( 50,50,300,65 );
        g.setColor ( Color.yellow );   //Se declara color Amarillo.
        g.fillRect( 50,115,300,65 );
        g.setColor ( Color.green );    //Se declara color verde.
        g.fillRect( 50,180,300,65 );
    }
}
```

Lo nuevo en este ejercicio es la forma de declarar colores específicos claro que estos deben estar en ingles.

Ejercicio 9 Crear un círculo sin relleno de 25 píxeles de radio.

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;
public class Redondo extends Applet
{
    public void paint( Graphics g )
    {
        g.drawOval (100, 50, 50, 50 );
        //Método para trazar redondos sin relleno
    }
}
```

En el anterior ejercicio se usa el método **g.drawOval** que sirve para crear círculos sin relleno lo que se encuentra en el paréntesis se entiende como (**derecha, arriba, ancho, alto**).

- **derecha** espacio entre el comienzo del applet con la derecha del círculo.
- **arriba** es el espacio entre el círculo con la parte superior del applet.
- **ancho** es el ancho del círculo u ovalo.
- **alto** es la altura del círculo u ovalo.

Como en el ejercicio nos pedían que sea de 25 de radio entonces se tendrá que poner 50 de ancho y 50 de alto.

Ejercicio 10 Crear un círculo relleno de 25 píxeles de radio.

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;
public class Redondo2 extends Applet
{
    public void paint( Graphics g )
    {
        g.fillOval (200, 100, 100, 100 );
        //Método para trazar redondos con relleno
    }
}
```

En el anterior ejercicio se usa el método **g.fillOval** que sirve para crear círculos con relleno, ya que no se declaró ningún color antes se usa el color negro para rellenar la figura, En el paréntesis se entiende como (**derecha, arriba, ancho, alto**) en este orden.

- **derecha** espacio entre el comienzo del applet con la derecha del círculo.

- **arriba** es el espacio entre el círculo con la parte superior del applet.
- **ancho** es el ancho del círculo u ovalo.
- **alto** es la altura del círculo u ovalo.

Como en el ejercicio nos pedían que sea de 25 de radio entonces se tendrá que poner 50 de ancho y 50 de alto.

Ejercicio 11 Crear un applet que contenga un par de ojos que fijen su mirada al cursor del ratón.

```
// realizado por Edwin Marcelo Guzman Buezo
import java.awt.*;
import java.applet.Applet;

public class Ojos extends Applet {
    int Mx = -1;
    int My = -1;
    int OjoR1;
    int OjoR2;
    int Ancho;
    int Alto;
    int Ojolzq;
    int OjoDch;
    int OjoPY;
    Color Pupila = Color.black;
    Color Iris = Color.green.brighter();
    Color Orbita = Color.white;
    Image Imagen;
    Graphics OjoCG;
    public void init() {
        setBackground( Color.darkGray );
        Dimension d = size();
        // Fijamos las variables que nos van a posicionar los ojos sobre el applet
        Ojolzq = d.width >> 2;
        OjoDch = Ojolzq * 3;
        OjoPY = d.height >> 2;
        OjoR2 = d.width >> 4;
        OjoR1 = d.width >> 5;
        Ancho = ( d.width >> 3 ) + OjoR1;
        Alto = Ancho >> 1;
    }
}
```

```

public void update( Graphics g ) {
    paint( g );
}
// Función auxiliar, para que no se
int swap( int i,int c ) {
// salgan los ojos de las orbitas
    if( i > c )
        i = c;
    else if( i < -c )
        i = -c;
    return( i );
}
// Pintamos el ojo sobre el applet
void pintaOjo( Graphics g,int x )
{
// Fijamos los desplazamientos, las nuevas posiciones de
//Referencia, en función de la posición del cursor del
// ratón, determinada por Mx y My
    int dx = x-Mx;
    int dy = OjoPY-My;
// Pintamos el ojo solamente bordeado es decir cerrado
// para cuando el ratón pase por encima.
    if( dx < Ancho && dx > -Ancho && dy < Alto && dy > -Alto )
    {
        g.setColor( getBackground() );
        g.fillOval( x-Ancho,OjoPY-Alto,Ancho << 1,Alto << 1 );
        g.setColor( getBackground().brighter() );
        g.drawOval( x-Ancho,OjoPY-Alto,Ancho << 1,Alto << 1 );
    }
    else
    {
        // Pintamos el hueco del ojo, por el que se moverá el iris
        g.setColor( Orbita );
        g.fillOval( x-Ancho,OjoPY-Alto,Ancho << 1,Alto << 1 );
        int y = OjoPY;
        dx = swap( dx >> 3,OjoR1 << 1 );
        dy = swap( dy >> 5,OjoR1 >> 1 );
        if( Mx >= 0 && My >= 0 )
        {
            x -= dx;
            y -= dy;
        }

        // Pintamos el iris, sobre el que se moverá la pupila
        g.setColor( Iris );
        g.fillOval( x-OjoR2,y-OjoR2,OjoR2 << 1,OjoR2 << 1 );
        if( Mx >= 0 && My >= 0 )
        {
            x -= ( dx >> 1 );

```

```
        y -= dy;
    }
    // Pintamos la pupila dentro del iris
    g.setColor( Pupila );
    g.fillOval( x-OjoR1,y-OjoR1,OjoR1 << 1,OjoR1 << 1 );
}

}

public void paint( Graphics g ) {
    Dimension d = size();
    // La primera vez que se llama a este método, todavía no
    // hay nada, así que creamos el soporte de los ojos
    if( Imagen == null || OjoCG == null )
    {
        Imagen = createImage( d.width,d.height >> 1 );
        OjoCG = Imagen.getGraphics();
    }








    // Pintamos los ojos
    OjoCG.setColor( getBackground() );
    OjoCG.fillRect( 0,0,d.width,d.height );
    pintaOjo( OjoCG,Ojolzq );
    pintaOjo( OjoCG,OjoDch );
    g.drawImage( Imagen,0,0,this );
} // Cuando movemos el curso dentro del applet, recibimos las
    // coordenadas y repintamos el ojo
public boolean mouseMove( Event evt,int x,int y ) {
    Mx = x;
    My = y;
    repaint();

    return true;
}
public boolean mouseExit( Event evt,int x,int y ) {
    Mx = My = -1;
    repaint();




    return true;
}
}
```

En este ejercicio se trató de explicar algunos movimientos que se pueden obtener con métodos parecidos a los ya vistos, las explicaciones se las encuentra en el código y las que no se explican están en ejercicios anteriores.

BIBLIOGRAFIA

-  JOYANES AGUILAR LUIS, FERNANDEZ AZUELA MATILDE, JAVA 2 MANUAL DE PROGRAMACION, McGraw-Hill, 2001
-  <http://www.monografias.com/trabajos/java/java.shtml>
-  <http://www.arrakis.es/~abelp/ApuntesJava/Introduccion.htm>
-  <http://www.di.ujaen.es/~mgarenas/java/general/indice.html>
-  Apuntes propios de clases.
-  <http://www.lcc.uma.es/~afdez/apuntes/elell/relacion1.pdf>
-  <http://www.itapizaco.edu.mx/paginas/JavaTut/froufe/parte2/>

Ejercicios Propuestos:

-  Ing. Orlando Rivera, Introduccion a la Programación, 2007
-  <http://usuarios.lycos.es/sanplaale/programacio/EjerciciosProgramacion.htm>
-  Los mismos autores.