EXPLICACIÓN DE LOS EJERCICIOS:

EJERCICIO 1: En este ejercicio creamos una clase estática número con un método mostrarValor que reciba un número como final y recibe un valor. Este modificador final lo que nos esta indicando es que ese último valor recibido no va a poder ser modificado. Es por esto que al intentar cambiarle el valor me genera un error de compilación. Para llamar al método desde el main hacemos una importación estática del método y con solo poner el nombre del método con su valor ya es suficiente.

EJERCICIO 2: En este ejercicio creamos un arreglo de cinco notas que lo inicializamos con llaves separados entre comas. Declaramos la suma como double para cuando calculemos el promedio. Creamos un arreglo y recorremos el arreglo con el bucle for y utilizamos el length para obtener la cantidad de elementos del arreglo.

Es recomendable utilizar el length en vez de poner 5 por si el tamaño del arreglo cambia en algún momento dado. Dentro de este for sumamos las notas y después fuera del for calculamos el promedio haciendo suma/notas.length y las mostramos por pantalla.

EJERCICIO 3: En este ejercicio creamos una matriz utilizando el operador new que sea 3*3 y declaramos un número que empiece en 1 ya que el ejercicio me pide llenar la matriz partiendo desde el 1 y como en un arreglo las posiciones se empiezan a contar desde la posición 0 es necesario declararlo. Vamos a necesitar declarar dos índices uno para las filas y otro para las columnas a través de un bucle for anidado que recorra cada uno de sus elementos. Dentro de él vamos a tener que sumarle 1 a cada posición que se recorra para de esta manera hacer que la matriz se llene con números del 1 al 9.Mostramos la matriz y en cuanto a la suma de las filas creamos de vuelta dos bucles for anidados pero en este utilice el método length para poder obtener la cantidad de elementos, declaramos un dato para la suma inicializada en cero, sumamos las filas y mostramos el resultado por pantalla.

EJERCICIO 4: En este ejercicio vamos a crear una ArrayList que para poder crearla vamos a necesitar importar la librería java.util.ArrayList. La clase ArrayList agrupa elementos como un arreglo (se puede utilizar para acceder a elementos, insertar o borrar en base a una posición). En este caso lo vamos a utilizar para agregar o insertar nombres a través del add. En cuanto al bucle for lo utilizamos para que me recorra todo el bucle y me muestre los nombres agregados, en este caso utilizamos un bucle for each ya que permite acceder a cada elemento del arreglo sin tener que hacer uso de índices específicos.

EJERCICIO 5: En este ejercicio importamos la libreria java.util.Scanner ya que le vamos a tener que pedir al usuario que ingrese dos cadenas. Una vez ingresadas las dos cadenas las vamos a tener que comparar a través de un if para el equals() y un if para el equalsIgnoreCase(). El método equals() es utilizado para comparar dos cadenas y tiene en cuenta las letras mayúsculas y minúsculas, en vez, el equalsIgnoreCase es utilizado para comparar dos cadenas y no tiene en cuenta las letras mayúsculas de las minúsculas. Y estos son los aspectos que va a tener en cuenta en cada uno de los if.

EJERCICIO 6: En este ejercicio importamos la librería java.util.Scanner para pedirle al usuario que ingrese una cadena por teclado. Para contar cuantas vocales tiene la cadena primero vamos convertir la cadena en minúscula a través del toLowerCase ya que cuando contemos las vocales estén todas las letras en el mismo formato y sean más fáciles de identificar. Declaramos un contador inicializado en cero y utilizamos un for para que recorrer cada uno de los caracteres de la cadena para poder encontrar las vocales. A la cantidad de caracteres lo obtenemos a través del length. Dentro de este for vamos a utilizar el método charAt(i) para de esta manera poder analizar carácter por carácter al recorrer la cadena. Luego colocamos un if para ver las vocales que puede tener la cadena luego de haber analizado cada uno de los caracteres. Una vez detectadas las vocales las vamos contando y mostramos el resultado por pantalla.

EJERCICIO 7: En este ejercicio importamos la librería java.util.Scanner para pedirle al usuario que ingrese una cadena por teclado. Una vez ingresada esta cadena vamos a reemplazar las letras ‘a’ por ‘@’ a través del método replace, este método es utilizado para reemplazar una parte de la cadena o toda la cadena por otra, que en este caso sería reemplazar las letras ‘a’ por ‘@’. En la primer parte de este método van a ir los valores del carácter a reemplazar y en el segundo el nuevo carácter que se le va asignar. Una vez realizado esto mostramos los resultados por pantalla.

EJERCICIO 8: En este ejercicio creamos una clase que contenga el método static palíndromo, al declararlo como static estoy diciendo que no hace falta crear un objeto desde el main para poder llamarlo. Para poder obtener un palíndromo primero vamos a tener que invertir la cadena utilizando el StringBuffer para poder manipular varias veces a una cadena, ya que con String una vez creada una cadena no se puede volver a modificar, luego vamos a utilizar el reverse() para invertir la cadena obtenida por StringBuilder y luego el toString() que convierte el StringBuilder obtenido de nuevo a String. Una vez obtenido esto hacemos cadena + invertida.substring(1) que esto lo que

va hacer va a ser unir la cadena original con la cadena invertida sin su primer carácter y de esta manera se obtiene el palíndromo. Mostramos los resultados por pantalla.

EJERCICIO 9: En este ejercicio creamos una clase saludar con dos métodos static, uno que salude sin parámetros y otro que muestre el nombre del usuario. Al declararlos como static estoy diciendo que no hace falta crear un objeto desde el main para poder llamarlos, sino que en el main se llaman poniendo el nombre de la clase y el método con sus párametros si es que tienen.

EJERCICIO 10:En este ejercicio declaramos la cadena "Hola Mundo". A la subcadena la vamos a obtener utilizando substring en la que se indica el índice inicial y el índice final de la parte de la cadena que queremos obtener. El substring es utilizado para obtener una subcadena de la original que se devuelve como otro objeto cadena. Para reemplazar "Hola" por "Adiós" se utiliza replace que este método es utilizado para reemplazar una parte de la cadena por otra en la que la primer parte tenemos que indicar lo que queremos reemplazar y en la segunda parte el valor por lo cual lo vamos a reemplazar y mostramos los resultados por pantalla.

EJERCICIO 11: En este ejercicio vamos a declarar una cadena y un número entero, luego realizamos la primer concatenación con el operador + que consiste en sumar la cadena con el número entero. Y luego realizamos la segunda concatenación con valueOf que consiste en convertir el número entero antes declarado en un valor de tipo String al utilizar el valueOf y realizamos la sumatoria entre cadena y este valor obtenido. El valueOf es utilizado para convertir datos de tipo primitivo a su representación en forma de cadena. Una vez realizado esto mostramos los resultados por pantalla.

EJERCICIO 12: En este ejercicio creamos una clase NumeroPar que contiene el método esPar que lo declaramos como boolean por retornar valores como true o false. A este método lo vamos a declarar como static y esto significa que no hace falta crear un objeto desde el main para poder inicializarlo. Dentro de este método si me retorna que al dividir el número en dos tiene resto cero decimos que es par, por el contrario es impar.

Dentro del main declaramos un número entero y llamamos a la clase junto con su método (debido a que el método es static) y utilizamos un if porque si el método me retorna true me va a decir que es par y si me retorna false me va a decir que es impar.