Tarea estructuras de control

Objetivos

- Utilizar la clase Scanner para el ingreso de datos por parte del usuario
- Aplicar las diferentes estructuras de control dentro un programa en Java

La clase Scanner

Scanner es una clase en el paquete java.util utilizada para obtener la entrada por teclado de los tipos primitivos como int, double etc. y también String.
Para utilizar la clase lo primero que hay que hacer es importarla.

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

Para crear un objeto de clase Scanner, normalmente pasamos el objeto predefinido System.in, que representa el flujo de entrada estándar.

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

Cuando se introducen caracteres por teclado, el objeto Scanner toma toda la cadena introducida y la divide en elementos llamados **tokens**. El carácter predeterminado que sirve de separador de tokens es el espacio en blanco. El delimitador puede cambiarse usando el método useDelimiter(String)

Para leer los diferentes valores que un usuario puede ingresar se tienen los siguientes métodos:

Método	**Descripción**
next()	Devuelve el siguiente token como un String
nextXyz()	Devuelve el siguiente token como un tipo básico. Xyz es el tipo. Por ejemplo, nextInt() para leer un entero, nextDouble para leer un double, etc.
nextLine()	Devuelve la línea entera como un String. Elimina el final \n del buffer

Al finalizar el uso del objeto Scanner, éste debe cerrarse usando el método **close()**

Ejemplo:

En el siguiente programa se solicita al usuario el ingreso de su nombre, edad y salario. Se utilizan tres métodos para recuperar los valores ingresados por el usuario.

```
import java.util.Scanner;
class EjemploScanner {
```

```
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Ingrese su nombre:");
    // ingreso de string
    String nombre = sc.nextLine();
    System.out.print("Ingrese su edad:");
    // ingreso de numero entero
    int edad = sc.nextInt();
    System.out.print("Ingrese su salario:");
    double salario = sc.nextDouble();
    sc.close();
    //mostsrar datos ingresados por el usuario
    System.out.println("Nombre: " + nombre);
    System.out.println("Edad: " + edad);
    System.out.println("Salario: " + salario);
  }
}
```

Posibles errores en el uso de Scanner

Si ingresa una entrada incorrecta (por ejemplo una cadena en un valor entero esperado), se produce un error y recibirá un mensaje de “InputMismatchException”

Por ejemplo si ingresa un texto cuando se solicita la edad, se produce la siguiente salida:

```
Ingrese su nombre:Juan Perez
Ingrese su edad:veinte
Exception in thread "main" java.util.InputMismatchException
        at java.util.Scanner.throwFor(Scanner.java:864)
        at java.util.Scanner.next(Scanner.java:1485)
        at java.util.Scanner.nextInt(Scanner.java:2117)
        at java.util.Scanner.nextInt(Scanner.java:2076)
        at unidad1_2020.EjemploScanner.main(EjemploScanner.java:23)
```

Este mismo error se origina también al ingresar un número decimal usando el punto para indicar la parte decimal. Al ingresar un decimal se debe utilizar **la coma.**

Una vez que el método close es invocado la lectura de datos ya no estará disponible. Si se intenta utilizar el objeto cerrado se produce un error con mensaje java.lang.IllegalStateException: Scanner closed.

Ejemplo:

Se va a tratar de leer la profesión luego de la línea del close

```
    System.out.print("Ingrese su salario:");
    double salario = sc.nextDouble();
    sc.close();

    System.out.print("Ingrese su profesión:");
    String profesión = sc.nextLine();
```

```
Exception in thread "main" java.lang.IllegalStateException: Scanner closed
Ingrese su profesión:   at java.util.Scanner.ensureOpen(Scanner.java:1070)
        at java.util.Scanner.findWithinHorizon(Scanner.java:1670)
        at java.util.Scanner.nextLine(Scanner.java:1538)
        at unidad1_2020.EjemploScanner.main(EjemploScanner.java:31)
```

Limpiar el buffer de entrada

Cada vez que se ingresan datos por teclado y se presiona enter, el salto de línea (\n) también pasa al buffer de entrada. El método nexLine() lo elimina del buffer pero los otros métodos no, por lo que este valor se queda en el buffer ocasionando comportamientos no esperados. Por ejemplo al ejecutar el siguiente código:

```
System.out.print("Ingrese un número:");
int numero = sc.nextInt();
System.out.print("Ingrese su nombre:");
// ingreso de string
String nombre = sc.nextLine();
System.out.print("Ingrese su edad:");
int edad = sc.nextInt();
```

La salida es la siguiente

```
run:
Ingrese un número:5
Ingrese su nombre:Ingrese su edad:
```

El valor de 5 se asigna a la variable número, pero el salto de línea se queda en el buffer. Al llegar a la instrucción nextLine(), ésta toma del buffer todos los caracteres hasta llegar al salto de línea y lo elimina. Para el ejemplo a la variable nombre se asigna una cadena vacía y se continúa con la ejecución impidiendo que el programa se detenga para realizar el ingreso del nombre.
Para solucionar este inconveniente se puede limpiar el buffer de entrada escribiendo un nextLine() luego de una instrucción que recibe un dato numérico. El ejemplo quedaría así:

```
System.out.print("Ingrese un número:");
int numero = sc.nextInt();
sc.nextLine();
System.out.print("Ingrese su nombre:");
// ingreso de string
String nombre = sc.nextLine();
System.out.print("Ingrese su edad:");
int edad = sc.nextInt();
```

Ejercicios

1. Lea la cantidad de kWh que ha consumido una familia y el precio por cada kWh. Si la cantidad es mayor a 700, incremente el precio en 5% para el exceso de kWh sobre 700. Muestre el valor total a pagar

2. Solicitar el ingreso de 5 números enteros y calcular el promedio. Realice este ejercicio usando for, while y do-while

3. Solicite el ingreso de 5 años y para cada año muestre si es un año bisiesto o no. Asuma que el año se ingresará como un valor entero esperado

4. Construir un programa que simule el juego de la adivinanza de un número. El ordenador debe generar un número aleatorio entre 1 y 100 y el usuario tiene cinco oportunidades para acertarlo. Después de cada intento el programa debe indicarle al usuario si el número introducido por él es mayor, menor o igual al número a adivinar, y el número de intentos restantes.

   Nota: para generar el valor aleatorio puede emplearse la sentencia:

```
short x  = (short) (100*Math.random()+1);
```

5. Escribir un programa que realice lo siguiente:

- Solicite el ingreso de las 3 notas de una materia obtenidas por un estudiante durante un semestre en ESPOL. Los datos que debe ingresar son:
  Número de matrícula
  Nota primer parcial
  Nota segundo parcial
  Nota mejoramiento.

Asuma que los datos ingresados son valores correctos(números enteros en el rango apropiado)

- Deberá mostrar un mensaje si el estudiante aprueba o no la materia. Recuerde que un estudiante aprueba una materia si suma 120 puntos en dos evaluaciones.
- Luego del ingreso preguntará si desea realizar más ingresos. Deberá volver a solicitar los datos hasta que escriba N.

Ejemplo de salida
I*ngrese numero de matricula 201799999*
Ingrese nota primer parcial 60
Ingrese nota segundo parcial 50
Ingrese nota mejoramiento 55
Estudiante reprueba la materia
Desea continuar (S/N): S
Ingrese numero de matricula 201788888
Ingrese nota primer parcial 70
Ingrese nota segundo parcial 50
Ingrese nota mejoramiento 0
Estudiante aprueba la materia
Desea continuar (S/N): N

6. Una línea de autobuses cobra un mínimo de $20 por persona y trayecto. Si el trayecto es mayor de 200 km el ticket tiene un recargo de 0.10 por km adicional. Sin embargo, para trayectos de más de 400 km el ticket tiene un descuento del 15 %. Por otro lado, para grupos de 3 o más personas el billete tiene un descuento del 10 %. Con las consideraciones anteriores, escriba en Java un programa que recibe por parámetros la distancia del viaje a realizar, así como el número de personas que viajan juntas. Con ello se debe calcular tanto el precio del billete individual como el total a pagar si viaja más de una persona. Muestre en pantalla los resultados.