UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ciencias

Introducción a las Ciencias de la Computación

Práctica 4: Instrucciones de control I

Profesora:
Amparo López Gaona
Ayudante:
Víctor Emiliano Cruz Hernández
Ayudante de Laboratorio:
Kevin Jair Torres Valencia

Objetivos

El objetivo de esta práctica es que el alumno ejercite la teoría acerca de la creación y uso de objetos de clases, así como la de aplicación de las instrucciones condicionales de un programa y refuerze sus conocimientos en sentencias de control de flujo.

Introducción

Para que un programa pueda trabajar con información proporcionada por el teclado, Java tiene en el paquete `java.util` la clase `Scanner`. Para crear objetos de esta clase se debe incluir la instrucción `import java.util.Scanner` al inicio del archivo en donde está el programa. Al crear un objeto de la clase Scanner se debe de dar como parámetro el objeto `System.in` para que la lectura sea de teclado.

En la clase Scanner hay un método de lectura para cada tipo de dato primitivo. La firma de los métodos son los siguientes:

- `boolean nextBoolean()`
- `byte nextByte()`
- `double nextDouble()`
- `int nextInt()`
- `long nextLong()`
- `float nextFloat()`
- `short nextShort()`
- `String nextLine()`

Por otro lado, la ejecución de las instrucciones en todo programa es secuencial, es decir, se realiza una instrucción después de otra a menos que se tenga una instrucción condicional.

La instrucción condicional `if` tiene la siguiente sintaxis: empieza con la palabra reservada `if` seguida de una expresión booleana, denominada condición, entre paréntesis; luego un bloque de instrucciones que se ejecutan cuando la evaluación de la condición es verdadera.

```
if( boolean condition ) { instructions }
```

Un bloque es un conjunto de instrucciones agrupadas con un par de llaves ({ }) con el propósito de ser tratadas como unidad. Dentro de un bloque puede haber otros bloques. Las instrucciones contenidas en el bloque de la instrucción `if` se ejecutaran siempre y cuando el resultado de la evaluación de la condición sea `true`, en caso contrario se ignoraran.

Las sentencias de control son fundamentales para cualquier lenguaje de programación, ya que estas permiten definir el flujo o comportamiento que debe seguir el programa.

También tenemos la palabra reservada `else` que nos sirve para indicar el bloque que se ejecutara si acaso la condición regresa un `false`.

```
if( boolean condition ) { instructions } else { instructions }
```

Donde la condición es una expresión que se debe evaluar y devolver un booleano el cual indica que líneas de código deberá ejecutar. Una manera alternativa y corta de escribir una instrucción de control como el `if` es el operador ternario, este operador tiene la siguiente sintaxis:

```
( boolean condition ) ?  Instructions :  Instructions;
```

Una alternativa cuando requerimos utilizar una mayor cantidad de valores para tomar decisiones es la utilización de la estructura `switch`. `switch` es una estructura de control en Java que se utiliza para tomar decisiones basadas en el valor de una variable. Permite evaluar una variable y ejecutar diferentes bloques de código dependiendo del valor de esa variable. Cada posible valor de la variable se denomina "caso", y se especifica con la palabra clave `case`. Se puede proporcionar un bloque de código para ejecutar para cada caso utilizando la palabra clave `break` para indicar el final de cada caso. Si ninguno de los casos coincide con el valor de la variable, se puede proporcionar un bloque de código predeterminado utilizando la palabra clave `default`. A continuación se da la estructura de la instrucción switch:

```
switch (variable) {
    case valor1:
        // código si variable es igual a valor1
        break;
    case valor2:
        // código si variable es igual a valor2
        break;
    // más casos...
    default:
        // código si variable no coincide con ninguno de los casos anteriores
}
```

Desarrollo

1. Escribe un programa que indique si una letra es vocal, consonante o ninguno de los dos. Debes usar `switch` y tú programa solo recibira un valor de tipo `char`.

2. Escribe un programa que reciba tres valores enteros positivos que representan las longitudes de los lados de un triángulo. El programa debe determinar y mostrar qué tipo de triángulo forman según las siguientes reglas:

- Equilátero: Todos los lados son iguales.
- Isósceles: Dos lados son iguales y el tercero es diferente.
- Escaleno: Todos los lados son diferentes.
- No es un triángulo: Si los valores no cumplen con la desigualdad triangular.

Como restricción los valores de los lados deben ser positivos (> 0). Si se ingresa un número inválido, debe mostrar un mensaje de error. Y si el programa solo puede resolverse usando `if-else` o `switch`

Formato de Entrega

1. Las prácticas serán entregadas en parejas y solo un integrante debe entregarla.
2. Cada práctica (sus archivos y directorios) deberá estar contenida en un directorio llamado `EquipoX_pY`, donde:
   (a) X es el número de equipo correspondiente.
   (b) Y es el número de la práctica.

   Por ejemplo: `Equipo01_p01`
3. NO incluir los archivos .class dentro del directorio a entregar.
4. Los archivos de código fuente deben estar documentados.
5. Se pueden discutir y resolver dudas entre los integrantes del grupo. Pero cualquier práctica plagiada total o parcialmente será penalizada con cero para los involucrados.
6. La práctica se debe entregar en el apartado de Github Classroom correspondiente.
7. El horario y día de entrega se acordará en la clase de laboratorio y no deberá sobrepasar 2 clases de laboratorio.