

UNIVERSIDAD NACIONAL AUTÓNOMA DE
MÉXICO
Facultad de Ciencias



Introducción a las Ciencias de la Computación

Práctica 3: Instrucciones de control de flujo I.

Profesora: Amparo López Gaona
Ayudante: Ramsés Antonio López Soto
Ayudantes de Laboratorio:
Adrián Aguilera Moreno
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 ejecutarán siempre y cuando el resultado de la evaluación de la condición sea `true`, en caso contrario se ignorarán.

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 ejecutará 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 genere mensajes telegráficos. El programa para generar mensajes telegráficos requiere los siguientes datos:

1. El nombre de la persona que va a enviar el telegrama.
2. La profesión del remitente.
3. Nombre de la persona a quien se va a enviar el telegrama.
4. La profesión del destinatario.
5. El contenido del telegrama a enviar.
6. Decir si el telegrama se va a enviar como Urgente.
7. La dirección a la que se enviara el telegrama en el formato `Calle numero.ciudad CP`

A partir de los datos proporcionados, el programa debe mostrar en pantalla el telegrama con el siguiente formato:

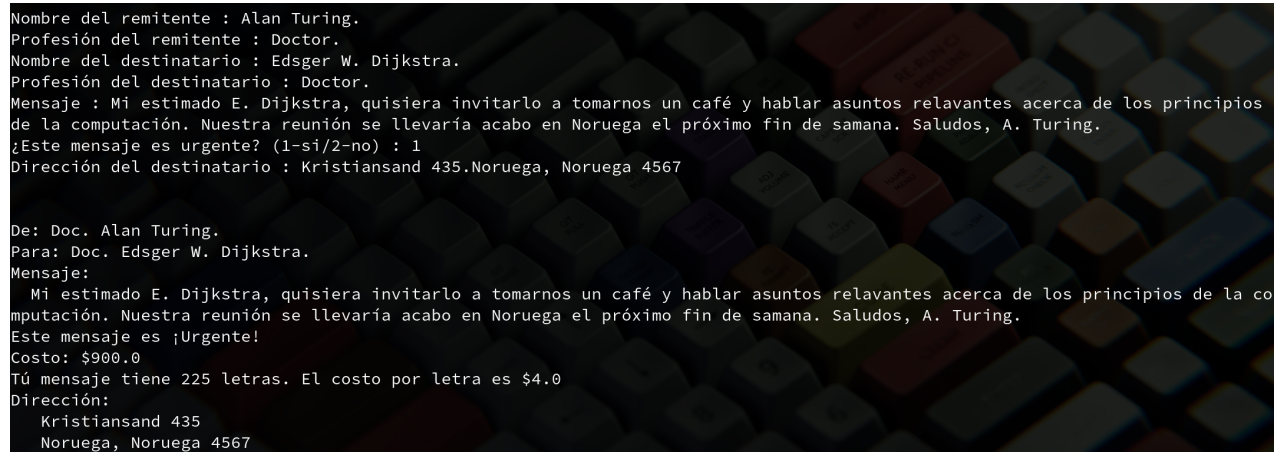
- **De:**
Seguida de las primeras tres letras de la profesión, un punto y del nombre del remitente. El nombre y la profesión deben escribirse solo con mayúsculas.
- **Para:**
Seguida de las primeras tres letras de la profesión, un punto y del nombre del destinatario. El nombre y la profesión deben escribirse solo con mayúsculas.
- **Mensaje:**
El contenido del mensaje.
- **Costo:** Costo del mensaje. Debe de indicarse si acaso el mensaje fue urgente o no.

Considerar que cada carácter en el contenido del mensaje, tiene un costo de 2.00 pesos en una entrega normal y de 4.00 pesos en una entrega urgente.

NOTA: Solo puedes usar el operador ternario como estructura de control en este apartado.

- La cantidad de caracteres en el texto y el costo de cada uno.
- La dirección en mayúsculas y ocupando dos lineas. En la primera, calle y número y en la segunda la ciudad y el código postal.

Un ejemplo del formato de entrada y salida para este problema es el siguiente:



Nombre del remitente : Alan Turing.
Profesión del remitente : Doctor.
Nombre del destinatario : Edsger W. Dijkstra.
Profesión del destinatario : Doctor.
Mensaje : Mi estimado E. Dijkstra, quisiera invitarlo a tomarnos un café y hablar asuntos relavantes acerca de los principios de la computación. Nuestra reunión se llevaría acabo en Noruega el próximo fin de samana. Saludos, A. Turing.
¿Este mensaje es urgente? (1-si/2-no) : 1
Dirección del destinatario : Kristiansand 435.Noruega, Noruega 4567

De: Doc. Alan Turing.
Para: Doc. Edsger W. Dijkstra.
Mensaje:
Mi estimado E. Dijkstra, quisiera invitarlo a tomarnos un café y hablar asuntos relavantes acerca de los principios de la co
mputación. Nuestra reunión se llevaría acabo en Noruega el próximo fin de samana. Saludos, A. Turing.
Este mensaje es ¡Urgente!
Costo: \$900.0
Tú mensaje tiene 225 letras. El costo por letra es \$4.0
Dirección:
Kristiansand 435
Noruega, Noruega 4567

2. Escribe un programa que lea tres enteros a, b, c y determine si c se encuentra en el intervalo $[a, b]$ a la izquierda o a la derecha del intervalo $[a, b]$. Debes imprimir INTERVALO, IZQUIERDA o DERECHA según sea el caso. E.g. Para los números 3 10 6 la salida de tú programa debe ser INTERVALO, para los números 4 20 30 la respuesta debe ser DERECHA y para los valores 2 30 1 la respuesta debe ser IZQUIERDA.

Observación: Para los valores 10 20 10 la salida debe ser INTERVALO.

NOTA: Para este ejercicio solo debes usar if, else if y else como estructura de control.

Formato de Entrega

1. Las prácticas serán entregadas de forma individual.
2. Cada práctica (sus archivos y directorios) deberá estar contenida en un directorio llamado `apellidoPaterno_nombre_pX`, donde X es el número de la práctica.
Por ejemplo: `aguilera_adrian_p1`
3. NO incluir los archivos `.class` dentro de la carpeta.
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 subir al Github Classroom correspondiente.
7. La entrega en classroom debe contener el link HTTPS y SSH de su repositorio y es lo único que se debe entregar.
8. El horario y día de entrega se acordará en la clase de laboratorio y no deberá sobrepasar 2 clases de laboratorio.