	Carátula Para Entrega De Prácticas
Facultad de Ingeniería	Laboratorios de docencia

Laboratorio de Computación Salas A y B

Profesor(a): Josè Antonio Ayala Barbosa

Asignatura: Programación Orientada a Objetos

Grupo: 8

No de Práctica(s): Practica #2 “Fundamentos y Sintaxis Del Lenguaje”

Integrante(s): Galindo Mayer Johann Zair

No. de Equipo de cómputo empleado:

Semestre: 2024 - 1

Fecha de entrega: 08/09/2023

Observaciones:

CALIFICACIÓN: _____

Practica 2 “Fundamentos y Sintaxis Del Lenguaje”

Galindo Mayer Johann Zair

Facultad De ingeniería

UNAM

CDMX, México

Josephum45@outlook.com

Cuestionario Previo a La Practica

1. Revisión de la clase KeyboardInput.java

Proporciona eventos, métodos y propiedades relacionados con el teclado que proporcionan información sobre el estado del teclado.

2. ¿Qué son las abreviaturas NetBeans?

Es un entorno de desarrollo integrado de código abierto y gratuito para el desarrollo de aplicaciones en los sistemas operativos Windows,Mac,Linux y Solaris.

3. Menciona al menos 10 abreviaturas

IDE: Abreviatura de "Integrated Development Environment" (Entorno de Desarrollo Integrado), que se refiere a la plataforma en la que los desarrolladores pueden escribir, compilar, depurar y probar su código en un solo lugar. NetBeans es un ejemplo de un IDE.

GUI: Abreviatura de "Graphical User Interface" (Interfaz de Usuario Gráfica), que se utiliza para crear interfaces visuales en aplicaciones. NetBeans proporciona herramientas para diseñar interfaces gráficas de usuario de manera intuitiva.

JDK: Abreviatura de "Java Development Kit" (Kit de Desarrollo de Java), que es un conjunto de herramientas necesarias para desarrollar y ejecutar aplicaciones Java. NetBeans generalmente se utiliza junto con un JDK para desarrollar aplicaciones Java.

API: Abreviatura de "Application Programming Interface" (Interfaz de Programación de Aplicaciones), que define cómo los componentes de software deben interactuar entre sí. NetBeans proporciona una variedad de API para desarrollar extensiones y complementos personalizados.

Maven: Herramienta de gestión de proyectos que se utiliza para administrar la construcción, dependencias y documentación de proyectos. NetBeans tiene integración con Maven para facilitar el manejo de proyectos Java.

JavaFX: Plataforma para crear aplicaciones de interfaz gráfica de usuario en Java. NetBeans

proporciona soporte para desarrollar aplicaciones JavaFX.

DB: Abreviatura de "Database" (Base de Datos). NetBeans ofrece herramientas para trabajar con bases de datos y desarrollar aplicaciones que interactúen con sistemas de gestión de bases de datos.

IDEA: No directamente relacionado con NetBeans, pero puede crear confusión. "IDEA" es la abreviatura de "IntelliJ IDEA", otro popular entorno de desarrollo integrado utilizado para desarrollar aplicaciones en varios lenguajes, incluido Java.

4. ¿Qué son los snippets?

Snippet es un fragmento con una pequeña muestra del contenido de la web que se muestra a los usuarios en la página de resultados. Este fragmento muestra información de lo que puede encontrarse en dicha dirección, incluyendo un título, la URL, una pequeña descripción y, dependiendo de la página, información adicional.

Objetivo De La Practica

Crear programas que implementen variables y constantes de diferentes tipos de datos, expresiones y estructuras de control de flujo.

Introducción

Dentro del contexto de la programación orientada a objetos, las clases tienen un papel fundamental, ya que nos permiten abstraer los datos y las operaciones relacionadas con ellos. Esto nos permite tratarlos como si fueran entidades opacas o cajas que no podemos ver ni modificar directamente. La mayoría de los lenguajes de programación modernos admiten el uso de clases y las ventajas que ofrecen se traducen en diversas utilidades. Una clase puede entenderse como la especificación de un conjunto de objetos similares, cada uno de los cuales posee métodos y datos que resumen las características comunes de ese conjunto. En un lenguaje de programación orientado a objetos como Java, podemos crear muchos objetos de la misma clase, pero cada uno con características distintas, como textura, material, acabado, color, entre otros. Las clases son semejantes a los tipos abstractos de datos y sirven como modelos que describen como se crean ciertos tipos de objetos.

Desarrollo

Se desarrollo un programa que posibilita el empleo de una variedad de variables y diversos tipos de datos simultáneamente. Además, se crearon múltiples expresiones y se incorporaron estructuras de control, tales como condiciones (if/else), casos (switch), bucles (for, while) entre otras. También utilizo la

función System.out.println para mostrar un mensaje de saludo al usuario.

```java
/**
 *
 * @author Galindo Mayer Johann Zair
 */
public class POO_2 {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println(x: "Hola querido compañero programador o programadora");
    }

}
```

Empleamos la estructura condicional conocida como “If else” para tomar decisiones basadas en la relación entre dos números, denominados “a” y “b”. En un primer paso, definimos una variable de tipo entero llamada “a” sin asignarle un valor inicial. Posteriormente asignamos un valor de 18 variables “a”.

```java
public static void main(String[] args) {
        System.out.println(x: "-------- if --------\n");
    int a;
    a = 18;
    int b = 10;
    if (a<b) {
        System.out.println(x: "a es menor que b");
    } else if (a==b){
        System.out.println(x: "a es igual a b");
    }else{
        System.out.println(x: "a es mayor que b");
    }
```

Utilizamos la estructura de control denominada “switch”, la cual nos habilita para tomar decisiones en función del valor almacenado en una variable llamada “Mes”. En otras palabras, dependiendo del numero asignado a esta variable, la estructura “switch” compara este valor con todos los casos posibles hasta encontrar la correspondencia adecuada. En este momento, mediante el uso de “break;”, se detiene la ejecución del programa.

```java
System.out.println(x: "-------- Switch --------\n");
int mes = 3;
switch (mes) {
    case 1 -> System.out.println(x: "Enero");
    case 2 -> System.out.println(x: "Febrero");
    case 3 -> System.out.println(x: "Marzo");
    case 4 -> System.out.println(x: "Abril");
    case 5 -> System.out.println(x: "Mayo");
    case 6 -> System.out.println(x: "Junio");
    case 7 -> System.out.println(x: "Julio");
    case 8 -> System.out.println(x: "Agosto");
    case 9 -> System.out.println(x: "Septiembre");
    case 10 -> System.out.println(x: "Octubre");
    case 11 -> System.out.println(x: "Novivembre");
    case 12 -> System.out.println(x: "Diciembre");
    default -> System.out.println(x: "Mes Incorrecto");
}
```

Ahora, utilizamos los bucles conocidos como "While" y "Do-While" en Java, los cuales nos permiten llevar a cabo tareas de conteo tanto hacia arriba como hacia abajo. Comenzamos declarando una variable llamada "n" y le asignamos un valor inicial de 0, asegurándonos de que sea de tipo entero. En el caso del ciclo "While", lo empleamos para ejecutar un bloque de código mientras una condición específica no se cumpla. Utilizamos la línea de código "System.out.println("Contando hacia arriba n="+n);" para mostrar el valor de "n" en cada

iteración del bucle, mientras la condición se mantenga verdadera.

Luego, con el bucle "Do-While", iniciamos con la línea de código "do{…} while (n > 0)", que establece un bucle que se ejecutará al menos una vez antes de verificar la condición. La condición en este caso es que "n" sea mayor que cero, lo que significa que el bucle continuará ejecutándose mientras "n" tenga un valor superior a 0. Para visualizar el progreso del bucle, utilizamos "System.out.println("Contando hacia abajo");" para imprimir un mensaje en cada iteración.

Dentro del bucle "Do-While", decrementamos el valor de "n" en 1 en cada iteración, lo que significa que disminuye en cada ciclo. Cuando "n" finalmente llega a 0, el bucle "do-while" concluye su ejecución y se procede al siguiente bloque de código. Al final, empleamos "System.out.println("n="+n)" para mostrar el valor actual de "n" después del bucle "do-while". En este punto, "n" será igual a -1, ya que el bucle "do-while" se ejecuta al menos una vez antes de verificar la condición.

```
System.out.println(x: "-------- While --------\n");
int n=0;
while (n<10) {
    System.out.println("Contando hacia arriba n="+n);
    n++;
}
while (n>0) {
    System.out.println("Contando hacia abajo n="+n);
    n--;
}
System.out.println("n="+n);

System.out.println(x: "-------- Do-While --------\n");
do {
    System.out.println(x: "Contando hacia abajo");
    n--;
} while (n>0);
System.out.println("n="+n);

System.out.println(x: "-------- For --------\n");
for (int i = 0; i < 10; i++) {
    System.out.println("Contando hacia arriba "+i);
}

for (int i = 10; i > 0; i--) {
    System.out.println("Contando hacia abajo " + i);
}
```

En este fragmento de código, empleamos los bucles "for" y "for each" para trabajar con arreglos que podemos definir dentro del programa. Comenzamos declarando un arreglo llamado "miArreglo" y le asignamos los valores del 1 al 10, lo que se logra mediante la línea "int[] miArreglo = {1,2,3,4,5, 6, 7, 8, 9, 10}."

A continuación, utilizamos la variable "numElementos" para calcular la longitud del arreglo "miArreglo" con la expresión "int numElementos = miArreglo.length;". Esto nos proporciona el número de elementos en el arreglo, que luego imprimimos en la consola con "System.out.println("miArreglo tiene "+numElementos+" elementos");".

Luego, declaramos un nuevo arreglo llamado "miArreglo2" con 10 elementos, inicialmente todos establecidos en 0 mediante la línea "int[] miArreglo2 = new int[10];". En el primer bucle "for", recorremos "miArreglo2" y asignamos valores a cada posición multiplicando "i" por 10. Esto resulta en que "miArreglo2" se llena con valores que son múltiplos de 10.

En el segundo bucle "for", volvemos a recorrer "miArreglo2" y mostramos cada elemento en la consola. En el tercer bucle "for", iteramos nuevamente sobre "miArreglo2" y aplicamos una operación de porcentaje a cada valor, seguido de la impresión de los resultados.

Finalmente, empleamos el bucle "for-each" para trabajar con "miArreglo2" y utilizamos la línea "System.out.println("-------- For-each --------");" para indicar en la terminal que se está ejecutando la sección de código relacionada con el bucle "for-each".

```
System.out.println(x: "-------- For --------\n");
  int[] Array = {1,2,3,4,5, 6, 7, 8, 9,10};
  int numElementos = Array.length;
  System.out.println("Array tiene "+numElementos+" elementos");
  int[] Array2 = new int[10];
  numElementos = Array2.length;
  System.out.println("Array2 tiene "+numElementos+" posiciones");
  for (int i = 0; i < Array2.length; i++) {
      Array2[i] = i*10;
  }
  for (int i = 0; i < Array2.length; i++) {
      System.out.println("Array2["+i+"]="+Array2[i]);
      System.out.println(x: "\n");
  }
  for (int i = 0; i < Array2.length; i++) {
      int j = Array2[i];
      System.out.println("Mapeo en porcentaje "+j+"%\n");
  }

  System.out.println(x: "-------- For-each --------");
  for (int i : Array2) {
      System.out.println("Sustraccion de datos completado "+i+"%");
  }
}

private static boolean menor(int x, int y) {
  return x<y;
}
```

```
/**
```

```
 *
 * @author Galindo Mayer Johann Zair
 */
public class POO_2 {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println("Hola querido compañero programador o programadora");
    }

}
```

```
        System.out.println("-------- if --------\n");
        int a;
        a = 18;
        int b = 10;
        if (a<b) {
            System.out.println("a es menor que b");
        } else if (a==b){
            System.out.println("a es igual a b");
        }else{
            System.out.println("a es mayor que b");
        }
System.out.println("-------- if --------");
        if(!menor(a,b)){
            System.out.println("metodo menor retorna falso");
        }else{
            System.out.println("metodo menor retorna verdadero");
        }

System.out.println("-------- Switch --------\n");
        int mes = 3;
        switch (mes) {
            case 1 -> System.out.println("Enero");
            case 2 -> System.out.println("Febrero");
            case 3 -> System.out.println("Marzo");
            case 4 -> System.out.println("Abril");
            case 5 -> System.out.println("Mayo");
            case 6 -> System.out.println("Junio");
            case 7 -> System.out.println("Julio");
            case 8 -> System.out.println("Agosto");
            case 9 -> System.out.println("Septiembre");
            case 10 -> System.out.println("Octubre");
            case 11 -> System.out.println("Novivembre");
            case 12 -> System.out.println("Diciembre");
```

```
        default -> System.out.println("Mes Incorrecto");
    }
System.out.println("-------- Switch --------");
    char vocal = '9';
    switch (vocal) {
        case 'a' -> System.out.println("Seleccionó número 1");
        case 'e' -> System.out.println("Seleccionó número 2");
        case 'i' -> System.out.println("Seleccionó número 3");
        case 'o' -> System.out.println("Seleccionó número 4");
        case 'u' -> System.out.println("Seleccionó número 5");
        default -> {
            System.out.println("No existe el numero seleccionado");
            System.out.println("\n");
        }
System.out.println("-------- While --------\n");
    int n=0;
    while (n<10) {
        System.out.println("Contando hacia arriba n="+n);
        n++;
    }
    while (n>0) {
        System.out.println("Contando hacia abajo n="+n);
        n--;
    }
    System.out.println("n="+n);

    System.out.println("-------- Do-While --------\n");
    do {
        System.out.println("Contando hacia abajo");
        n--;
    } while (n>0);
    System.out.println("n="+n);

    System.out.println("-------- For --------\n");
    for (int i = 0; i < 10; i++) {
```

```
        System.out.println("Contando hacia arriba "+i);
      }

      for (int i = 10; i > 0; i--) {
        System.out.println("Contando hacia abajo " + i);
      }

System.out.println("-------- For --------\n");
      int[] Array = {1,2,3,4,5, 6, 7, 8, 9,10};
      int numElementos = Array.length;
      System.out.println("Array tiene "+numElementos+" elementos");
      int[] Array2 = new int[10];
      numElementos = Array2.length;
      System.out.println("Array2 tiene "+numElementos+" posiciones");
      for (int i = 0; i < Array2.length; i++) {
        Array2[i] = i*10;
      }
      for (int i = 0; i < Array2.length; i++) {
    System.out.println("Array2["+i+"]="+Array2[i]);
        System.out.println("\n");
      }
      for (int i = 0; i < Array2.length; i++) {
        int j = Array2[i];
        System.out.println("Mapeo en porcentaje "+j+"%\n");
      }
      System.out.println("-------- For-each --------");
      for (int i : Array2) {
        System.out.println("Sustraccion de datos completado "+i+"%");
      }
    }

  private static boolean menor(int x, int y) {
    return x<y;
  }
```

Conclusiones

En resumen, la realización de la segunda práctica nos brindó una valiosa experiencia en el manejo de variables y constantes con diversos tipos de datos, así como en la implementación de expresiones y estructuras de control de flujo en programación. Además, esta práctica nos permitió tener un primer contacto con los conceptos de programación orientada a objetos, aunque no profundizamos en ellos, nos sirvió como una introducción importante para comprender mejor esta área de estudio.

Durante la práctica, pudimos aplicar los conocimientos teóricos adquiridos en clase y poner en práctica lo aprendido en la primera actividad de laboratorio. En general, esta experiencia nos ayudó a fortalecer nuestra comprensión y habilidades en programación, sentando las bases para abordar futuros desafíos en la asignatura.