	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 #3 “Utilerías y Clases De Uso General”
Integrante(s): Galindo Mayer Johann Zair
No. de Equipo de cómputo empleado:
Semestre: 2024 - 1
Fecha de entrega: 22/09/2023
Observaciones:

CALIFICACIÓN: _____

Practica 2 “Fundamentos y Sintaxis Del Lenguaje”

Galindo Mayer Johann Zair

Facultad De ingeniería

UNAM

CDMX, México

Cuestionario Previo a La Practica

1. ¿Qué es el API en Java?

Application Programming Interface es un conjunto de clases, interfaces, métodos y constantes predefinidos que proporcionan funcionalidades específicas para desarrollar aplicaciones en el lenguaje de programación Java. Estas funcionalidades pueden abarcar desde operaciones básicas como manipulación de cadenas y estructuras de datos hasta tareas más avanzadas.

2. ¿Qué es una utilería en programación?

Es una herramienta que sirve de soporte para la construcción y ejecución de programas, en donde se incluyen las bibliotecas de sistema, middleware, herramientas de desarrollo, etc.

3. ¿Qué es una colección en programación?

Representa un grupo de objetos, conocidos como sus elementos. Las colecciones permiten elementos duplicados y los mismos no están ordenados. Los elementos de una colección no son accedidos a través de un índice. Por el contrario, su única forma de recuperación es uno a uno, secuencialmente.

4. ¿Qué es un wrapper?

Se designa un embalaje o envoltorio. En el contexto del *software*, este término hace referencia a programas o códigos que rodean otros componentes de programa. Los pueden utilizarse por diversos motivos: a menudo se usan para mejorar la **compatibilidad** o **interoperabilidad** entre diferentes estructuras de software o para poder representar algo a nivel visual.

Objetivo:

Utilizar bibliotecas propias del lenguaje para realizar algunas tareas comunes y recurrentes.

Introducción:

Al trabajar en un problema de programación, normalmente se debe verificar si hay clases preconstruidas que satisfagan las necesidades del programa. Si existen esas clases, entonces hay que utilizarlas: “no tratar de reinventar la rueda”. Hay dos ventajas principales de usar clases preconstruidas: se puede ahorrar tiempo ya que no es necesario escribir nuevas; y el uso de clases preconstruidas también puede mejorar la calidad de los programas ya que han sido probadas completamente, depuradas y sometidas a un proceso de escrutinio para asegurar su eficiencia.

Desarrollo de la practica:

En esta práctica, el profesor no proporcionó una explicación detallada sobre cómo emplear las diversas bibliotecas, utilidades y clases de uso general, tal como se describe en los objetivos e introducción de la actividad. Durante la sesión, abordamos varios temas programáticos. El primero de ellos se centró en los arreglos, donde declaramos varios valores de tipo entero en arreglos diferentes. Luego, empleamos la instrucción System.out.println para mostrar estos valores en la terminal del entorno de desarrollo integrado (IDE). Además, hicimos uso de los valores previamente declarados en los arreglos al implementar ciclos de repetición, incluyendo el ciclo for y el ciclo for-each, para imprimirlos en la pantalla de manera similar.

```
public static void main(String[] args) {

    System.out.println(x: "######### Arrays #########");
    int [] numeros;
    int [] nums;
    int num2;
    int num3[];
    int [] nums4 ={1,2,3,4};
    System.out.println(x: "######### for #########");
    for (int i=0; i < nums4.length; i++){
        int j= nums4[i];
        System.out.println(x: i);
    }
    System.out.println(x: "######### for - each #########");
    for (int i: nums4){
        System.out.println(x: i);
```

Observamos diversas técnicas para concatenar caracteres diferentes utilizando objetos de tipo String. Comenzamos desde la declaración en el programa de las palabras que se utilizarían y llegamos a la concatenación de múltiples variables para crear una sola línea de texto. Pasamos de escribir nuestros nombres y apellidos de forma separada a ser capaces de combinarlos utilizando el

operador "+" para luego imprimirlos en la terminal del entorno de desarrollo integrado (IDE).

```
System.out.println(x: "######### Concatenar #########");
String s = new String(original: "Hola Mundo");
String s1 = "Hola Mundo";
System.out.println(x: s);
System.out.println(x: s1);
String nombre ="Johann Zair";
String apellido = "Galindo Mayer";
String nombreCompleto = nombre+ " " +apellido;
System.out.println(x: nombreCompleto);
```

El profesor nos explicó la función del operador punto y su utilidad en la programación. En resumen, el operador punto se emplea para acceder a los miembros de una clase, que pueden ser campos (variables) o métodos (funciones) de un objeto. Esto se ilustró para demostrar tanto el número de elementos en un arreglo declarado como el número de elementos en la variable que contiene nuestro nombre.

```
System.out.println(x: "######### Operador Punto #########");
System.out.println("Numero de elementos del arreglo: "+nums4.length);
System.out.println("Numero de elementos nombre: "+nombre.length());
System.out.println(x: nombreCompleto.toUpperCase());
System.out.println(x: nombreCompleto);
```

Utilizamos lo que se conoce como "Wrappers" y "autoboxing". En Java, los "wrappers" y el "autoboxing" están relacionados con la conversión automática entre tipos primitivos y sus correspondientes objetos (clases envolventes o "wrappers"). El profesor nos explicó que esta característica se implementó en Java para simplificar el manejo de tipos de datos primitivos en situaciones en las que se requieren objetos.
A lo largo de la práctica, fuimos utilizando el código que ya estaba programado, como se muestra en la captura siguiente, en la que se aprecian ejemplos de cómo se aplican estas técnicas.

```
System.out.println(x: "######### Wrappers y Autob

Integer k = 50;
int r = 1;
int b = 7;
//Casteo
boolean s3 = false;

System.out.println(x: s3);
String s4 = r + "";
System.out.println(x: s4);
System.out.println (x: s4.length());
//Parseo to partse
//Caasteo to cast
```

Utilizamos lo que se conoce como "tablas hash," o en su nombre original "Hash Tables." Estas son una parte integral de la biblioteca estándar y se emplean de manera común para asociar un valor (value) con una clave (key) de manera rápida y eficiente. Las tablas hash se basan en el principio de dispersión (hashing), que permite indexar y buscar elementos en función de su clave. En nuestro programa, empleamos las tablas hash para crear una agenda en la que asignamos un número telefónico (Key) y un nombre (Value) a cualquier persona que deseamos registrar.

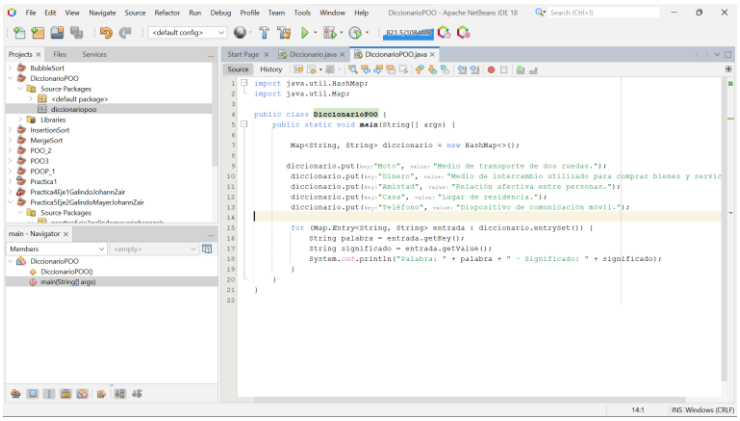
```
System.out.println(x: "######### Hash Table ########
Hashtable<Integer,String> agenda =new Hashtable<>()
agenda.put(key:k,value: "Zayra");
agenda.put(key:553884390,value: "Iron Man");
agenda.put(key:553677124,value: "Katherine");
agenda.put(key:554249890,value: "Maria Elena");
agenda.put(key:558366979,value: "Santiago");
System.out.println(x: agenda.size());
System.out.println (x: "***");
for (String valor : agenda.values()){
    System.out.println(x: valor);
}
for (Integer clave : agenda.keySet()){
    System.out.println(x: clave);
```

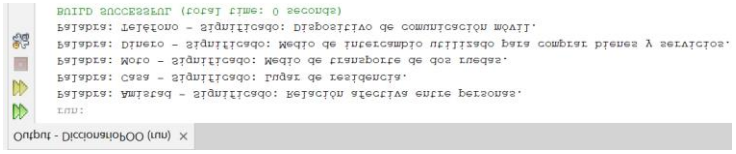
Por último, el profesor nos compartió dos consejos valiosos relacionados con la inclusión de paquetes en nuestro código, como es el caso del paquete "date." Cuando empleamos la instrucción "import java.util.Date;" en Java, estamos efectuando una declaración de importación que nos permite incorporar la clase "Date" del paquete "java.util" directamente en nuestro programa Java. De esta manera, podemos utilizar la clase en nuestro código sin necesidad de hacer mención al paquete completo cada vez que la requiramos. Este enfoque simplifica el uso de la clase y contribuye a que nuestro código sea más claro y conciso.

```
System.out.println(x: "######### Math ########
System.out.println(x: Math.PI);
System.out.println(x: Math.abs(a: -5));
System.out.println(x: Math.pow(a: 3,b: 2));
System.out.println(x: Math.sqrt(a: 9));
System.out.println(x: Math.max(a: 80,b: 7));
System.out.println(x: Math.min(a: 3,b: 65));

System.out.println(x: "######### Date ########
Date hoy = new Date();
System.out.println(x: hoy);
System.out.println(x: hoy.toString());
```

Actividad 1. Realiza un diccionario con 5 palabras usando Hash Tables y que imprima los elementos.





Código Fuente

```
package poo3;

/**
 *
 * @author Galindo Mayer Johann Zair
 */
import java.util.ArrayList;
import java.util.Date;
import java.util.Enumeration;
import java.util.Hashtable;

public class POO3 {

   public static void main(String[] args) {

      System.out.println("######### Arrays #########");
      int [] nums4 ={1,2,3,4};
      System.out.println("######### for #########");
      for (int i=0; i < nums4.length; i++){
         System.out.println(i);
      }
      System.out.println("######### for - each #########");
      for (int i: nums4){
         System.out.println(i);
      }
      System.out.println("######### Concatenar #########");
      String s = "Hola Mundo";
      String s1 = "Hola Mundo";
      System.out.println(s);
      System.out.println(s1);
      String nombre ="Johann Zair";
      String apellido = "Galindo Mayer";
      String nombreCompleto = nombre+ " " +apellido;
      System.out.println(nombreCompleto);

      System.out.println("######### Operador Punto #########");
      System.out.println("Numero de elementos del arreglo: "+nums4.length);
      System.out.println("Numero de elementos nombre: "+nombre.length());
      System.out.println(nombreCompleto.toUpperCase());
      System.out.println(nombreCompleto);

      System.out.println("######### Wrappers y Autoboxing #########");

      Integer k = 50;
      int r = 1;
      int b = 7;
      //Casteo
      boolean s3 = false;

      System.out.println(s3);
      String s4 = r + "";
      System.out.println(s4);
      System.out.println (s4.length());
      //Parseo to partse
      //Caasteo to cast

      System.out.println ("######### Collecciones #########");

      System.out.println ("######### Array List #########");
      ArrayList<Integer>miArrayList = new ArrayList<>();
      miArrayList.add(b);
      miArrayList.add(9);
      System.out.println (miArrayList.size());
      System.out.println (miArrayList.get(0));
      miArrayList.add(0,20);
      System.out.println ("***");
      for (Integer integer : miArrayList){
         System.out.println(integer);
      }
      System.out.println("######### Hash Table #########");
      Hashtable<Integer,String> agenda =new Hashtable<>();
      agenda.put(k,"Zayra");
      agenda.put(553884390,"Iron Man");
      agenda.put(553677124,"Katherine");
      agenda.put(554249890,"Maria Elena");
      agenda.put(558366979,"Santiago");
      System.out.println(agenda.size());
      System.out.println ("***");
      for (String valor : agenda.values()){
         System.out.println(valor);
      }
      for (Integer clave : agenda.keySet()){
         System.out.println(clave);
   }
      System.out.println("######### Enumeracion #########");
      Integer clave;
      String valor;
      Enumeration <Integer> llaves = (Enumeration<Integer>)agenda.keySet();
      while (llaves.hasMoreElements()){
         clave = llaves.nextElement();
         Object Clave = null;
         valor = agenda.get(Clave);
         System.out.println("clave: "+clave+ "valor: "+valor);
      }
      System.out.println("######### Math #########");
      System.out.println(Math.PI);
      System.out.println(Math.abs(-5));
      System.out.println(Math.pow(3,2));
      System.out.println(Math.sqrt(9));
      System.out.println(Math.max(80,7));
      System.out.println(Math.min(3,65));
```

```
        System.out.println("######### Date #########");
        Date hoy = new Date();
        System.out.println(hoy);
        System.out.println(hoy.toString());
    }
}
```

Conclusiones

Es evidente que comprender cómo emplear las bibliotecas, utilidades y clases de uso general en Java es esencial para la programación, ya sea para proyectos prácticos o tareas, como fue el caso de esta práctica. El dominio de los ciclos, como el ciclo for, do-while, while, entre otros, resulta fundamental para trabajar con arreglos o tablas hash. Sin lugar a duda, la realización de esta práctica ha sido de gran relevancia en nuestra preparación, ya que ha sentado las bases para un estudio más profundo en nuestras clases teóricas futuras.