**GUÍA DE APRENDIZAJE**

**Paquetes y Java API**

|  |  |
| --- | --- |
| **OBJETIVO DE APRENDIZAJE** | **OA 4**. Construir aplicaciones computacionales basadas en programación orientada a objetos, de manera de cumplir con las exigencias técnicas y de los usuarios. |
| **APRENDIZAJE ESPERADO** | **AE 5.** Construye programas, utilizando los componentes del paradigma orientado a objetos, a partir del análisis de problemáticas y requerimientos, modelando el comportamiento del sistema.  **AE 1.** Construye unidades de prueba para verificar el correcto funcionamiento de la codificación realizada, de acuerdo a exigencias técnicas de confiabilidad.  **AE 2.** Detecta y corrige errores de codificación, analizando el comportamiento del código de programación, de acuerdo a especificaciones y manuales de referencia. |
| **CRITERIOS DE EVALUACIÓN** | **5.1.** Construye los componentes a utilizar en el programa a partir del análisis de requerimientos para resolución de problema, resguardando los principios de POO.  **5.2.** Desarrolla interfaz de usuario simple, basado en programación orientada a objetos, para navegar a través del programa, considerando requerimientos circunstanciales de un determinado problema.  **1.1.** Realiza pruebas para detectar problemas previos a la codificación de una unidad de software con una herramienta de software disponible en el mercado.  **1.2.** Construye una unidad de prueba para detectar excepciones en un entorno de lenguaje con una herramienta compatible que permita desarrollar, configurar e Implementar aplicaciones en función del lenguaje utilizado.  **1.3.** Construye una unidad de prueba para detectar casos límites, en un entorno de lenguaje, con una herramienta compatible, propia del software utilizado como marco de trabajo.  **2.1.** Utiliza la herramienta de depuración de un entorno de desarrollo, para revisar la codificación de clases programadas.  **2.2.** Revisa y corrige las variables, del programa desarrollado para la solución de un caso, indagando en las funciones internas de acuerdo a protocolos de revisión.  **2.3.** Revisa las variables de una función de clases programadas, utilizando opciones del lenguaje empleado, para asegurar su correcto funcionamiento. |
| **OBJETIVOS DE APRENDIZAJE GENÉRICOS** | **A – B – C – G - H** |

## PAQUETES

Los paquetes son la herramienta que usa Java para estructurar y facilitar la modularidad del código. Estas proporcionan una forma de nombrar una colección de clases y actúan como mecanismo de control de acceso en Java, es decir, que las clases definidas dentro del paquete se pueden hacer privadas para ese paquete, proporcionando un grado de encapsulamiento.









package contabilidad;

Java usa el sistema de archivos para administrar los paquetes, almacenándolos en un directorio **(carpeta).** Por ejemplo, los archivos **.class** para las clases que sean parte del paquete contabilidad, se encuentran en un directorio con el mismo nombre.

También es posible declarar una jerarquía de paquetes mediante el uso del punto.

package ruta.al.paquete.algo;  
  
public class Algo {  
 *//Hago algo*  
}

Para usar los paquetes se usa el **comando import.**

import ruta.al.paquete.algo.Algo;  
import ruta.al.paquete.algo.\*;

Tal como se había mencionado, los paquetes participan en el mecanismo de control de acceso de Java, por lo tanto, la visibilidad de un elemento está determinada por su definición dentro de la clase y el paquete. Si un miembro de una clase no tiene modificador de acceso explícito, será visible dentro de su paquete, pero no fuera de él. Los miembros declarados como ***public***, son los más visibles y se pueden declarar de diferentes clases y paquetes. Por último, los miembros declarados como ***private***son únicamente accesibles por otros miembros de la misma clase.

## JAVA API

En primer lugar, una API significa Application Programming Interface (Interfaz de programación de aplicaciones) es un conjunto de subrutinas, métodos que ofrece una librería para ser utilizada por otro *software*. Java proporciona un conjunto de paquetes que podemos ver en la documentación [**oficial**](https://docs.oracle.com/javase/6/docs/api/), encontrando paquetes muy usados como **java.util** y contiene clases ya usadas por ustedes anteriormente.

1. Contiene la clase **Scanner** que se utiliza para obtener la entrada del usuario.
2. Contiene la clase **Random** que es usada para crear números de manera aleatoria.
3. Permite gestionar fechas, calendarios y horas. ¿Se imaginan programar un calendario?
4. Permite crear [**estructuras de datos**](https://es.wikipedia.org/wiki/Estructura_de_datos) dinámicas.
5. ¡Y mucho más!

A continuación, se revisarán algunas estructuras de datos dinámicos como **ArrayList**, **Set** y **Map**, proporcionadas por el paquete java.util.

## ARRAYLIST

Supongamos que necesitamos crear un programa que gestione los libros de una biblioteca. Si usamos un arreglo de libros, el programa tiene un límite de libros que puede agregar. **¿Cómo podemos ingresar más libros?**

Book[] books = new Book [10];

Creando un **ArrayList**, que básicamente es una lista de objetos de tamaño modificable, internamente modificada con arreglos. Dentro de sus características están:

1. Obtener/colocar elementos por índice.
2. Agregar elementos.
3. Eliminar elementos.
4. Iterar sobre los elementos.

import java.util.ArrayList;  
  
class ArrayListEjemplo {  
 public static void main (String [] args) {  
 *//Se crea el ArrayList nombre*  
 ArrayList<String> nombre = new ArrayList<String>();  
 *//Se agregan nombres al Arraylist*  
 nombre.add("Aquiles");  
 nombre.add("Temistocles");  
 nombre.add("Petronila");  
  
 System.out.println(nombre.size());  
 System.out.println(nombre.get(0));  
 System.out.println(nombre.get(1));

*//Reemplaza un elemento en la posición especificada*  
 nombre.set (0, "Adiós");  
 *//Remueve un elemento en la posición especificada*  
 nombre.remove(1);  
   
 *//obteniendo el listado de nombres*  
 for (int i = 0; i < nombre.size() ; i++) {  
 System.out.println (nombre.get(i));  
 }  
 for (String s: nombre) {  
 System.out.println (s);  
 }  
 }  
}

## SETS

Es similar a un **ArrayList** pero no contiene elementos duplicados y tampoco posee un índice en el arreglo. Dentro de sus características están:

1. Agregar elementos al **Set.**
2. Eliminar elemento del **Set.**
3. Verificar si un elemento se encuentra en el **Set.**
4. **TreeSet:** Ordena de menor a mayor.
5. **HashSet:** Desordenado (pseudo-aleatorio).

import java.util.TreeSet;  
  
class SetEjemplo {  
 public static void main (String [] args) {  
 TreeSet<String> strings = new TreeSet<String>();  
 strings.add("Aquiles");  
 strings.add("Temistocles");  
 strings.add("Petronila");  
 System.out.println(strings.size());  
 System.out.println(strings.first());  
 System.out.println(strings.last());  
  
 strings.remove("Temistocles");  
  
 for (String s: strings) {  
 System.out.println(s);  
 }  
 }  
}

## MAP

Tienen la característica de guardar un par de objetos **(clave, valor),** de tal manera que busca la clave para retornar el valor. Es similar a un diccionario.

import java.util.HashMap;;  
  
class Main {  
 public static void main(String[] args) {  
 HashMap<String, String> strings = new HashMap<String, String>();  
 strings.put("Aquiles", "aquilesbailo@mailtp.edu");  
 strings.put("Temistocles", "temis\_99@mailtp.edu");  
 strings.put("Petronila", "Petro.nila@mailtp.edu");  
 System.out.println(strings.size());  
 strings.remove("Aquiles");  
 System.out.println(strings.get("Temistocles"));  
  
 for (String s: strings.keySet()) {  
 System.out.println(s);  
 }  
 for (String s: strings.values()) {  
 System.out.println(s);  
 }  
 }  
}

Revisa la [documentación oficial](https://docs.oracle.com/javase/7/docs/api/) para acceder a la lista completa de clases y métodos que proporciona “java.util”.

## RETOS

|  |
| --- |
| **¡Ya trabajaste en los Retos anteriores!**  **Ahora es tu turno y aplica los pasos efectuados para los siguientes desafíos.**  **Recuerda consultar con el/la docente en caso dudas o comentarios.** |

1. Organizar grupos de pares Forma un grupo de pares con un compañero o compañera (¿tienes problemas de cómo organizar el trabajo? [Click aquí](https://www.youtube.com/watch?v=T5zkEpaMlfw&ab_channel=CristianRizzi)) y atiendan en siguiente desafío: ustedes son creadores de la nueva plataforma de música llamada \_\_\_\_\_\_, y para un lanzamiento exitoso en internet deben administrar los grupos musicales que deseen incorporarse a su negocio. Para esto deberán:
   1. Crear un logo llamativo de la plataforma usando esta [herramienta](https://www.canva.com/).
   2. Crear una interfaz de usuario que permita
      * Agregar grupos musicales.
      * Eliminar grupos musicales.
      * **Bonus 1:** Actualizar datos.
      * **Bonus 2:** ¿Qué otra funcionalidad podría tener el panel de administración de su plataforma?
   3. Debe usar **ArrayList** para la gestión de grupos musicales.
   4. Entregable

**Deberán subir a GitHub su código fuente con un README.txt que detalle la solución del programa realizado y un paso a paso para que el programa funcione correctamente.**

**Deberán subir un video promocional de su plataforma musical.**

**¡Pueden usar TikTok, Instagram o el editor de video que ustedes deseen!**

**¡Usen su creatividad!**

**Insertar link del video en el README.**

**Fecha de entrega: xx/xx/xxxx**

* 1. Consideraciones

**Buena documentación de código fuente a través de comentarios.**

**Realizar commit en Github por cada funcionalidad o ajustes realizados.**

**¡Evitar subir el trabajo a Github de una sola vez!**

**Usar vocabulario y buena gesticulación en el video promocional.**



1. A ti se te encarga desarrollar un diccionario tecnológico incorporando conceptos de la Industria 4.0. Para ello deberán crear un programa que reciba por pantalla un **String** con el siguiente formato:
   1. **Concepto:** Definición.
   2. Deberán dividir el **String** para guardarlo en un **Map** ordenado alfabéticamente.
   3. Además, el programa deberá entregar la definición de un concepto ingresado por pantalla.
      * ¿Qué ocurre si el concepto no existe?
        1. **Bonus 1:** Dar la posibilidad de ingresar la definición de ese concepto.
   4. Definir al menos 10 conceptos.
   5. **Bonus 2:** Desarrollar el diccionario en inglés.
   6. Entregable



* 1. Consideraciones









1. Para este desafío Para este desafío deberán crear grupos de cuatro personas. Cada grupo tendrá un identificador (ej: Grupo 1, Grupo 2, Grupo 3, …, Grupo n).
   1. Deberán investigar sobre uno de los siguientes paquetes que aparecen a continuación.

|  |  |
| --- | --- |
| **ID** | **NOMBRE DEL PAQUETE** |
| **1** | java.awt |
| **2** | java.io |
| **3** | java.security |
| **4** | java.sql |
| **5** | java.util |
| **6** | java.util.logging |
| **7** | java.util.zip |
| **8** | javax.swing |

* 1. Según la asignación de los paquetes realizada en clases, deberán:
     + Definir su propósito.
     + Definir 3 clases\* del paquete incluyendo una descripción y dos métodos a elección (seleccionen uno significativo).
       1. Incorporar líneas de código que ejemplifiquen el uso del paquete (puede ser extraído de Internet).
     + **Bonus 1:** Crear el código que ejemplifique el uso del paquete.
     + **Bonus 2:** Crear un meme para el paquete.
  2. Finalmente deberán presentar los requerimientos, poniendo énfasis en la explicación del código de ejemplo.
     + Se tendrá en consideración el manejo del contenido, material audiovisual, pronunciación, postura, volumen, conclusiones y capacidad de responder preguntas.





## RECURSOS

1. Píldorasinformáticas. Acercamiento a la API Paquetes: <https://www.youtube.com/watch?v=0M6qHYdlqpc&feature=emb_title>
2. Programación ATS. Curso completo de programación en Java: <https://www.youtube.com/watch?v=2ZXiuh0rg3M&list=PLWtYZ2ejMVJkjOuTCzIk61j7XKfpIR74K>
3. TutorialesYa. Curso completo de programación en java: <http://www.tutorialesprogramacionya.com/javaya/>
4. La Geekipedia de Ernesto. Curso de Java desde cero:  
   <https://www.youtube.com/watch?v=L1oMLsiMusQ&list=PLyvsggKtwbLX9LrDnl1-K6QtYo7m0yXWB>
5. Java desde cero:   
   <https://javadesdecero.es/>
6. w3schools:  
   <https://www.w3schools.com/java/default.asp>
7. CodingBat:  
   <https://codingbat.com/java>
8. Learn Java:  
   <https://www.learnjavaonline.org/>
9. CodeGym:  
   <https://codegym.cc/>
10. Programming Skills:  
    <http://www.pskills.org/corejava.jsp>
11. Tutorials Point:  
    <https://www.tutorialspoint.com/java/java_generics.htm>
12. Java Tests:  
    <https://www.studytonight.com/tests/?subject=java>
13. JavatPoint:  
    <https://www.javatpoint.com/core-java-quiz>
14. Recopilación de ejercicios Java POO  
    <https://www.academia.edu/36182478/Recopilaci%C3%B3n_de_Ejercicios_Java_POO_Programaci%C3%B3n_Orientada_a_Objetos>
15. Curso de programación orientada a objetos: <https://www.curriculumnacional.cl/614/w3-article-209545.html>
16. Curso de desarrollo de videojuegos: <https://www.curriculumnacional.cl/614/w3-article-209538.html>
17. Curso de desarrollo de aplicaciones móviles: <https://www.curriculumnacional.cl/614/w3-article-209536.html>
18. makigas. Tutorial de GIT: <https://www.youtube.com/watch?v=jSJ8xhKtfP4&list=PLTd5ehIj0goMCnj6V5NdzSIHBgrIXckGU>

## REFERENCIAS

1. MIT. (2010). Introduction to Programming in Java, Lecture 5: Access Control, Class Scope, Packages, Java Api. Recuperado 20 de agosto de 2020, de <https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-092-introduction-to-programming-in-java-january-iap-2010/lecture-notes/MIT6_092IAP10_lec06.pdf>
2. Java. (2020). Java™ Platform, Standard Edition 7 API Specification. Recuperado 20 de Agosto de 2020, de <https://docs.oracle.com/javase/7/docs/api/>.