Taller de nivelación PI a PII

Objetivos del Taller

- Introducción a markdown.
- Introducción a Git.
- Reforzar los conceptos fundamentales de PI básica en Java.
- Introducir conceptos básicos que se desarrollarán en PII.
- Fomentar la resolución de problemas y el pensamiento algorítmico.

Estructura del Taller

El taller estará dividido en dos partes: teórica y práctica. Adicionalmente se debe entregar las solución de los ejercicios prácticos del taller en un repositorio, siguiendo los pasos descritos en la guía.

Parte Teórica

Investigar que es markdown

GIT:

1. ¿Qué es un repositorio en Git y cómo se diferencia de un proyecto “normal”?
2. ¿Cuáles son las tres áreas principales de Git (working directory, staging area/index y repository) y qué papel cumple cada una?
3. ¿Cómo representa Git los cambios internamente? (objetos blob, tree, commit y tag).
4. ¿Cómo se crea un commit y qué información almacena un objeto commit?
5. ¿Cuál es la diferencia entre git pull y git fetch?
6. ¿Qué es un branch (rama) en Git y cómo Git gestiona los punteros a commits?
7. ¿Cómo se realiza un merge y qué conflictos pueden surgir? ¿Cómo se resuelven?
8. ¿Cómo funciona el área de staging (git add) y qué pasa si omito este paso?
9. ¿Qué es el archivo .gitignore y cómo influye en el seguimiento de archivos?
10. ¿Cuál es la diferencia entre un “commit amend” (--amend) y un nuevo commit?
11. ¿Cómo se utiliza git stash y en qué escenarios es útil?
12. ¿Qué mecanismos ofrece Git para deshacer cambios (por ejemplo, git reset, git revert, git checkout)?
13. ¿Cómo funciona la configuración de remotos (origin, upstream) y qué comandos uso para gestión de forks?
14. ¿Cómo puedo inspeccionar el historial de commits (por ejemplo, `git log`, `git diff`, `git show`)?

Programación:

15. ¿Cuáles son los tipos de datos primitivos en Java?
16. ¿Cómo funcionan las estructuras de control de flujo como if, else, switch y bucles en Java?
17. ¿Por qué es importante usar nombres significativos para variables y métodos?
18. ¿Qué es la Programación Orientada a Objetos (POO)?
19. ¿Cuáles son los cuatro pilares de la Programación Orientada a Objetos?
20. ¿Qué es la herencia en POO y cómo se utiliza en Java?
21. ¿Qué son los modificadores de acceso y cuáles son los más comunes en Java?
22. ¿Qué es una variable de entorno y por qué son importantes para Java o la programación en general?

Parte Práctica

1. Crear un programa que utilice estructuras de control para resolver un problema sencillo, como una calculadora básica que realice operaciones de suma, resta, multiplicación y división.

2. Escribir un programa que cuente el número de vocales y consonantes en una palabra. La palabra no contendrá símbolos, caracteres especiales, acentos, ni números y siempre estará en minúsculas.

3. Escribir un programa que invierta una cadena de texto ingresada por el usuario.

Guía para Entregar Ejercicios en GitHub

Paso 1: Crear una Cuenta en GitHub

1. Visita GitHub.
2. Regístrate para obtener una cuenta gratuita si aún no tienes una.

Paso 2: Instalar Git

1. Descarga e instala Git desde git-scm.com.
2. Configura tu nombre de usuario y correo electrónico en Git:

```
git config --global user.name "Tu Nombre"
git config --global user.email "tuemail@example.com"
```

Paso 3: Crear un Nuevo Repositorio

1. Inicia sesión en tu cuenta de GitHub.
2. Haz clic en el botón **New** en la página principal para crear un nuevo repositorio.
3. Nombra tu repositorio (por ejemplo, "**Ejercicios"**) y añade una descripción breve.
4. Marca la casilla para inicializar el repositorio con un archivo "**README.md"**.
5. Haz clic en **Create repository.**

Paso 4: Clonar el Repositorio en tu Computadora

1. Copia la URL de tu nuevo repositorio desde GitHub.
2. Abre una terminal o consola en tu computadora.
3. Clona el repositorio usando Git:

```
git clone <URL-del-repositorio>
```

Reemplaza **<URL-del-repositorio>** con la URL de tu repositorio de GitHub.

Paso 5: Añadir los Archivos del Proyecto

1. Navega al directorio del repositorio clonado:

```
cd Ejercicios
```

2. Añade tus archivos de código al directorio del repositorio.

Paso 6: Crear el Archivo README.md

1. Abre el archivo **"README.md"** en un editor de texto.
2. Añade información relevante, como el propósito del proyecto, instrucciones de instalación y uso, y cualquier otra información importante.
   Ejemplo de contenido para el **"README.md"**:

```
# Ejercicios de Programación

Este repositorio contiene los ejercicios de programación en Java del taller de nivelación.

## Contenido

- Calculadora Básica
- Contador de Vocales y Consonantes
- Invertir una Cadena

## Instrucciones

1. Clona el repositorio.
2. Navega a cada directorio de ejercicios.
3. Compila y ejecuta los archivos `.java` para probar los ejercicios.
```

Paso 7: Realizar el Commit de los Cambios

1. Añade los archivos al área de preparación (staging):

   ```
   git add .
   ```

2. Realiza un commit de los cambios:

   ```
   git commit -m "Añadir ejercicios de programación y actualizar README"
   ```

Paso 8: Subir los Cambios a GitHub

1. Sube los cambios a tu repositorio en GitHub:

   ```
   git push origin main
   ```

Nota: Si creaste una rama diferente a main, usa el nombre correcto de tu rama.

Paso 9: Compartir el Enlace del Repositorio

1. Copia la URL de tu repositorio desde GitHub.
2. Envía el enlace a tu profesor o al lugar designado para la entrega.

O también puedes seguir este video para mayor facilidad:

https://www.youtube.com/watch?v=q6RKq91FKC4

Recursos:

https://netbeans.apache.org/
https://netbeans.apache.org/tutorial/main/kb/docs/java/