**Comandos Git**

**git init:** Este comando inicializa un nuevo repositorio de Git en tu proyecto. Para hacerlo, solo tienes que ejecutar git init en el directorio raíz de tu proyecto.

**git add:** Este comando agrega los cambios realizados en un archivo a la zona de preparación (staging area) de Git. Para hacerlo, ejecuta git add [nombre\_del\_archivo] o git add . para agregar todos los archivos modificados.

**git commit:** Este comando guarda los cambios en la zona de preparación en el historial de Git. Para hacerlo, ejecuta git commit -m "[mensaje\_de\_commit]". El mensaje de commit debe ser una descripción breve de los cambios realizados.

**git push:** Este comando sube los cambios realizados a un repositorio remoto en Git. Para hacerlo, ejecuta git push [nombre\_del\_repositorio] [rama]. El nombre del repositorio y la rama deben ser los correspondientes a tu proyecto.

**git pull:** Este comando descarga los cambios realizados en un repositorio remoto a tu repositorio local. Para hacerlo, ejecuta git pull [nombre\_del\_repositorio] [rama].

**git status:** Este comando muestra el estado actual de tu repositorio, incluyendo los archivos modificados, eliminados o agregados. Para hacerlo, ejecuta git status.

**git log:** Este comando muestra el historial de commits de tu repositorio. Para hacerlo, ejecuta git log.

**git branch:** Este comando muestra las ramas de tu repositorio. Para hacerlo, ejecuta git branch.

**git checkout:** Este comando cambia de rama o de commit en tu repositorio. Para hacerlo, ejecuta git checkout [rama\_o\_commit].

**git merge:** Este comando fusiona los cambios realizados en una rama a otra rama. Para hacerlo, primero cambia a la rama de destino con git checkout [rama\_destino] y luego ejecuta git merge [rama\_origen].

**git clone:** Este comando crea una copia de un repositorio remoto en tu computadora. Para hacerlo, ejecuta git clone [url\_del\_repositorio].

**git pull --rebase:** Este comando descarga los cambios realizados en un repositorio remoto y los aplica por encima de tus cambios locales. Para hacerlo, ejecuta git pull --rebase.

**git stash:** Este comando guarda temporalmente los cambios realizados en tus archivos sin hacer commit. Para hacerlo, ejecuta git stash.

**git reset:** Este comando deshace los cambios realizados en un archivo y lo regresa al último commit. Para hacerlo, ejecuta git reset [nombre\_del\_archivo].

**git remote:** Este comando muestra los repositorios remotos asociados a tu repositorio local. Para hacerlo, ejecuta git remote.

**Comandos JAVA**

**Ctrl. + E :** Borra línea de codigo donde estoy posicionado.

**Ctrl. + S :** Guarda los cambios en la clase que estoy posicionado.

**Shift. + F6 :** RUN main proyect (actual).

**Alt. + Shift + F :** Le da formato al código.(queda mas legible, ordenado)

**Alt + Enter :** Se usa sobre el texto subrayado para ver la sugerencia que nos hace el IDE cuando hay

algún error de sintaxis o se necesita realizar algún import o surround.

**Ctrl. + Shift + :** Hace una copia de la línea de código que estabas posicionado.

**Shift + DELETE :** Borra toda la línea de código donde estoy posicionada.

**Ctrl. + Space :** Completa el código que estamos escribiendo.

**Ctrl. + U U :** Convierte el código a mayúsculas.

**Ctrl. + U L :** Convierte el código a minúsculas.

**Alt + Insert :** Agrega constructores, métodos accesotes, propiedades.(codigos predeterminados)

**Ctrl. + G :** Ir numero de línea concreto(introducida x ventana auxiliar).

**Ctrl. +[NOMBRECLASE] :** Me despliega todas las características de la clase

**if. + TAB :** Genera los bloques if.

**sout. + TAB :** Genera los System.out.println

**fori + TAB :** Genera el codigo necesario para hacer un for.

**sw + TAB :** Genera el switch.

**whilexp/whilen/whileit + TAB :** Genera los bloques while.

**dowhile + TAB :** Genera los bloques dowhile.

**PSVM + TAB :** Genera la función main.

System.out.print(“”) : (sin el in) escribe sin saltar.