Cuestiones teóricas a responder:

1. Explica qué son los ***sistemas de control de versiones***, cuáles son sus principales funciones.

-Son herramientas que gestionan los cambios en archivos permitiendo la colaboración, seguimiento de versiones y resolución de conflictos

2. Explica las diferencias entre un ***sistema local***, un ***sistema centralizado*** y un ***sistema distribuido*** de control de versiones.

-Sistema local: El control de versiones ocurre de manera local en el equipo del usuario
-Sistema centralizado: Mantiene un repositorio central en un servidor, y los usuarios descargan y suben cambios
-Sistema distribuido: Cada usuario tiene una copia del repositorio en su máquina, lo que le permite una independencia para trabajar

3. Explica qué es ***Git*** y cuáles son sus principales características.

-Git es un sistema de control de versiones
-Sus características son que es distribuido, rápido y eficiente que permite el trabajo con ramas y tiene un historial de cambios detallado

4. Define ***Repositorio*** en el contexto de los sistemas de control de versiones.

-Un repositorio es un lugar que almacena los archivos y el historial de cambios de un fichero

5. Un repositorio local de Git funciona con 3 “árboles” Working Tree, Stage Area y HEAD. Explica la dinámica de añadir (add) y confirmar (commit) ficheros o cambios al repositorio en Git.

-Working Tree: Es el espacio de trabajo local en donde los archivos estan descomprimidos y listos para ser editados
-Stage Area: Es el almacenamiento intermedio usado para el procesamiento de datos
-HEAD: Una referencia a la confirmación actual en la rama que se encuentra actualmente en el repositorio

-Realizas git add para agregar cambios al Stage Area y usas git commit para confirmar los cambios realizados

6. Busca información y explica cuál es el propósito del fichero ***.gitignore*** en un repositorio.

-Gitignore se utiliza para especificar que archivos o directorios deben ser ignorados por git

7. Explica detalladamente las opciones para deshacer cambios ***Revert***, ***Soft Reset*** y ***Hard Reset***.

-Revert: Crea un nuevo commit que deshace los cambios del anterior
-Soft Reset: Mueve el puntero de HEAD pero mantiene los cambios sin confirmar
-Hard Reset: Mueve el puntero HEAD y borra los cambios locales

8. Define ***etiqueta***, ***tag*** o ***label***, en el contexto de los sistemas de control de versiones.

-Una tag o etiqueta sirve básicamente como una rama firmada que no permuta, es decir, siempre se mantiene inalterable

9. Define ***rama***, ***branch***, en el contexto de los sistemas de control de versiones.

-Una rama o branch representa una línea independiente de desarrollo, es como crear un nuevo área de trabajo con su área de ensayo y su historial

10. Define ***fusión***, ***merge***, en el contexto de los sistemas de control de versiones.
-Fusión o merge permite tomar las líneas independientes de desarrollo creadas por git branch e integrarlas en una sola rama

11. Explica las diferencias entre ***Merge*** y ***Rebase*** para integrar cambios de una rama a otra usando Git.
-El rebase unifica las ramas dejando un árbol lineal o más bonito
-El merge aun deja el gráfico de las ramas

12. Explica en qué consiste el mecanismo ***Cherry Pick*** para incorporar cambios de una rama a otra.
-Cherry pick es un potente comando que permite que las confirmaciones arbitrarias de Git se elijan por referencia y se añadan al actual HEAD de trabajo