**Subtarea 1: Uso de GitHub y Git**

Tras haber creado nuestro perfil, y clonado el repositorio al nuestro local mediante el comando *git clone:*

5. Ejecutamos el comando *ls -a*, para mostrar todo el contenido del repositorio, incluyendo los archivos ocultos y comprobamos que está vacío.

Texto

Descripción generada automáticamente

6. Ejecutamos el comando *git status* para comprobar el estado del repositorio, al no contener ningún archivo lógicamente está “al día”.

Texto

Descripción generada automáticamente

7. Pegamos el trabajo en la carpeta, y ejecutamos de nuevo el comando *git status*. Podemos comprobar que ahora en rojo aparece el archivo copiado y así nos indica que no se está rastreando.

Texto

Descripción generada automáticamente

8. Usando el comando *git add*, adjuntamos al rastreo el archivo de la tarea, y mediante *git status* comprobamos que estaría ya preparado para un commit, es decir, en fase *staged.*

Texto

Descripción generada automáticamente

9. Después, con el comando *git commit -m*, realizamos un commit de todos los archivos que tengamos, en este caso sólo el .doc de la tarea.

Texto

Descripción generada automáticamente

Tras usar *git status*, comprobamos que no tiene nada para hacer commit, y avisa que la rama se encuentra limpia, el repositorio local está al día.

Texto

Descripción generada automáticamente

10. Finalmente, para poder sincronizar todo lo que tenemos en el commit con nuestro repositorio de GitHub tenemos que utilizar el comando *git push*, y así subiríamos todo lo que tenemos en local.

Texto

Descripción generada automáticamente

**Subtarea 2: Comparación de dos IDEs o editores de texto**

**Notepad++:**

* Lenguajes de programación compatibles:

Soporta una gran variedad de lenguajes de programación, alrededor de unos 70, igualmente permite crear diferentes esquemas de color por si fuesen necesarios para poder utilizar y visualizar perfectamente otros tipos de lenguajes.

* Facilidad de uso:

Dispone de una interfaz simple y sencilla, para poder ser usada por cualquier tipo de usuario, sin importar sus conocimientos. Además, usa pestañas para facilitar la navegación entre documentos, los menús son fáciles de navegar y está todo perfectamente etiquetado.

* Depuración:

No tiene un depurador propio, depende de otras herramientas externas para ello.

* Gestión de versiones:

No dispone de ningún tipo de soporte integrado para la gestión de versiones, pero tiene la opción de añadir complementos externos para ello.

* Personalización y extensibilidad:

Es suficientemente personalizable dentro de lo básico que es, mediante códigos de color, modificaciones en las preferencias de edición… Se adecúa bastante a las necesidades de un programador novato.

* Otros aspectos que te resulten de interés:
  + Sólo está disponible para Windows.
  + Se puede ejecutar desde una unidad USB.
  + Está programado en C++.

**Visual Studio Code:**

* Lenguajes de programación compatibles:

Soporta aun más lenguajes que Notepad++, incluyendo aquellos que son más específicos científicamente.

* Facilidad de uso:

Es relativamente mas complejo que Notepad++, no está enfocado a usuarios principiantes, pero igualmente su interfaz no es muy dificil de aprender. En comparativa, dispone de muchas más opciones para modificar según las necesidades del usuario.

* Depuración:

Tiene herramienta de depuración integrada: puede poner puntos de interrupción, inspeccionar variables, realizar seguimiento de la ejecución del código…

* Gestión de versiones:

Tiene integración directa para el control de versiones con Git y GitHub.

* Personalización y extensibilidad:

Dispone de muchas opciones de personalización, tanto propias como a través de extensiones de terceros que le permiten añadir incluso nuevas funciones.

* Otros aspectos que te resulten de interés:
  + Desarrollado por Microsoft
  + Disponible para Windows, MacOS y Linux.
  + Si no es desactivado, recopila datos que se envían a Microsoft

**Subtarea 3: Tu herramienta favorita**

Herramientas L-Case

Dentro de las herramientas CASE, creo que las que me serían de mayor utilidad son las L-Case principalmente porque son las que se acercan más a mi ámbito de trabajo, obviando el caso que tuviese que dedicarme a un enfoque más de diseño y análisis, que de primeras no es algo que me resulte muy llamativo.

Dentro de las L-Case, seguramente me apoyase bastante en aquellas centradas en el mantenimiento y la gestión de pruebas, principalmente porque en el desarrollo de cualquier proyecto, y aun siendo todas importantes, son fases extremadamente cruciales, y en las que por despiste podrían surgir fallos severos, que de no ser por esas herramientas pasasen desapercibidos y pudiesen ocasionar problemas.

Gracias a estas herramientas, podrían automatizarse y crear programas de detección de errores, así como para la depuración de los proyectos, y además algunas de ellas incluso me generarían la documentación, parte de los programas que en ocasiones debido a la urgencia que se pueda tener puedes olvidarte de crear.