**C**

C es un lenguaje de programación creado en 1972 por Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL.   
  
Al igual que B, es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.   
  
Se trata de un lenguaje débilmente tipificado de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos.   
  
La primera estandarización del lenguaje C fue en ANSI, con el estándar X3.159-1989. El lenguaje que define este estándar fue conocido vulgarmente como ANSI C. Posteriormente, en 1990, fue ratificado como estándar ISO (ISO/IEC 9899:1990). La adopción de este estándar es muy amplia por lo que, si los programas creados lo siguen, el código es portátil entre plataformas y/o arquitecturas.

**C++**

C++ es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.   
  
Posteriormente se añadieron facilidades de programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje de programación multiparadigma.   
  
Actualmente existe un estándar, denominado ISO C++, al que se han adherido la mayoría de los fabricantes de compiladores más modernos. Existen también algunos intérpretes, tales como ROOT.   
  
Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales.   
  
El nombre C++ fue propuesto por Rick Mascitti en el año 1983, cuando el lenguaje fue utilizado por primera vez fuera de un laboratorio científico. Antes se había usado el nombre "C con clases". En C++, la expresión "C++" significa "incremento de C" y se refiere a que C++ es una extensión de C.

**C#**

C# (pronunciado si sharp en inglés) es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común.   
  
Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes.   
  
El nombre C Sharp fue inspirado por la notación musical, donde ♯ (sostenido, en inglés sharp) indica que la nota (C es la nota do en inglés) es un semitono más alta, sugiriendo que C# es superior a C/C++. Además, el signo '#' viene de cuatro '+' pegados.1   
  
Aunque C# forma parte de la plataforma .NET, ésta es una API, mientras que C# es un lenguaje de programación independiente diseñado para generar programas sobre dicha plataforma. Ya existe un compilador implementado que provee el marco Mono - DotGNU, el cual genera programas para distintas plataformas como Windows, Unix y GNU/Linux.

**Que es Git**

Es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente.

**Qué es Github**

Básicamente Github es un servicio de alojamiento de repositorios de software que ofrece opciones muy interesantes para trabajar en equipo. GitHub está basado en Git, un sistema de control de versiones muy popular entre los desarrolladores.

**Principales comandos Git y GitHub.**

**a) Buscando Ayuda:**

1. **git help** comando ó **git** comando --help

    Muestra la ayuda para ese comando

**b) Creación de un repositorio:**

2. **git init**

    Crea un repositorio en el directorio actual

3. **git clone** url

    Clona un repositorio remoto dentro de un directorio

**c) Operaciones sobre Archivos:**

4. **git add** path

    Adiciona un archivo o un directorio de manera recursiva

5. **git rm** ruta

    Remueve un archivo o directorio del árbol de trabajo

      -f : Fuerza la eliminación de un archivo del repositorio

6. **git mv** origen destino

    Mueve el archivo o directorio a una nueva ruta

      -f : Sobre-escribe los archivos existentes en la ruta destino

7. **git checkout** [rev] archivo

    Recupera un archivo desde la rama o revisión actual

      -f : Sobre-escribe los cambios locales no guardados

**d) Trabajando sobre el código:**

8. **git status**

    Imprime un reporte del estado actual del árbol de trabajo local

9. **git diff** [ruta]

    Muestra la diferencia entre los cambios en el árbol de trabajo local

10. **git diff** HEAD ruta

    Muestra las diferencias entre los cambios registrados y los no registrados

11. **git add** path

    Selecciona el archivo para que sea incluido en el próximo commit

12. **git reset** HEAD ruta

    Marca el archivo para que no sea incluido en el próximo commit

13. **git commit**

    Realiza el commit de los archivos que han sido registrados (con git-add)

      -a : Automáticamente registra todos los archivos modificados

14. **git reset** --soft HEAD^

    Deshace commit & conserva los cambios en el árbol de trabajo local

15. **git reset** --hard HEAD^

    Restablece el árbol de trabajo local a la versión del ultimo commit

16. **git clean**

    Elimina archivos desconocidos del árbol de trabajo local

**e) Examinando el histórico:**

17. **git log** [ruta]

    Muestra el log del commit, opcionalmente de la ruta especifica

18. **git log** [desde [..hasta]]

    Muestra el log del commit para un rango de revisiones dado

      --stat : Lista el reporte de diferencias de cada revisión

      -S'pattern' : Busca el historial de cambios que concuerden con el patrón de búsqueda

19. **git blame** [archivo]

    Muestra el archivo relacionado con las modificaciones realizadas

**f) Repositorios remotos:**

20. **git fetch** [remote]

    Trae los cambios desde un repositorio remoto

21. **git pull** [remote]

    Descarga y guarda los cambios realizados desde un repositorio remoto

22. **git push** [remote]

    Guarda los cambios en un repositorio remoto

23. **git remote**

    Lista los repositorios remotos

24. **git remote add** remote url

    Añade un repositorio remoto a la lista de repositorios registrados

**g) Ramas:**

25. **git checkout** rama

    Cambia el árbol de trabajo local a la rama indicada

      -b rama : Crea la rama antes de cambiar el árbol de trabajo local a dicha rama

26. **git branch**

    Lista las ramas locales

27. **git branch** -f rama rev

    Sobre-escribe la rama existente y comienza desde la revisión

28. **git merge** rama

    Guarda los cambios desde la rama

**h) Exportando e importando:**

29. **git apply** - < archivo

    Aplica el parche desde consola (stdin)

30. **git format-patch** desde [..hasta]

    Formatea un parche con un mensaje de log y un reporte de diferencias (diffstat)

31. **git archive** rev > archivo

    Exporta resumen de la revisión (snapshot) a un archivo

      --prefix=dir/ : Anida todos los archivos del snapshot en el directorio

      --format=[tar|zip] : Especifica el formato de archivo a utilizar: tar or zip

**i) Etiquetas:**

32. **git tag** name [revision]

    Crea una etiqueta para la revisión referida

      -s : Firma la etiqueta con su llave privada usando GPG

      -l [patrón] : Imprime etiquetas y opcionalmente los registros que concuerden con el patrón de busqueda

**j) Banderas de Estado de los Archivos:**

**M** (modified) : El archivo ha sido modificado

**C** (copy-edit) : El archivo ha sido copiado y modificado

**R** (rename-edit) : El archivo ha sido renombrado y modificado

**A** (added) : El archivo ha sido añadido

**D** (deleted) : El archivo ha sido eliminado

**U** (unmerged) : El archivo presenta conflictos después de ser guardado en el servidor (merge)