C es un lenguaje libre estandarizado por ISO muy pequeño que admite programación estructurada (la de toda la vida), nada más. Útil en programación de microchips, sistemas operativos, drivers y programación web... Poco más.  
  
C++ es un lenguaje libre estandarizado por ISO muy grande, que admite: Programación estructurada (la de toda la vida), la totalidad de la POO (objetos, herencia simple, herencia múltiple, polimorfismo, upcasting, downcasting, RTTI, interfaces, clases abstractas, clases amigas, operadores, sobrecarga... y mil cosas más avanzadas), programación genérica (plantillas, STL, conceptos de contenedores, estructuras de datos genéricas, <b>metaprogramación</b>... es una programación que no dispone ningún otro lenguaje mayoritario por el momento)... y otras características más avanzadas como los punteros inteligentes, programación lambda, programación "física"... También se usa como programación por eventos (MFC, Qt, Gtk y otras GUIs). Se usa para cualquier cosa, desde sistemas operativos hasta los juegos 3D de última generación pasando por servidores, pasando por las típicas aplicaciones de escritorio o un Office o OpenOffice, un reproductor WinAMP y Windows Media Player, un explorador web (Internet Explorer, Mozilla, Firefox...), un cliente eMule, un cliente Torrent.  
  
C# es un invento de Microsoft (lenguaje propietario) que mezcla las características básicas de C++ (no las avanzadas) simplificándolas al estilo Java y ofreciendo un framework. El problema es que es .Net, y deja de ser código nativo/portable. Eso sí, el framework provee bastante facilidad de programación de tareas comunes, al igual que Java. Por ello se genera el debate ¿Java o C#? Su funcionalidad viene a ser parecida. .Net es más nativo y Java más virtual.

Git.

Git es un sistema de control de versiones distribuido. Git fue impulsado por Linus Torvalds y el equipo de desarrollo del Kernel de Linux. Ellos estaban usando otro sistema de control de versiones de código abierto, que ya por aquel entonces era distribuido. Todo iba bien hasta que los gestores de aquel sistema de control de versiones lo convirtieron en un software propietario. Lógicamente, no era compatible estar construyendo un sistema de código abierto, tan representativo como el núcleo de Linux, y estar pagando por usar un sistema de control de versiones propietario. Por ello, el mismo equipo de desarrollo del Kernel de Linux se tomó la tarea de construir desde cero un sistema de versionado de software, también distribuido, que aportase lo mejor de los sistemas existentes hasta el momento.

Así nació Git, un sistema de control de versiones de código abierto, relativamente nuevo que nos ofrece las mejores características en la actualidad, pero sin perder la sencillez y que a partir de entonces no ha parado de crecer y de ser usado por más desarrolladores en el mundo. A los programadores nos ha ayudado a ser más eficientes en nuestro trabajo, ya que ha universalizado las herramientas de control de versiones del software que hasta entonces no estaban tan popularizadas y tan al alcance del común de los desarrolladores.

Git es algo más general que nos sirve para controlar el estado de un desarrollo a lo largo del tiempo, mientras que Github es algo más particular: un sitio web que usa Git para ofrecer a la comunidad de desarrolladores repositorios de software.

Git es multiplataforma, por lo que puedes usarlo y crear repositorios locales en todos los sistemas operativos más comunes, Windows, Linux o Mac. Existen multitud de GUIs (Graphical User Interface o Interfaz de Usuario Gráfica) para trabajar con Git a golpe de ratón, no obstante para el aprendizaje se recomienda usarlo con línea de comandos, de modo que puedas dominar el sistema desde su base, en lugar de estar aprendiendo a usar un programa determinado.

**¿Qué nos aporta GIT?**

* Auditoría del código: saber quién ha tocado qué y cuándo
* Control sobre cómo ha cambiado nuestro proyecto con el paso del tiempo
* Volver hacia atrás de una forma rápida
* Control de versiones a través de etiquetas: versión 1.0, versión 1.0.1, versión 1.1, etc. Sabremos exactamente que había en cada una de ellas y las diferencias entre cualquiera de ellas dos
* Seguridad: todas las estructuras internas de datos están firmadas con SHA1. No se puede cambiar el código sin que nos enteremos
* Mejora nuestra capacidad de trabajar en equipo
* Merging y branching extremadamente eficientes

Realización de branch

La creación de un branch en git es bastante sencillo, a continuación se muestran los comandos necesarios para crear y cambiar el branch de trabajo.

  $ git branch (nombre-del-nuevo-branch)

$ git checkout (nuevo branch)

Con esto se crea un branch con el nombre que se le quiera dar y con el checkout nos movemos a él (el working directory queda apuntando a este branch para poder trabajar con él).

El nuevo branch se crea a partir del branch en el que nos encontramos, por lo que ahora mismo el *master* y el*nuevo-branch* tendrían los mismos cambios. A partir de este momento podemos trabajar en cada branch de forma independiente.

Un comando equivalente, donde se hacen las dos operaciones (crear y moverse al branch) en una sería:

    $ git checkout -b nuevo-branch

Utilización de servidor/master

Para poder colaborar, se necesita tener un repositorio remoto de Git. Aunque técnicamente es posible enviar (push) y recibir (pull) cambios directamente a o desde repositorios individuales, no es muy recomendable trabajar de esa manera por la gran facilidad para confundirse si no se maneja con extremo cuidado. Si se desea que los colaboradores puedan acceder al repositorio, incluso cuando el ordenador se encuentre apagado, puede ser de gran utilidad disponer de un repositorio común fiable. En este sentido, el método más recomendable para colaborar con otra persona es preparar un repositorio intermedio donde todos tengan acceso, enviando (push) y recibiendo (pull) a o desde allí. A este repositorio se le conoce como "servidor Git".

Al mantener éste repositorio actualizado se puede acceder a la ultima versión del código desde donde sea y cuando sea.

GITHUB

Github [github.com](https://github.com/) es un servicio para alojamiento de repositorios de software gestionados por el sistema de control de versiones Git. Por tanto, Git es algo más general que sirve para controlar el estado de un desarrollo a lo largo del tiempo, mientras que Github es algo más particular: un sitio web que usa Git para ofrecer a la comunidad de desarrolladores repositorios de software. Github es un sitio web pensado para hacer posible el compartir el código de una manera más fácil y al mismo tiempo darle popularidad a la herramienta de control de versiones en sí, que es Git.

Cabe destacar que Github es un proyecto comercial, a diferencia de la herramienta Git que es un proyecto de código abierto. No es el único sitio en Internet que mantiene ese modelo de negocio, pues existen otros sitios populares como Bitbucket que tienen la misma fórmula. No obstante, aunque Github tenga inversores que inyectan capital y esté movido por la rentabilidad económica, en el fondo es una iniciativa que siempre ha perseguido (y conseguido) el objetivo de hacer más popular el software libre. En ese sentido, en Github es gratuito alojar proyectos *Open Source*, lo que ha posibilitado que el número de proyectos no pare de crecer, y en estos momentos haya varios millones de repositorios y usuarios trabajando con la herramienta.

Además, al ser Git un sistema de control de versiones distribuido, no se necesita Github u otro sitio de alojamiento del código para usar Git. Simplemente con tener Git instalado en el ordenador, se tiene un sistema de control de versiones completo, perfectamente funcional, para hacer todas las operaciones que sean necesarias para el control de versiones. Claro que usar Github permite muchas facilidades, sobre todo a la hora de compartir código fuente, ya que puede ser compartido a cualquier parte del mundo.

Esa facilidad para compartir código del repositorio alojado en la nube con Gitgub y la misma sencillez que ofrece el sistema de control de versiones Git para trabajar, ha permitido que muchos proyectos Open Source se hayan pasado a Github como repositorio y a partir de ahí hayan comenzado a recibir muchas más contribuciones en su código.