

**TSU Software, Materia: Programación, Clave 50086, Semestre 17-P**  
**Practica 01. Compilar, Ejecutar y Comitear.**

## 1. Objetivos

Los objetivos buscados en esta practica son los siguientes:

- Compilación y Ejecución de un programa básico en el lenguaje de programación java.
- Conocer el funcionamiento básico del software de control de versiones Git.
- Aprender el funcionamiento básico de la plataforma github.com

*\*Las imágenes, aplicación y comandos usados en esta practica, corresponden al sistema operativo Windows 7 X64.*

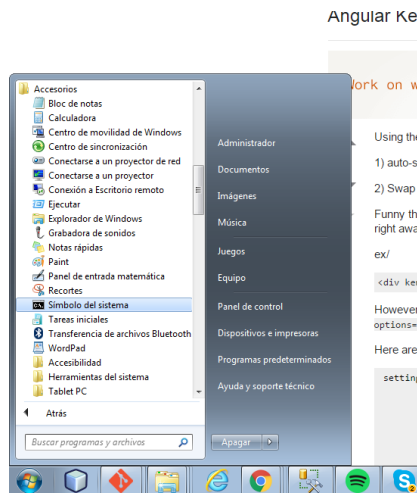
## 2. Compilación y Ejecución

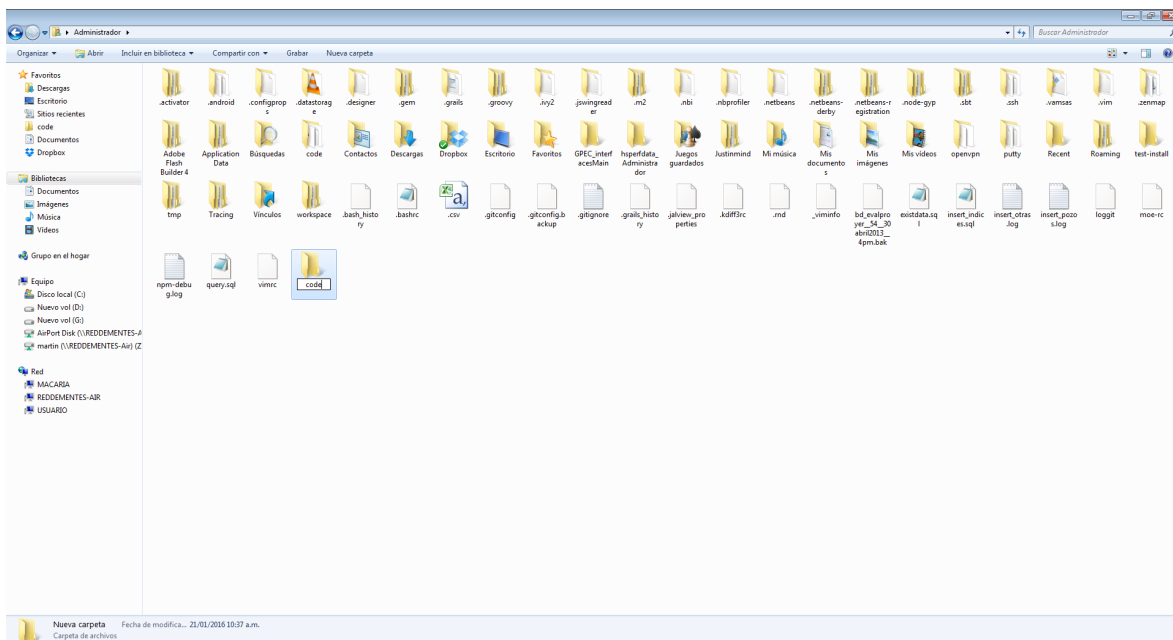
### 2.1 Instalación del JDK

Una vez que nosotros tenemos instalado el jdk (Java SE Development Kit), [pueden seguir esta guía para instalarlo <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>]

Debemos comprobar que se ha instalado correctamente, para esto abrimos la Línea de Comandos de Windows (command prompt, cmd), y tecleamos el comando:

*java -version*

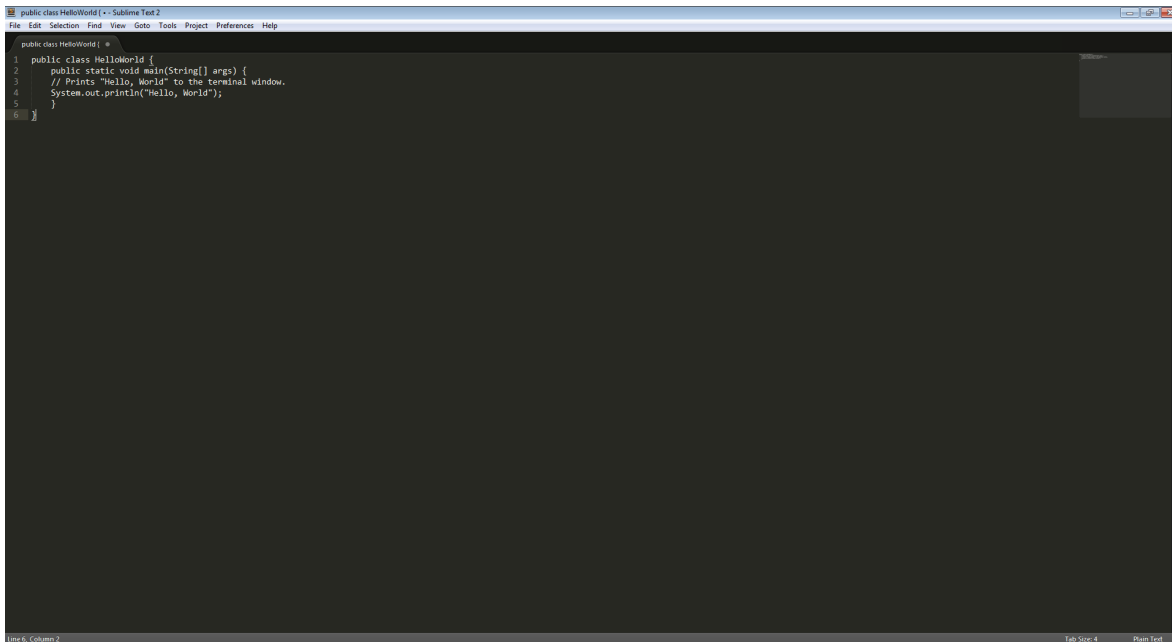




Estos pasos nos crearan una carpeta en la siguiente dirección:  
`C:\Users\Administrador\code`

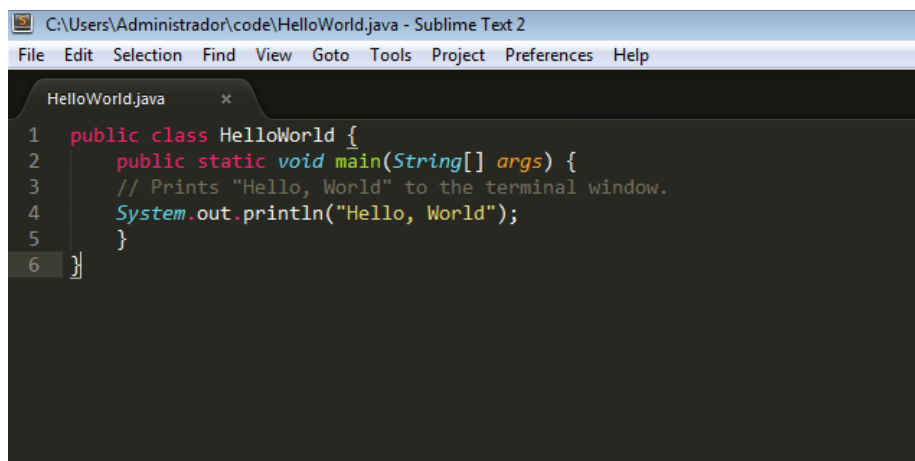
Ahora abrimos *sublime* (Sublime Text 2) y vamos a pegar el siguiente código escrito en java:

```
public class HelloWorld {  
    public static void main(String[] args) {  
        // Prints "Hello, World" to the terminal window.  
        System.out.println("Hello, World");  
    }  
}
```



Y guardaremos este archivo en nuestra carpeta que creamos en pasos anteriores:

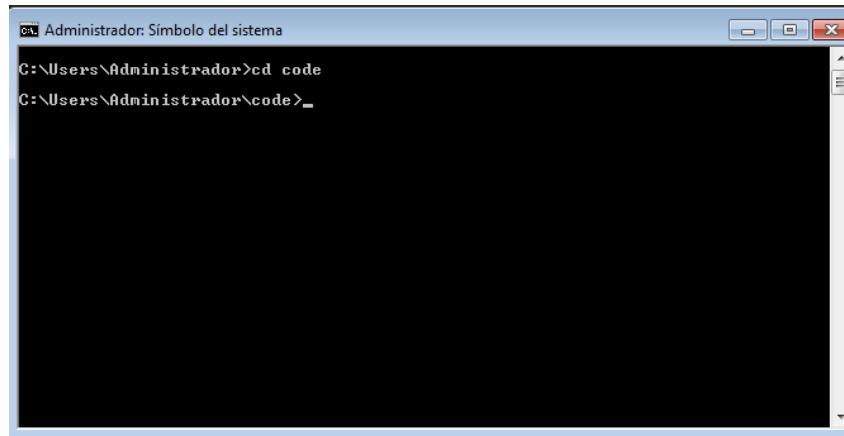
Con el nombre de archivo `HelloWorld.java`; Al guardar el archivo con extensión `.java`, Sublime Text da color a la sintaxis del lenguaje.



## 2.3 Compilar

Ahora que ya tenemos instalado el jdk, y hemos editado nuestro primer programa en java, necesitamos compilar nuestro código `.java` para obtener el **ByteCode**, el caso especial de java lo genera con extensión **`.class`**.

Para lograr esto abrimos la línea de comandos:



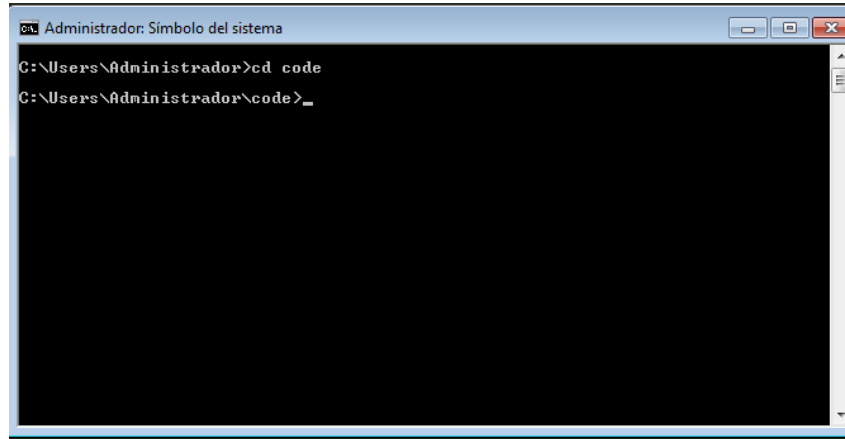
```
Administrador: Símbolo del sistema
C:\Users\Administrador>cd code
C:\Users\Administrador\code>_
```

Como se muestra en la imagen, la carpeta en donde se abre el programa es en la carpeta del Usuario; para poder llegar a nuestro archivo `HelloWorld.java` tenemos que navegar hasta su navegación, los comandos que ocuparemos serán

Command's Purpose	MS-DOS	Linux	Basic Linux Example
Crea una carpeta	<code>mkdir</code>	<code>mkdir</code>	<code>mkdir directory</code>
Renombra un archivo	<code>ren</code>	<code>mv([c])</code>	<code>mv thisfile.txt thatfile.txt</code>
Muestra la ubicación en la que estamos	<code>chdir</code>	<code>pwd</code>	<code>pwd</code>
Cambia de carpeta según la especificación de la dirección	<code>cd pathname</code>	<code>cd pathname</code>	<code>cd /directory/directory</code>
Regresa al nivel superior (botón atrás )	<code>cd..</code>	<code>cd ..</code>	<code>cd ..</code>

Para ir a la carpeta *code* que creamos en pasos anteriores, tenemos que navegar hacia ella, lo que haremos será teclear

*cd code*

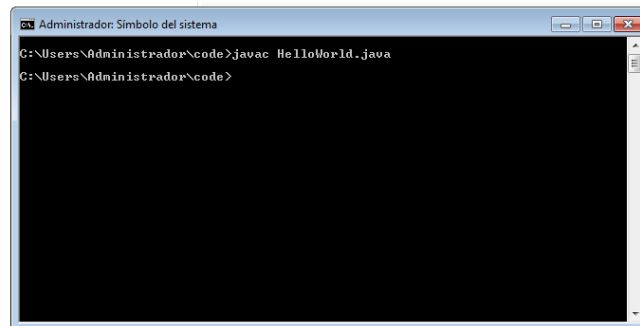


```
Administrador: Símbolo del sistema
C:\Users\Administrador>cd code
C:\Users\Administrador\code>_
```

y como podemos ver en la imagen nos hemos cambiado a la carpeta que se localiza en *C:\Users\Administrador\code* \* tome nota que en sus computadoras no aparecerá *Administrador*, si no, el nombre de usuario que tengan registrado.

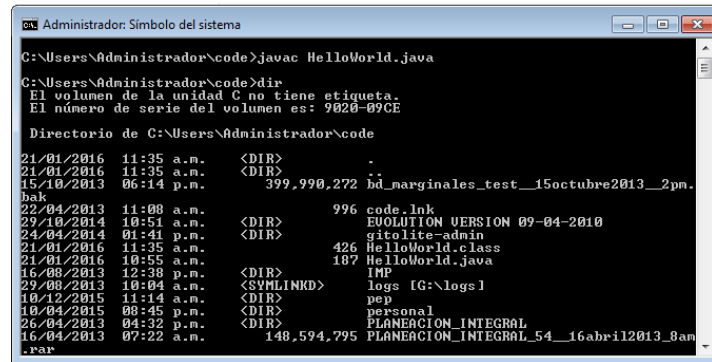
Para compilar nuestro archivo tenemos que usar el programa *javac* de la siguiente manera:

*javac HelloWorld.java*



```
Administrador: Símbolo del sistema
C:\Users\Administrador\code>javac HelloWorld.java
C:\Users\Administrador\code>
```

Esto nos dará como resultado la compilación de nuestro programa y el resultado será la creación del archivo *HelloWorld.class* si quieren confirmar que se a compilado su programa usen el comando ***dir***, para listar los archivos en ese directorio y el resultado será el siguiente:



```
Administrador: Símbolo del sistema
C:\Users\Administrador\code>javac HelloWorld.java
C:\Users\Administrador\code>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 9020-09CE

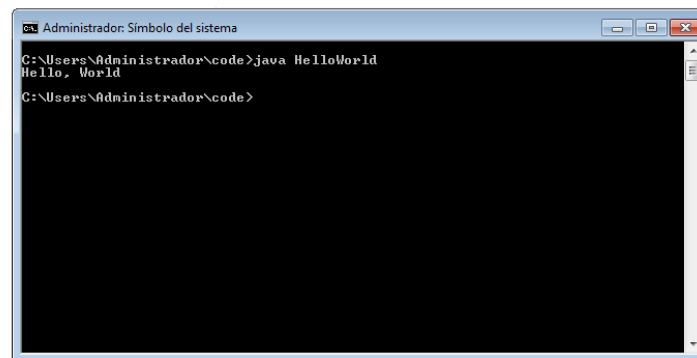
Directorio de C:\Users\Administrador\code

21/01/2016 11:35 a.m. <DIR> .
21/01/2016 11:35 a.m. <DIR> ..
15/10/2013 06:14 p.m. 399,990,272 bd_marginales_test_15octubre2013_2pm.
bak
22/04/2013 11:08 a.m. 996 code.lnk
29/10/2014 10:51 a.m. <DIR> EVOLUTION VERSION 09-04-2010
24/04/2014 01:41 p.m. <DIR> gitolite-admin
21/01/2016 11:35 a.m. 426 HelloWorld.class
21/01/2016 10:55 a.m. 187 HelloWorld.java
16/08/2013 12:38 p.m. <DIR> IMP
29/08/2013 10:04 a.m. <SYMLINKD> logs [G:\logs]
10/12/2015 11:14 a.m. <DIR> pep
10/04/2015 00:45 p.m. <DIR> personal
26/04/2013 04:32 p.m. <DIR> PLANEACION_INTEGRAL
16/04/2013 07:22 a.m. 148,594,795 PLANEACION_INTEGRAL_54_16abril2013_Ban
.rar
```

## 2.4 Ejecución

Para ejecutar nuestro programa compilar basta con teclear el siguiente comando:

```
java HelloWorld
```



```
Administrador: Símbolo del sistema
C:\Users\Administrador\code>java HelloWorld
Hello, World
C:\Users\Administrador\code>
```

## 3. Conceptos básicos sobre git

Git es una herramienta de control de versiones de código desarrollado por Linus Torvalds, creador del kernel Linux, el conocimiento de esta herramienta es requisito necesario en la mayoría de las ofertas laborales.

Básicamente git te permite 4 cosas fundamentales:

1. Descargar un repositorio remoto `git clone`
2. Descargas cambios de un repositorio `git pull`
3. Guardar cambios de tu código `git commit -am "que es lo que guardo"`
4. Subir cambios de tu código `git push`

<b>Check out a repository</b>	Create a working copy of a local repository:	<code>git clone /path/to/repository</code>
	For a remote server, use:	<code>git clone username@host:/path/to/repository</code>
<b>Commit</b>	Commit changes to head (but not yet to the remote repository):	<code>git commit -m "Commit message"</code>
	Commit any files you've added with <code>git add</code> , and also commit any files you've changed since then:	<code>git commit -a</code>
<b>Push</b>	Send changes to the master branch of your remote repository:	<code>git push origin master</code>
<b>Status</b>	List the files you've changed and those you still need to add or commit:	<code>git status</code>
<b>Update from the remote repository</b>	Fetch and merge changes on the remote server to your working directory:	<code>git pull</code>

#### 4. Conceptos básicos sobre github.com

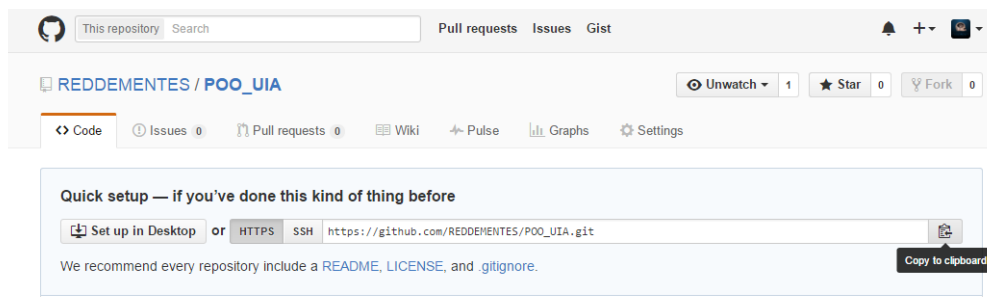
Suponiendo que ya tienen instalado y configurada la herramienta git, de no ser el caso seguir la guía <https://help.github.com/articles/set-up-git/>.

Los comando que se listaron en la sección anterior servirán en la aplicación llamada git bash, que es un emulador de línea de comandos Linux.

## 4.1 Clonar

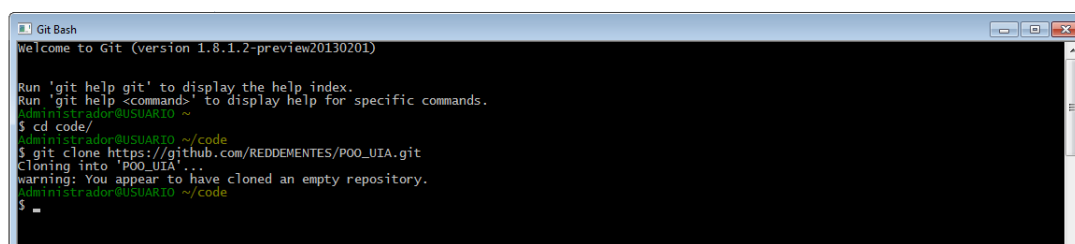
Clonemos el repositorio que se creo durante la clase:

Desde su perfil de github diríjase a su repositorio y en los botones de clic donde dice https y copie la dirección que le proporciona:



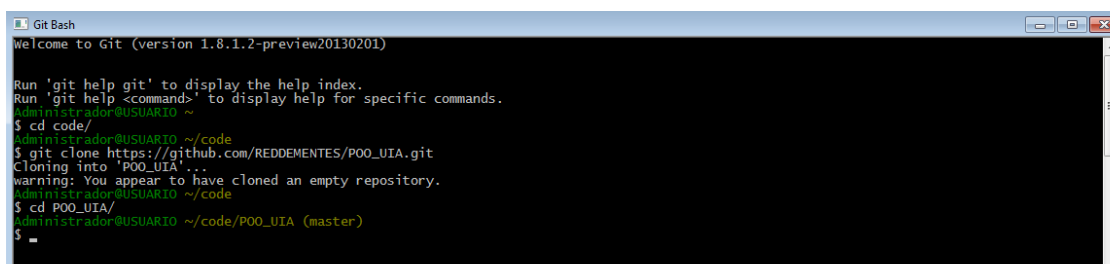
Abra la aplicación git bash y navegue hasta su carpeta que creo en la sección 2.2 y teclee el siguiente comando

```
git clone <url_de_su_repositorio>
```



Naveguen a su repositorio con el siguiente comando:

```
cd <nombre_repositorio>
```



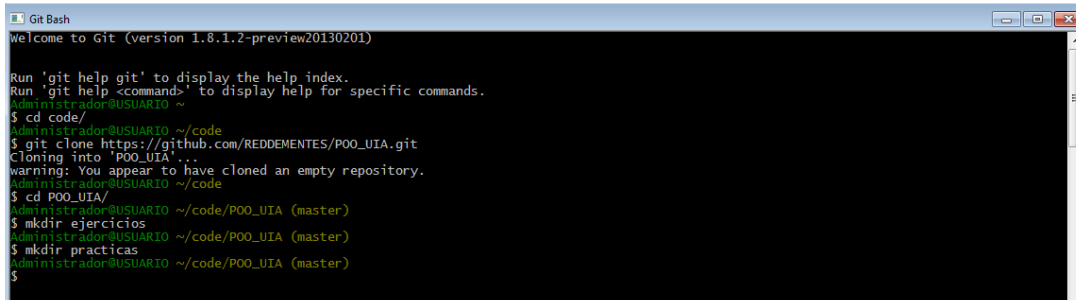
Ahora verán que la pantalla a cambiado y agrego el nombre de su repositorio algo como (master), esto significa que ya están dentro del repositorio.



Vamos a crear dos carpetas nuevas en el repositorio tecleando los siguientes comandos

```
mkdir ejercicios
```

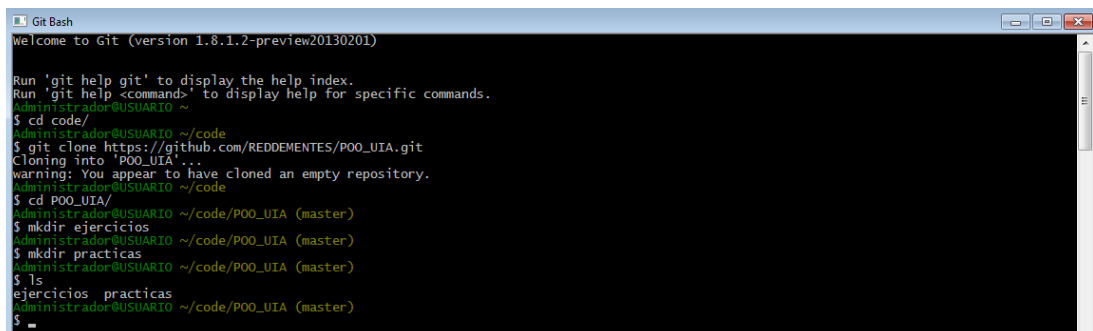
```
mkdir practicas
```



```
Git Bash
Welcome to Git (version 1.8.1.2-preview20130201)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.
Administrador@USUARIO ~
$ cd code/
Administrador@USUARIO ~/code
$ git clone https://github.com/REDDEMENTES/POO_UIA.git
Cloning into 'POO_UIA'...
warning: You appear to have cloned an empty repository.
Administrador@USUARIO ~/code
$ cd POO_UIA/
Administrador@USUARIO ~/code/POO_UIA (master)
$ mkdir ejercicios
Administrador@USUARIO ~/code/POO_UIA (master)
$ mkdir practicas
Administrador@USUARIO ~/code/POO_UIA (master)
$
```

Para visualizar que en verdad crearon las carpetas vamos a usar el comando `ls`

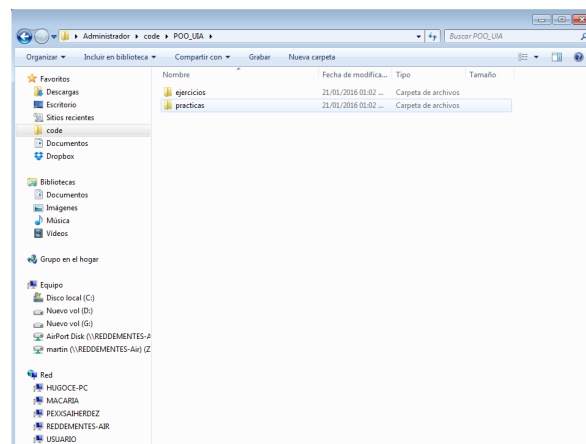


```
Git Bash
Welcome to Git (version 1.8.1.2-preview20130201)

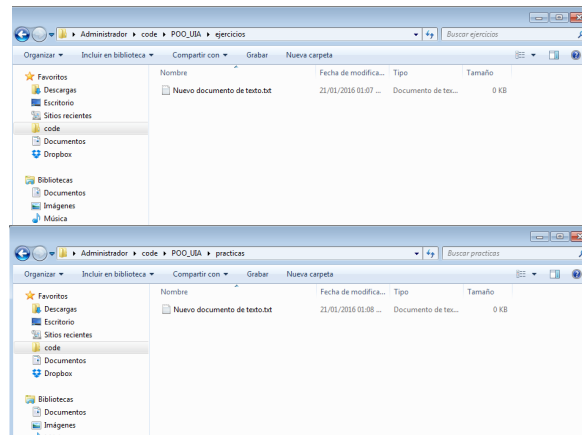
Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.
Administrador@USUARIO ~
$ cd code/
Administrador@USUARIO ~/code
$ git clone https://github.com/REDDEMENTES/POO_UIA.git
Cloning into 'POO_UIA'...
warning: You appear to have cloned an empty repository.
Administrador@USUARIO ~/code
$ cd POO_UIA/
Administrador@USUARIO ~/code/POO_UIA (master)
$ mkdir ejercicios
Administrador@USUARIO ~/code/POO_UIA (master)
$ mkdir practicas
Administrador@USUARIO ~/code/POO_UIA (master)
$ ls
ejercicios practicas
Administrador@USUARIO ~/code/POO_UIA (master)
$
```

Este comando visualiza el contenido de la carpeta en donde ejecutemos el comando.

Ahora naveguen esa carpeta desde su navegador de ventanas y tendrán algo como esto:



En cada una de las carpetas agreguen un archivo vacío:



## 4.2 Guardar

En este instante nuestro repositorio ya cambio pues agregamos dos nuevas carpetas, para ver que nuestro repositorio cambio podemos visualizarlo con el siguiente comando dentro de git bash:

```
git status
```

```
Git Bash
Administrador@USUARIO ~/code/POO_UIA (master)
$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#     ejercicios/
#     practicas/
nothing added to commit but untracked files present (use "git add" to track)
Administrador@USUARIO ~/code/POO_UIA (master)
$ _
```

Esto nos mostrara las nuevas carpetas que agregamos, ahora para guardar estos cambios usamos el comando:

```
git add .
```

```
Git Bash
Administrador@USUARIO ~/code/POO_UIA (master)
$ git add .
Administrador@USUARIO ~/code/POO_UIA (master)
$ _
```

Esto añadirá todos los archivos que cambiaron para ser guardados, ahora para terminar de guardarlo utilizamos el comando:

```
git commit -am 'Estructura del repositorio'
```

```
Git Bash
Administrador@USUARIO ~/code/POO_UIA (master)
$ git add.
Administrador@USUARIO ~/code/POO_UIA (master)
$ git commit -am 'Estructura del repositorio'
[master (root-commit) 5f2333c] Estructura del repositorio
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 ejercicios/Nuevo documento de texto.txt
create mode 100644 practicas/Nuevo documento de texto.txt
Administrador@USUARIO ~/code/POO_UIA (master)
$
```

Lo que logramos al ejecutar esta instrucción es salvar con esa descripción los cambios, es como si en un programa de edición diéramos clic en salvar como... y asignarle un nombre.

### 4.3 Subir cambios

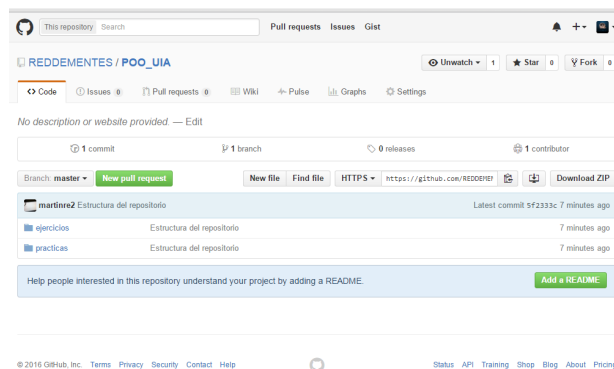
Ahora, estos cambios se encuentran en su maquina, pero no en nuestro repositorio remoto, para subir nuestros cambios a internet ejecutamos el siguiente comando:

```
git push origin master
```

Esto les pedira el username con el que se registraron en github y su contraseña.

```
Git Bash
Administrador@USUARIO ~/code/POO_UIA (master)
$ git push origin master
Username for 'https://github.com': reddementes
Password for 'https://reddementes@github.com':
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 302 bytes, done.
Total 4 (delta 0), reused 0 (delta 0)
To https://github.com:REDDEMENTES/POO_UIA.git
 * [new branch] master -> master
Administrador@USUARIO ~/code/POO_UIA (master)
$
```

Para corroborar pueden entrar al sitio de github para ver que sus archivos fueron subidos con éxito.



## **5. Ejercicios.**

La practica será muy sencilla, se pide que haga una búsqueda en internet de dos programas escritos en el lenguaje java, pueden ser sencillos o complejos, eso depende de usted. Una vez que los busquen y elijan, analícenlos y traten de indagar que es lo que hacen. Seguido de esto compílenlos y ejecútenlos (Sección 2).

Los archivos .java y .class los deben de subir dentro de la carpeta practicas/P01/ a su repositorio que crearon en la clase anterior (Sección 4).