

Programación Avanzada

II C2233

Nebil Kawas - Lucas Valenzuela - Ivania Donoso



Agenda

- Programa del curso
- Herramientas del curso
- Recomendaciones para pasar el curso
- Repaso de Intro

Programa



Este curso enseña técnicas para diseñar, implementar, ejecutar y evaluar software que resuelva problemas a partir de especificaciones detalladas.

Objetivos

1. Descomponer problemas complejos, para diseñar y estructurar sus soluciones.
2. Crear diseños orientados a objetos y comunicar estos diseños a través de documentación externa y comentarios en el código.
3. Aplicar conceptos de orientación a objetos y estructuras de datos fundamentales, para diseñar y escribir programas complejos en el lenguaje de programación Python, pudiendo extender este conocimiento a distintos lenguajes.
4. Usar herramientas de programación estándares; técnicas de programación; y un entorno de desarrollo de software para editar, ejecutar y depurar programas.
5. Generar software desde cero, con código de alto nivel y calidad, de fácil re-utilización, actualización y mantenimiento. Incluyendo también interfaces gráficas totalmente funcionales.

Contenidos del curso

- Programación Orientada a Objetos
- Estructuras de Datos
- Excepciones
- Testing
- Funciones en Python y programación funcional
- Meta Clases
- Simulación
- Threading
- Interfaces Gráficas
- I/O (strings, bytes, serialización)
- Networking
- Contenidos adicionales
 - Webservices
 - Regex

Programación Avanzada vs

Exploratorio de computación



Introducción a la programación



Comenzar su formación como desarrolladores de software <3



Metodología



Actividades

Antes de clases Ustedes **estudian** el material y resuelven dudas en las issues.

Antes de la actividad **Repaso** de la materia. Resuelven más dudas con los profesores.

Durante la actividad Resuelven **dudas** de materia y actividad con ayudantes y profesores. Las actividades serán en parejas e individuales.

Actividades

- Tendrán puesto asignados y se publicarán en el syllabus
- Los ayudantes pasarán lista, si están marcados como ausentes tendrán un 1.0 (uno) en esa actividad
- Es responsabilidad de ustedes verificar que los ayudantes los hayan puesto presentes
- No se recuperan actividades

Tareas

Tienen por objetivo que resuelvan un problema complejo.

1. Leer el enunciado
2. Hacer el entregable que les pidan (4 - 5 días)
3. Reciben *feedback* general sobre el entregable (2 -3 días después)
4. Tienen un semana para **seguir** trabajando

Controles y Examen

- Los controles duran de 10 a 20 minutos
- Son similares a las preguntas más fáciles del Examen
- Enfocados a la lectura de código
- El examen es el día 1 de diciembre y durará ~2.5 hrs

Evaluaciones

- 15 Actividades (A)
- 6 o 7 Tareas (T)
- 8 Controles (C)
- 1 Examen Final (E)

$$NP = 0.25xA + 0.4xT + 0.2xC + 0.15xE$$

Evaluaciones

- Adicionalmente para aprobar el curso el alumno debe cumplir con:
 $E \geq 3.50$
 $\text{promedio}(A) \geq 4.00$
 $\text{promedio}(T) \geq 3.950$
- Si el alumno cumple con las condiciones anteriores **$NF = NP$** . En caso contrario **$NF = \min(3.9; NP)$**

Evaluaciones

- La inasistencia a alguna de las evaluaciones (actividad, control, examen) se evalúa con nota 1.0
- Se pueden eliminar
 - las dos peores notas de actividades.
 - la nota de un control
- **NO se borrará ninguna otra evaluación**
- La tarea 7 es opcional y deberán inscribirse.
- Solo será aproximada la nota final NF. El resto de las notas serán usadas con dos decimales.

Correcciones y Recorrecciones

- Las notas de actividades y tareas se publican **a más tardar 15 días** hábiles después de haber realizado la evaluación.
- Tendrán **una semana** para corregir después de que se publiquen las notas.
- Los controles se van a buscar al DCC, escriben su solicitud en un papel y se lo entregan a Yessenia (secretaria del DCC).
- Las actividades y tareas se recorren a través del [formulario de corrección](#).

Solicitud de corrección

- La nota puede bajar
- No se aceptarán correcciones del tipo
 - Me merezco más puntaje
 - El ayudante dice que mi programa no corre pero a mí sí me funciona
- La solicitud de corrección debe indicar cuáles fueron los puntos mal corregidos y por qué están mal corregidos.
- Si no están de acuerdo con la respuesta de los ayudantes después de la corrección deberán esperar hasta la corrección final para resolver el problema

**La corrección final es
el 4 de diciembre. NO
HAY OTRA FECHA.**

**En la recorreción final
no se corregirán
evaluaciones que no
fueron mandadas a
recorregir en el período
que correspondía.**

Normas en evaluaciones

No respetar las indicaciones de cada evaluación tiene como sanción inmediata un 1.0 en dicha evaluación.

Por ejemplo: cambios arbitrarios en las parejas de trabajo, no respetar los medios de entrega de evaluaciones, formatos, etc.

Integridad Académica

Cualquier situación de copia en alguna evaluación tendrá como **sanción un 1.1 final en el curso**. Esto sin perjuicio de sanciones posteriores que estén de acuerdo a la Política de Integridad Académica de la Escuela de Ingeniería y de la Universidad, que sean aplicables para el caso.

Integridad Académica

- Deben indicar la fuente de cualquier código que encuentren en internet y que usen en sus tareas y/o actividades
- Deben indicar si están usando código del material del curso o de las ayudantías
- Si no lo hacen, se considerará plagio.

Fechas

<https://iic2233.github.io/calendario/>

Cuerpo Docente



Nebil



Lucas



Ivania

Ayudantes Jefes



Florence



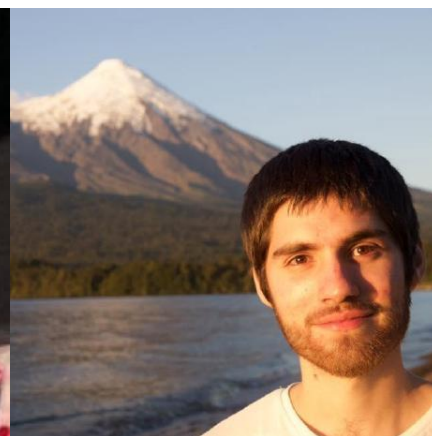
Fernando



Stephanie



Joaquín



Andrés

Ayudantes

Tareas

- **Hernan Valdivieso**
- **Hugo Navarrete**
- **Ignacio Acevedo**
- **Sebastián Cruz**
- Isaac Carrera
- Vicente Juárez
- Camilo López
- Felipe Quinteros
- Erick Romero
- Nicolás Balbontín
- Felipe Dominguez
- Jessica Hormazabal
- Tanya Garrido
- Tomás Salvadores
- Gustavo Prudencio

Docencia

- **Enzo Tamburini**
- **Sebastian Guerra**
- **Benjamín Cárcamo**
- Octavio Vera
- Raúl Álvarez
- Alejandro Kaminetzky
- Santiago Torres
- José Manuel Larraín
- Paula Salvo
- Jorge Riesco
- Gabriel Lyon

Consultas

- Administrativas o casos particulares:

<https://iic2233.github.io/contacto/>

- Contenidos del curso, enunciados y pautas

<https://github.com/IIC2233/Syllabus/issues>

NO MANDEN MAILS :)

Herramientas del curso



Python

<https://www.python.org/>

<https://zen-of-python.info/>



Guido van Rossum, creador de Python, en la convención OSCON 2006. Fuente: [Wikipedia](#).

PEP 8

Guía de estilo

PEP8

- Python Enhancement Proposal 8 es la guía de estilo de Python
- Se usa para hacer más legible y consistente el código
- <https://www.python.org/dev/peps/pep-0008/>

PEP8

- Imports al comienzo del módulo
- Nombres de variables descriptivos
- Espacios entre líneas
 - 2 líneas después de los imports
 - 2 líneas alrededor de las clases y funciones
 - 1 línea entre métodos de clase
 - 1 espacio después de “,” y a cada lado de los operadores
- Líneas de máximo 80 caracteres (incluyendo espacios)
- NO usar tabs. Solo usar espacios.

CamelCase y snake_case

```
CONST_PI = 3.1415
```

```
class ClaseDeEjemplo:
```

```
    def __init__(self, hola):  
        self.variable_de_ejemplo = hola
```

```
    def metodo_de_ejemplo(self):  
        return 1 + 1 == 2
```

**Siempre recuerda
que el código se
lee más veces de
lo que se escribe
y que otro lo va a
leer.**

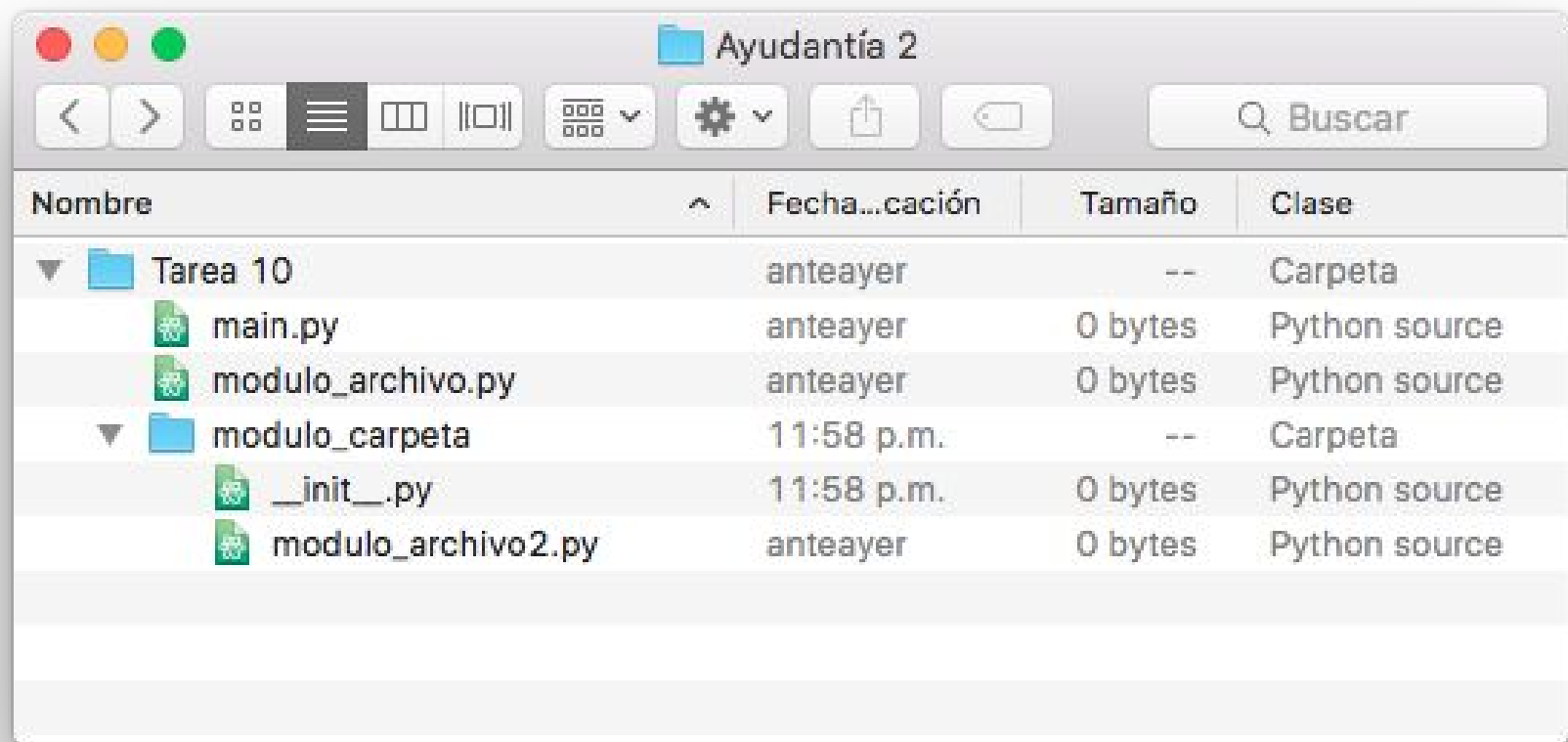
Modularización

—

Modularización: ¿Por qué?

- Cuando un programa crece, se hace inviable mantenerlo en un solo archivo:
 - El mantenimiento es difícil
 - El trabajo en equipo es difícil
 - Es desordenado
- Como es un archivo de python normal puede tener:
 - Variables
 - Métodos
 - Clases

Modularización



Cómo usar módulos

Importándolo entero

```
import modulo_archivo
```

```
if __name__ == "__main__":
```

```
    variable_tipica = modulo_archivo.VALOR_FIJO
```

```
    objeto_tipico = modulo_archivo.Clase()
```

```
    modulo_archivo.funcion()
```


Cómo usar módulos

Importándolo entero con pseudónimo

```
import modulo_archivo as ma
```

```
if __name__ == "__main__":  
    variable_tipica = ma.VALOR_FIJO  
  
    objeto_tipico = ma.Clase()  
  
    ma.funcion()
```

Cómo usar módulos

Importando lo necesario

```
from modulo_archivo import VALOR_FIJO, Clase, funcion
```

```
if __name__ == "__main__":  
    variable_tipica = VALOR_FIJO  
  
    objeto_tipico = Clase()  
  
    funcion()
```

Cómo usar módulos

Importando un package

Un package es una carpeta que tiene tener el archivo vacío “__init__.py”

```
import modulo_carpeta as mc
```

```
if __name__ == "__main__":
```

```
    variable_tipica = mc.modulo_archivo2.VALOR_FIJO
```

```
    objeto_tipico = mc.modulo_archivo2.Clase()
```

```
    mc.modulo_archivo2.funcion()
```

Cómo usar módulos

- Cuando se importa un módulo se ejecuta todo el código en él
- Para evitar que se ejecute código de un módulo al ser importado se utiliza el siguiente if:

Código del módulo

```
if __name__ == "__main__":
```

```
    # Mucho código escrito
```

`__name__?`

bar.py

```
import foo

# Código

if __name__ == "__main__":
    # Código
```

foo.py

```
# Código
def method():
    pass
```

Cómo **NO** usar módulos

Importando todo sin referencia al módulo

```
from modulo_archivo import *  
  
if __name__ == "__main__":  
    variable_tipica = VALOR_FIJO  
    objeto_tipico = Clase()  
    funcion()
```



Cómo **NO** usar módulos

- Evita crear módulos que se llamen igual a los que vienen incluidos en python
- ¿Cómo busca los módulos python?:
 - Módulo de la librería estándar
 - Módulo en la misma carpeta
 - Módulo en el directorio de instalación

Git

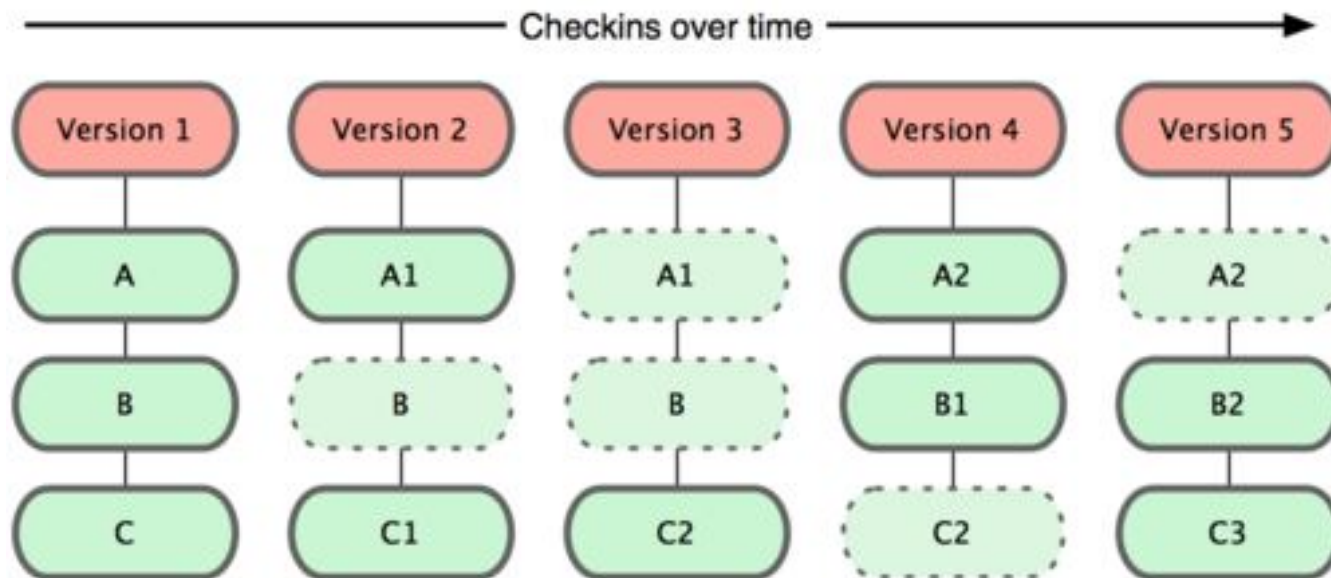


¿Qué es git?

Git es un sistema distribuido de control de versión, gratuito y open source, diseñado para manejar de pequeños a enormes proyectos de forma rápida y eficiente ¹



¹ <https://git-scm.com/>



Fuente: git-scm.com

Ventajas

- Trabajo en equipo fluido (No hay problemas como en Dropbox)
- Versiones disponibles en cualquier momento
- Control de cambios
- Programar versiones en paralelo y luego juntarlas
- Múltiples backup de sus programas

¿Qué es GitHub?

Es una plataforma para alojar proyectos usando el sistema de control de versiones git

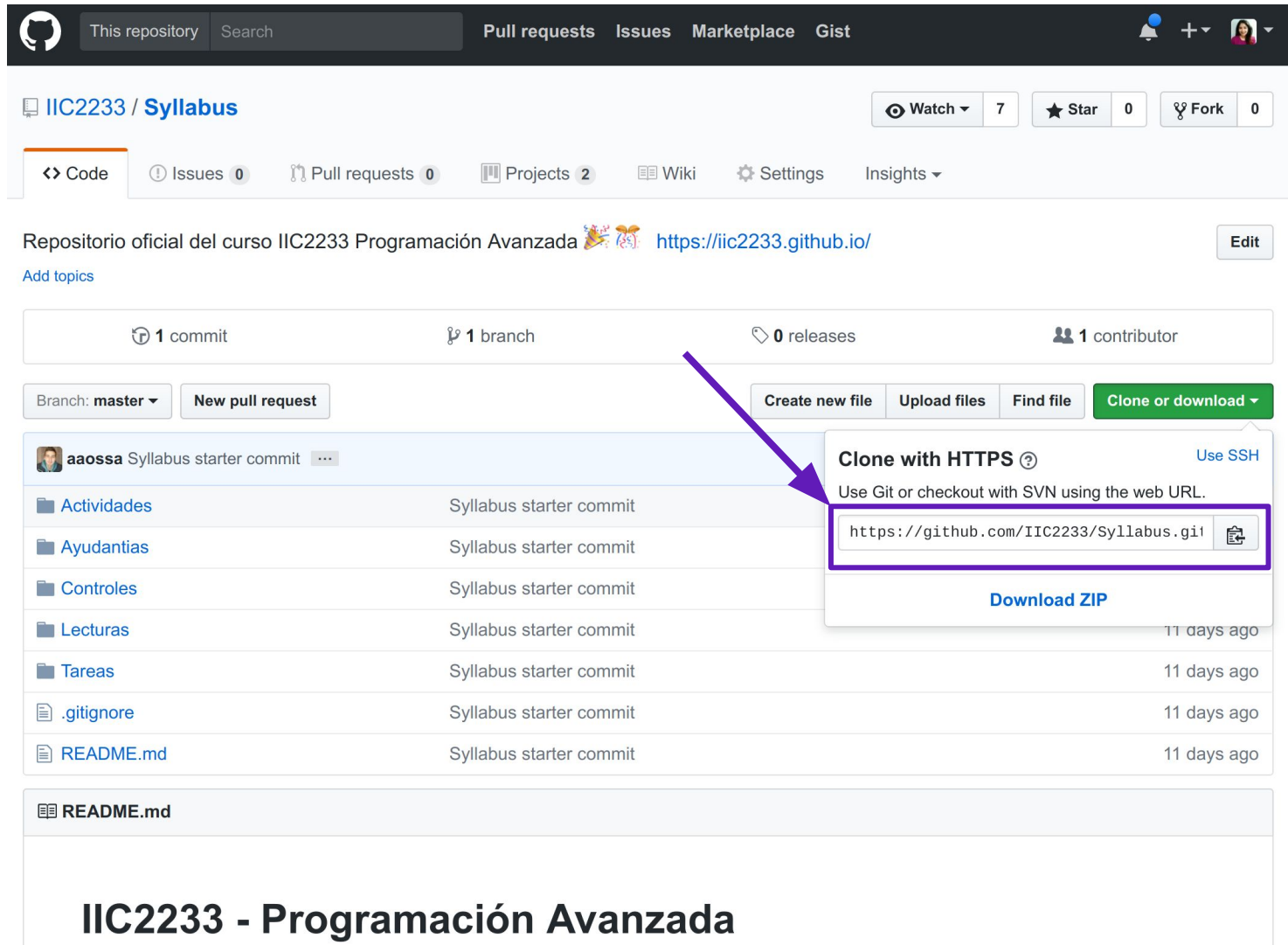


**Se usa en la vida
real. Es
obligatorio
conocerlo :)**

Setup

—

Obtener dirección



The screenshot shows the GitHub interface for the repository 'IIC2233 / Syllabus'. The repository has 7 watches, 0 stars, and 0 forks. It contains 1 commit, 1 branch, 0 releases, and 1 contributor. The 'Clone or download' button is highlighted with a green box, and a purple arrow points to the 'Clone with HTTPS' modal. The modal displays the URL 'https://github.com/IIC2233/Syllabus.git' and a 'Download ZIP' button. Below the repository information, a table lists the files and folders in the repository, all of which were committed by 'aaossa' on 'Syllabus starter commit' 11 days ago.

Repository: IIC2233 / Syllabus

Watch 7 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 2 Wiki Settings Insights

Repositorio oficial del curso IIC2233 Programación Avanzada <https://iic2233.github.io/> Edit

Add topics

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request

Create new file Upload files Find file Clone or download

Clone with HTTPS Use SSH

Use Git or checkout with SVN using the web URL.

<https://github.com/IIC2233/Syllabus.git>

Download ZIP

File/Folder	Commit	Time
Actividades	Syllabus starter commit	11 days ago
Ayudantias	Syllabus starter commit	11 days ago
Controles	Syllabus starter commit	11 days ago
Lecturas	Syllabus starter commit	11 days ago
Tareas	Syllabus starter commit	11 days ago
.gitignore	Syllabus starter commit	11 days ago
README.md	Syllabus starter commit	11 days ago

README.md

IIC2233 - Programación Avanzada

Clonar el repositorio

En la consola escribir

```
git clone https://github.com/IIC2233/syllabus.git
```

Recuerda estar en la carpeta en la que quieren mantener el repo

Clonar el repositorio

PrograAvanzada



syllabus

ivanía@Ivania-Ubuntu: ~/PrograAvanzada

```
ivanía@Ivania-Ubuntu:~/PrograAvanzada$ git clone https://github.com/IIC2233-2016-02/syllabus.git
Clonar en «syllabus»...
remote: Counting objects: 14, done.
remote: Total 14 (delta 0), reused 0 (delta 0), pack-reused 14
Unpacking objects: 100% (14/14), done.
Checking connectivity... hecho.
ivanía@Ivania-Ubuntu:~/PrograAvanzada$
```

Clonen sus repositorios

Ejercicio

Vayan al sitio de github, obtengan la dirección y clonen sus repos en sus computadores.

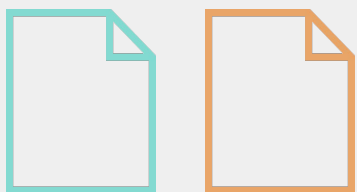
Si no tienen repositorio, creen uno en el sitio de git

**¿Cómo funciona
git y github?**

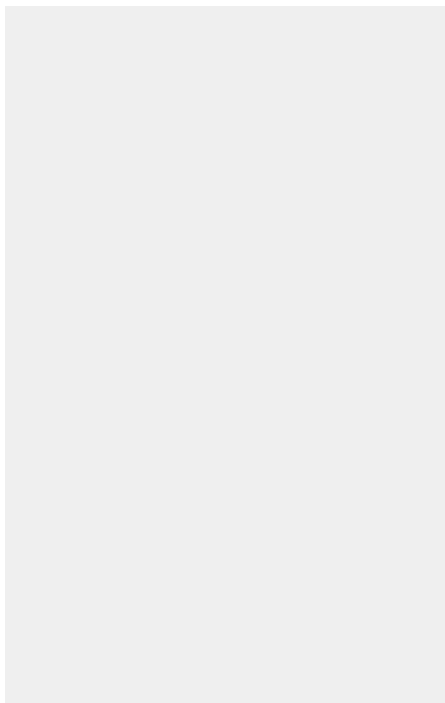
Conceptos

1. **Working directory:** lugar local en donde están los archivos que luego serán parte del repositorio
2. **Staging Area:** “lugar” en dónde están los archivos que se van a commitear.
3. **Repositorio local (o repo):** lugar local que contiene todos los archivos que han sido “commiteados”
4. **Repositorio remoto:** lugar remoto (en un servidor) que contiene todos los archivos que han sido “commiteados”. Para esto utilizamos el servicio de GitHub.

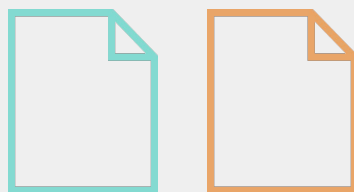
Working directory



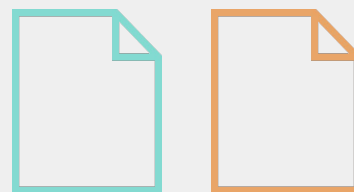
Staging Area



Repositorio local



Repositorio remoto

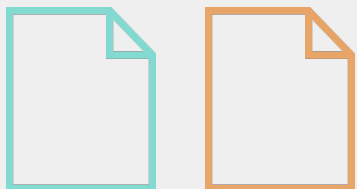


Working directory

Staging Area

Repositorio local

Repositorio remoto

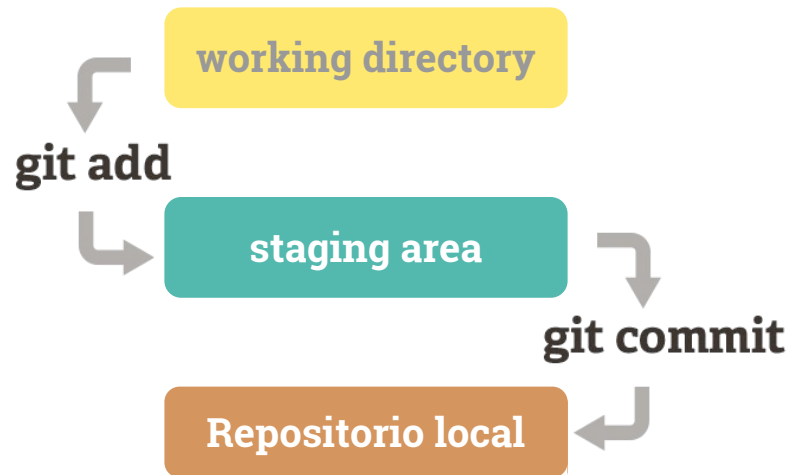


?

Crear una versión

`git add file_name`

`git add *.txt`



`git commit -m`
"Mensaje descriptivo"

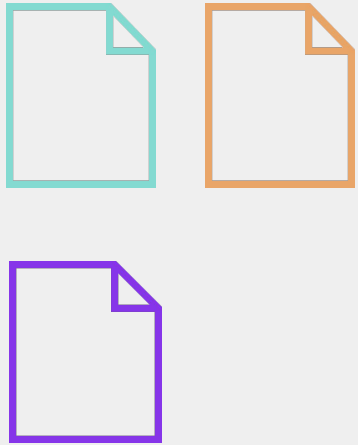
Los mensajes son MUY importantes. Son una ayuda al ustedes del futuro.

Revisen esta guía de estilo

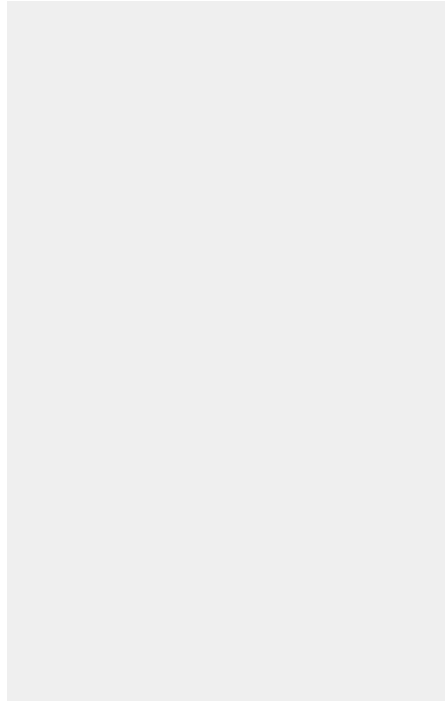
<https://gist.github.com/nebil/f96a2f0bfe1e059d589d6a2190a2ac81>

Subir un archivo

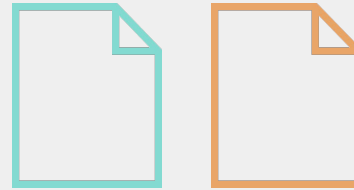
Working directory



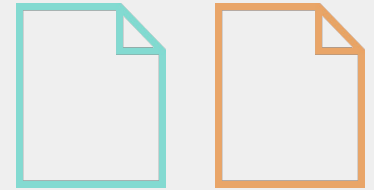
Staging Area



Repositorio local

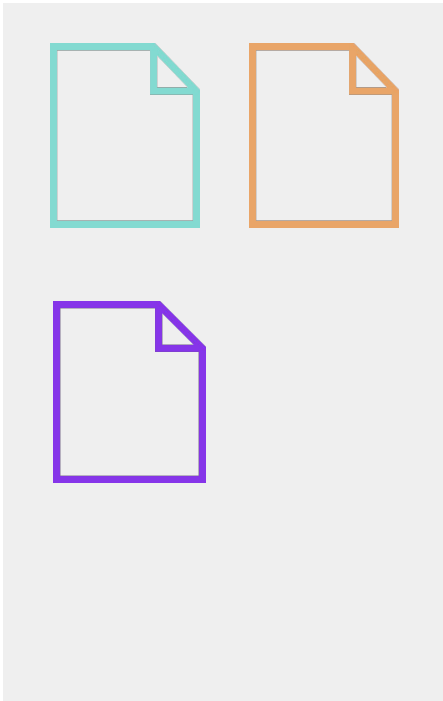


Repositorio remoto

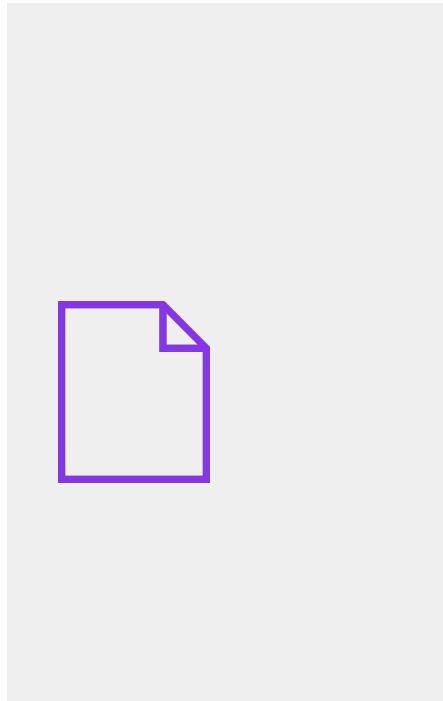


Subir un archivo

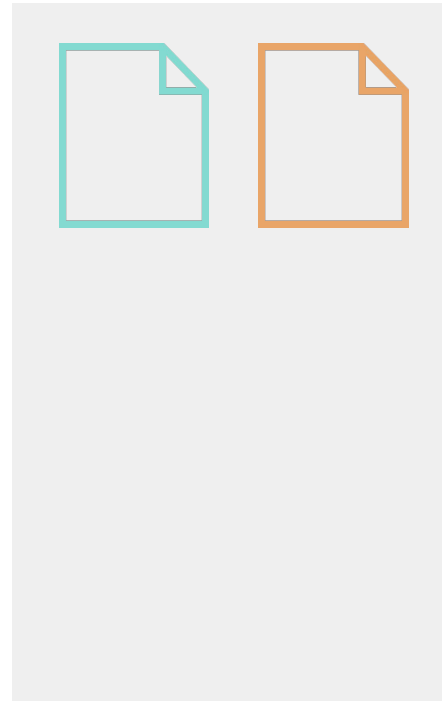
Working directory



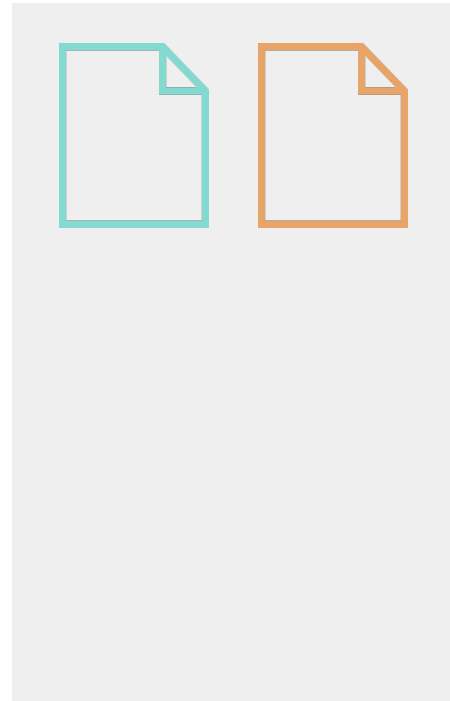
Staging Area



Repositorio local



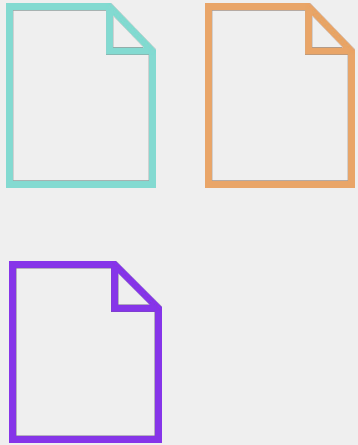
Repositorio remoto



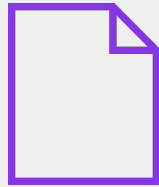
git add

Subir un archivo

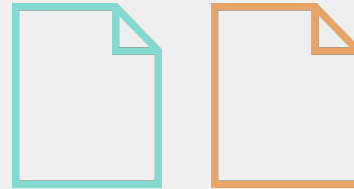
Working directory



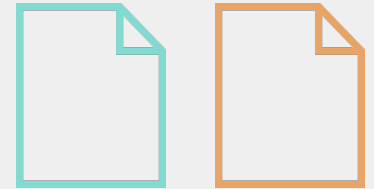
Staging Area



Repositorio local

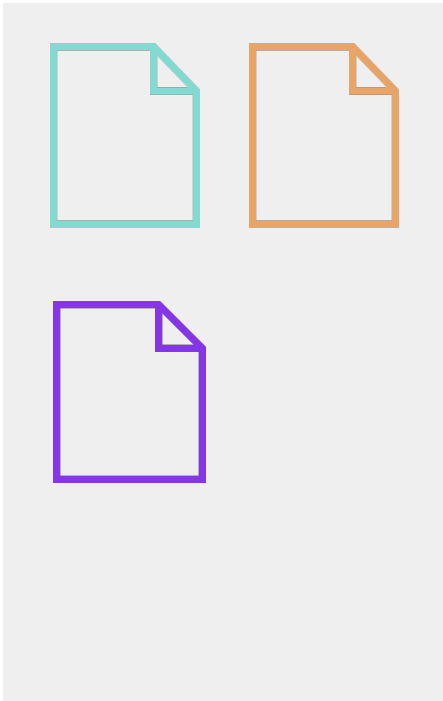


Repositorio remoto

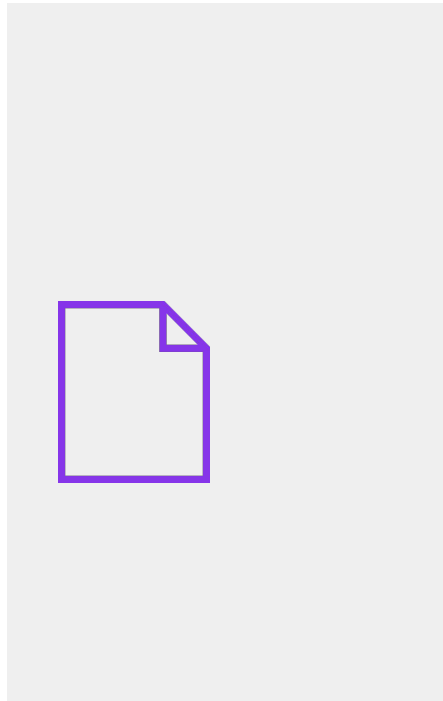


Subir un archivo

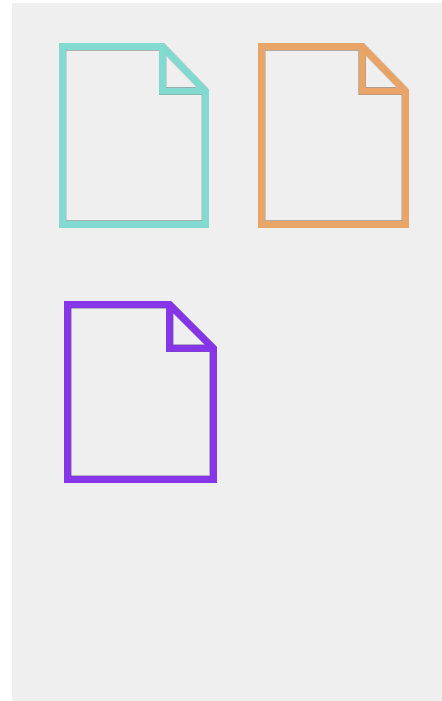
Working directory



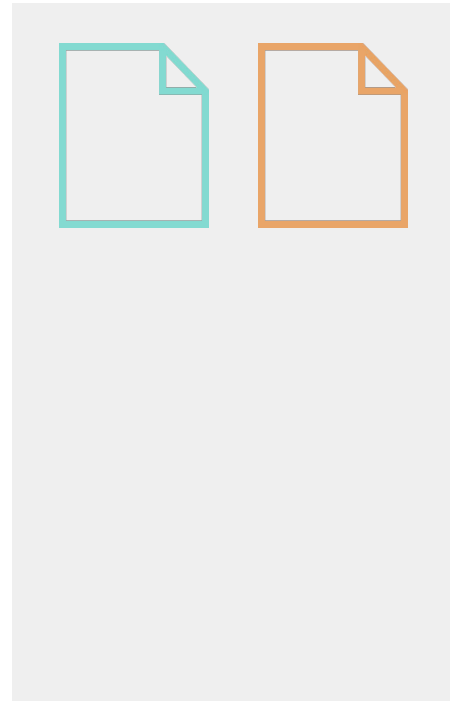
Staging Area



Repositorio local



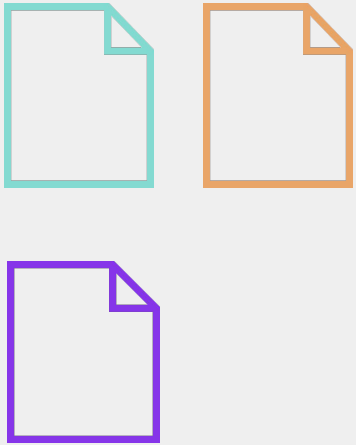
Repositorio remoto



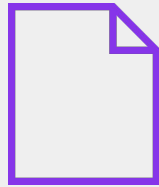
git commit

Subir un archivo

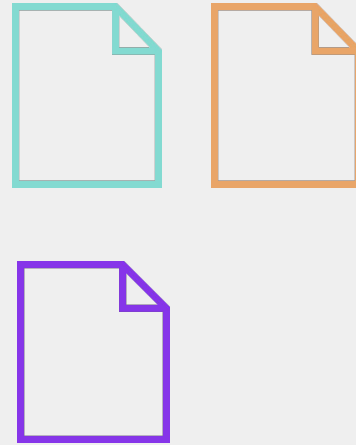
Working directory



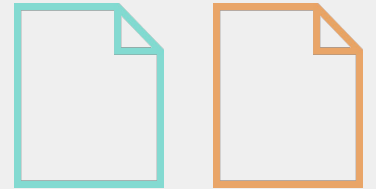
Staging Area



Repositorio local

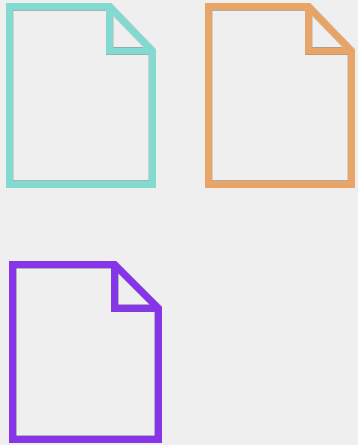


Repositorio remoto

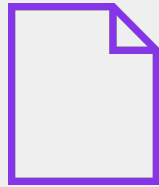


Subir un archivo

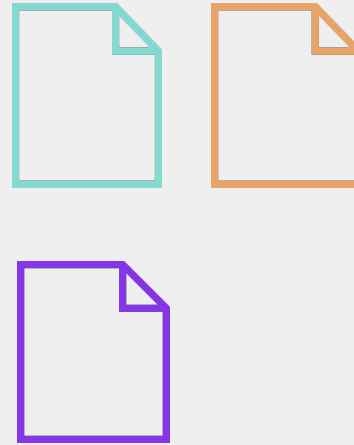
Working directory



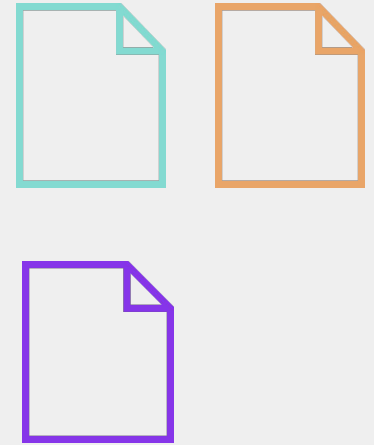
Staging Area



Repositorio local



Repositorio remoto



git push

Modificar un archivo

Ejercicio

1. Abre el archivo README.md de tu repositorio y agrega los datos que faltan
2. Agrégalo al staging area
3. Ejecuta el comando git status. Debería aparecer el mensaje “Cambios para hacer commit”
4. Agrégalo al repositorio local
5. Agrégalo al repositorio remoto

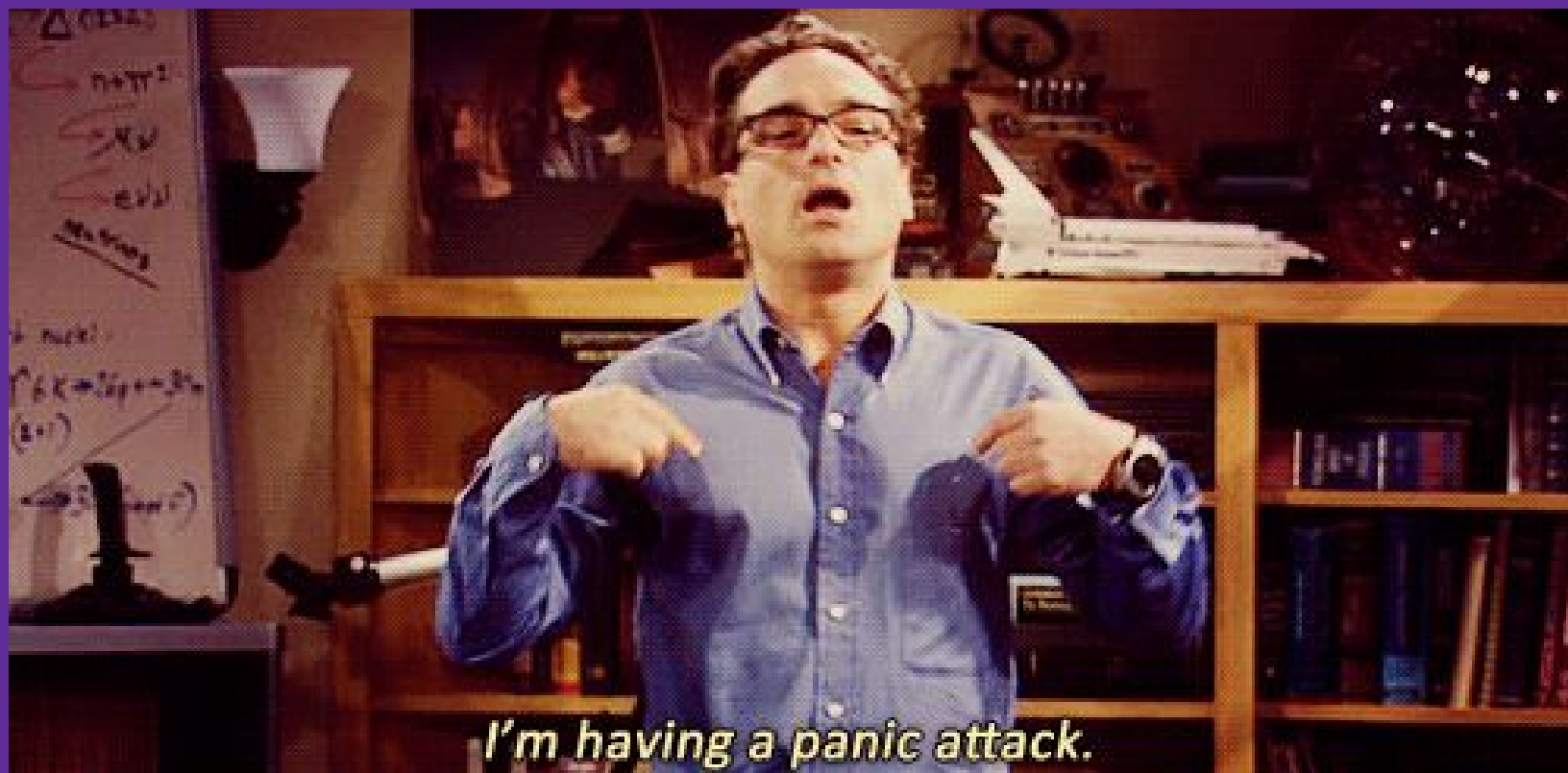
Subir un archivo

Ejercicio

1. Crea un archivo llamado hola.txt en la carpeta AC00
2. Agrégalo al staging area
3. Ejecuta el comando git status. Debería aparecer el mensaje “Cambios para hacer commit”
4. Agrégalo al repositorio local
5. Agrégalo al repositorio remoto

Cambiar el stage

Son las 16:48. Las instrucciones dicen que no debo subir el archivo “VeryHeavyFile.txt” que pesa 100 MB. Hice `git add *.txt` y solo me queda un minuto para poder subir la actividad.



I'm having a panic attack.

```
git reset HEAD file_name
```

Ya hice commit



```
git reset HEAD~1
```

Cambiar el stage

Ejercicio

1. Crea un archivo llamado hola1.txt en la carpeta AC00 y agrégalo al staging area.
2. Crea un archivo llamado hola2.txt y agrégalo al staging area.
3. Elimina el archivo hola1.txt del staging area.
4. Agrega el archivo hola2.txt al repositorio remoto.

Sitios útiles

www.git-scm.com

<http://ndpsoftware.com/git-cheatsheet.html>

<http://rogerdudler.github.io/git-guide/>

** <http://rypress.com/tutorials/git/index>

<https://www.udacity.com/courses/ud775>

<https://gist.github.com/nebil/f96a2f0bfe1e059d589d6a2190a2ac81>

<https://gist.github.com/nebil/10ac0626bd2ef06e5fc158f6504c3ecd>

Jupyter Notebook



“The Jupyter Notebook is a web application that allows you to create and share documents that contain **live code**, equations, visualizations and **explanatory text**.” *(<http://jupyter.org/>)*

Instalación: <http://jupyter.org/install.html>

Google

+



stackoverflow

¿Cómo buscar soluciones?

python [versión] [librería] [duda]



¡EN INGLÉS!

¿Cómo imprimir una cola con Python?



Python 3.5 collections print queue



¿Cómo buscar soluciones?

python [versión] [error]



¡EN INGLÉS!

NameError: name "MiVariable" is not defined



NameError: name * is not defined





python3.5 NameError: name * is not defined



Todos

Videos

Maps

Imágenes

Noticias

Más

Preferencias

Herramientas

Cerca de 95,800 resultados (0.50 segundos)

[In Python3.5:NameError: name 'image_to_string' is not defined](#)

<https://stackoverflow.com/.../in-python3-5nameerror-name-image-...> ▼ Traducir esta página

11 jun. 2017 - Please post your source code so we can look over the code and get more details. Also your error is caused by a variable declaration without a ...

[oop - Python3 NameError: name 'method' is not defined - Stack Overflow](#)

<https://stackoverflow.com/.../python3-nameerror-name-method-is-...> ▼ Traducir esta página

18 mar. 2016 - consider you have the function **defined** in the global scope: def recursive(x): if (x>5): print (x) recursive(x - 1). you would simply call this with ...

[input\(\) error - NameError: name '...' is not defined - Stack Overflow](#)

<https://stackoverflow.com/.../input-error-nameerror-name-is-not-...> ▼ Traducir esta página

14 ene. 2014 - input_variable = input("Enter your name: ") print("your name is" + input_variable) ... input("Enter your name: ") File "<string>", line 1, in <module> **NameError: name 'dude' is not defined** ... I did what Kevin said and it is version 2.7.5! ... If you are using **Python 3.x**, raw_input has been renamed to input .

[python NameError: name 'file' is not defined in python 3.5 - Stack ...](#)

<https://stackoverflow.com/.../python-nameerror-name-file-is-not-...> ▼ Traducir esta página

26 nov. 2015 - Traceback (most recent call last): File "c:\python3.5\lib\runpy.py", line python 3.x from this Q: python **NameError: name 'file' is not defined** But ...

[python 3.x - NameError: name 'value' is not defined - Stack Overflow](#)

<https://stackoverflow.com/.../nameerror-name-value-is-not-define-...> ▼ Traducir esta página

5 abr. 2014 - **NameError: name 'value' is not defined** ... variable defined in a function is not visible outside the function. ... answered Apr 5 '14 at 2:36

[NameError: global name 'unicode' is not defined in Python 3 - Stack ...](#)

<https://stackoverflow.com/.../nameerror-global-name-unicode-is-...> ▼ Traducir esta página

9 nov. 2013 - **Python 3** renamed the unicode type to str, the old str type has been replaced by bytes . if isinstance(unicode or str, str): text = unicode or str ...

Otras recomendaciones



- Empezar las tareas cuando entreguen el enunciado
- Buscar más en Google
- Estudiar el material de clases
- Ir a las ayudantías
- Estudiar el ramo en serio desde el principio
- Ser estratégico con las tareas
- Dedicarle tiempo a otros ramos
- Dormir

Referencias

- www.git-scm.com
- Imagen de archivo creada por Richard Schumann desde Noun Project