

Programación Avanzada

IIC2233 2019-1

Fernando Florenzano – Antonio Ossa – Cristian Ruz – Vicente Domínguez



Agenda

- Motivación
- Equipo docente
- Programa y metodología del curso
- Recomendaciones
- Herramientas del curso
- Actividad de Git y GitHub

Motivación



¿Qué implicaría **programar** la aplicación **Uber** usando lo que sabemos de **Introducción a la Programación**?

Vamos por partes:

- Pedir y mostrar datos de perfil de usuario
- Revisar historial de viajes.
- Definir camino entre dos direcciones.
- Interfaz gráfica.
- Comunicación con conductor.



Edit Account



First Name

Fernando



Last Name

Florenzano



Phone Number

Verified >

Email



Your Trips

Past

Upcoming

Family

9/16/18, 22:43

\$2,287

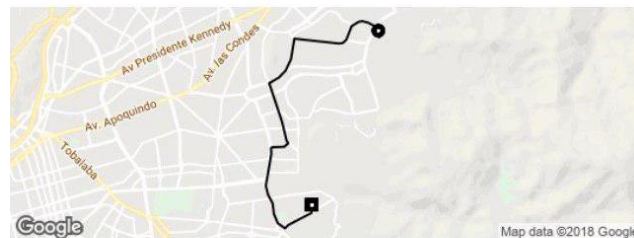
Peugeot 301



9/1/18, 19:30


\$6,083

Toyota Corolla



×

Edit Account



First Name

Fernando

>

Last Name

Florenzano

>

Phone Number

Verified >

Email

>

```
class Usuario:
```

```
    def __init__(self, f, l, p, e):
        self.first_name = f
        self.last_name = l
        self.phone_number = p
        self.email = e
        self.trips = []
        self.picture = None
```

```
    def add_trip(self, t):
        self.trips.append(t)
```

```
    def edit_picture(self, p):
        self.picture = p
```

```
u = Usuario("Fernando", "Florenzano", "123", "f@uc.cl")
v1 = Viaje("9/16/18, 22:43", "2287", A, B, driver1)
v2 = Viaje("9/1/18, 19:30", "6083", C, D, driver2)
u.add_trip(v1)
u.add_trip(v2)
```

×

Your Trips


Past

Upcoming

Family

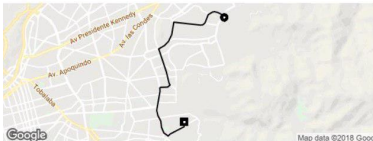
9/16/18, 22:43

\$2,287



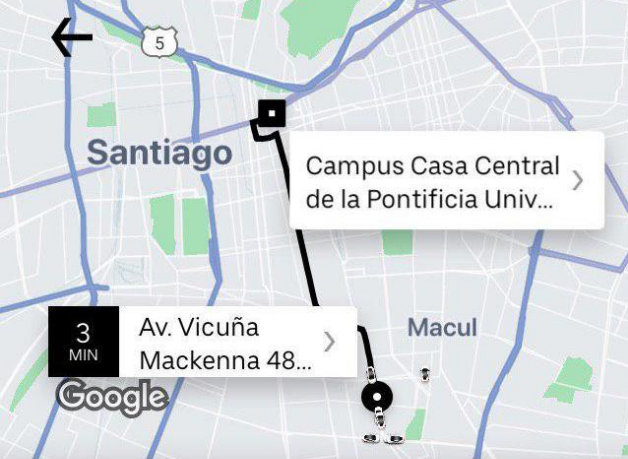
9/1/18, 19:30

\$6,083



```
class Viaje:
```

```
    def __init__(self, d, p, s, e, dr):
        self.date = d
        self.price = p
        self.start = s
        self.end = e
        self.driver = dr
```



Popular

Affordable, everyday rides



UberX

CLP4,664 ⓘ



Black

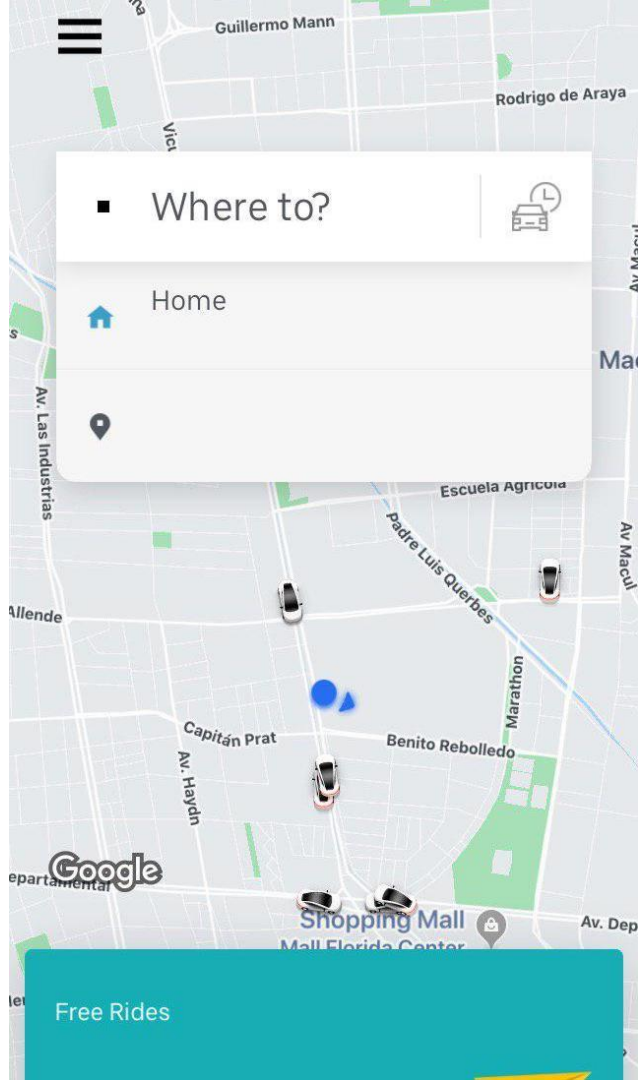
CLP9,829





1-4

CONFIRM UBERX



Where to?



Home



Gerardo

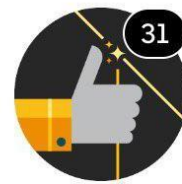
4.75 ★

Rating

2

Months

Compliments



¿Qué podemos hacer?

- Pedir y mostrar datos de perfil de usuario
- Revisar historial de viajes.
- Definir camino entre dos direcciones.
- Interfaz gráfica.
- Comunicación con conductor.



Equipo docente

Profesores



Sección 1
Fernando



Sección 2
Antonio



Sección 3
Cristian



Sección 4
Vicente

Ayudantes jefes

Área Tareas
Ignacio



Área Docencia
Enzo

Coordinador
Hernán

Cuerpo de ayudantes

Full Tareas

- Diego Collao
- Daniela Concha
- Ignacio Contreras
- Francisca Ibarra
- Hue Bin Kim
- Juan Schuwirth
- José Manuel Wielandt

Híbridos Tareas

- Raúl Álvarez
- Juan Echavarri
- Benjamín Farías
- Jessica Hormazabal
- Ariel Martínez
- Roberto Negrin
- Jorge Pérez
- Mario Reinike
- Juan Romero
- Ricardo Schilling

Híbridos Docencia

- Fernando Alfaro
- Ian Basly
- Benjamín Martínez
- Nicolás Quiroz
- Valentina Rojas
- Irina Salazar
- Maria Jose Varela

Full Docencia

- Juan José Aguillón
- Christian Eilers
- Michael Hund
- Pablo Olea
- Dante Pinto

Programa y metodología

La “fama” del curso

- Consume mucho tiempo
- Muchos contenidos
- Consume mucho tiempo
- Difícil
- Consume mucho tiempo

La “fama” del curso

- Consume mucho tiempo
- Muchos contenidos
- Consume mucho tiempo
- Difícil
- Consume mucho tiempo
- Se aprende mucho
- Prepara para el futuro
- Se aprende mucho
- Otorga autonomía
- Se aprende mucho

La nueva versión 2019-1

Durante diciembre 2018 y enero 2019 se reformularon los contenidos y competencias que el curso quiere que sus alumnos aprendan.

Y este es el resultado...

<https://iic2233.github.io>

Página del curso

*“A lo largo del este curso, el **alumno** desarrollará técnicas para diseñar, implementar, ejecutar y evaluar herramientas de software que resuelven problemas algorítmicos a partir de especificaciones detalladas.”*

Contenidos del curso

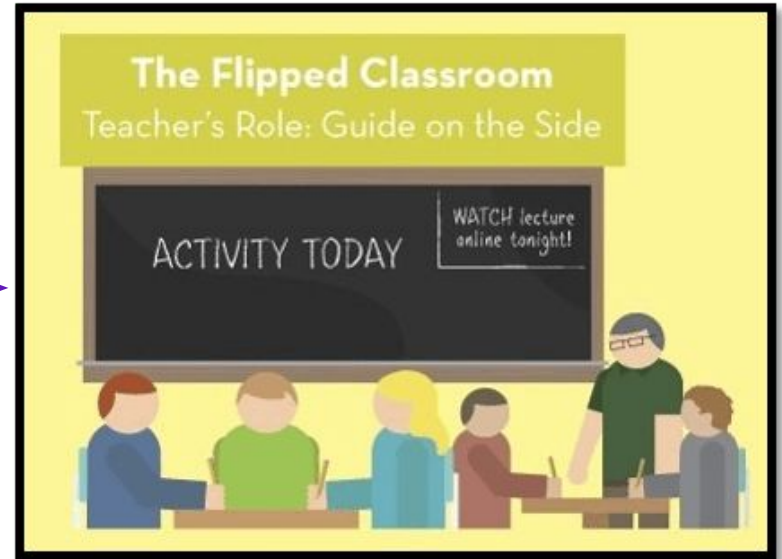
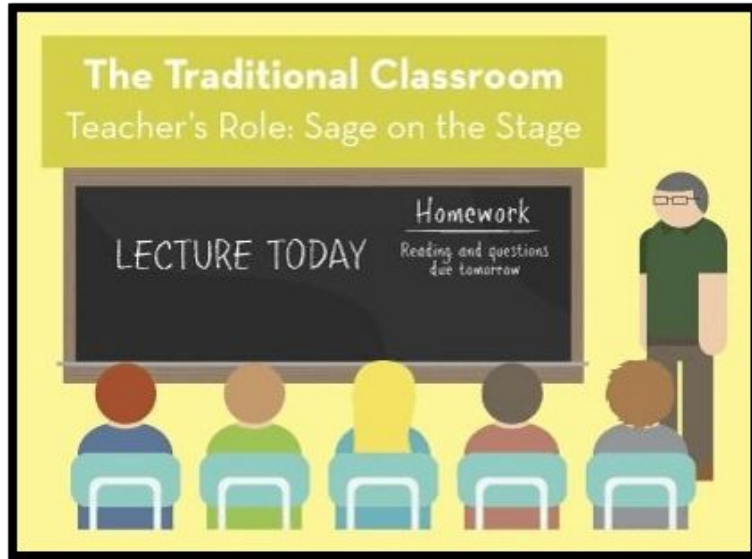
Fundamentos de programación

- Estructuras de datos *built-ins*
- Programación orientada a objetos
- Estructuras de datos con nodos
- Excepciones

Herramientas y patrones de programación

- *Threading*
- Interfaces gráficas
- I/O
- *Networking*

Metodología: *Flipped Classroom*



¿Cómo se refleja en el curso?

- Los contenidos y evaluaciones son preparados (y corregidos) por el equipo docente
- Los alumnos aprenden **haciendo**:

¿Cómo se refleja en el curso?

- Los contenidos y evaluaciones son preparados (y corregidos) por el equipo docente
- Los alumnos aprenden **haciendo**:
 - mediante **3 les y un examen**

¿Cómo se refleja en el curso?

- Los contenidos y evaluaciones son preparados (y corregidos) por el equipo docente
- Los alumnos aprenden **haciendo**:
 - mediante **actividades** en clase
 - mediante **tareas** en casa
 - y nada más.

¿Cómo se refleja en el curso?

- Los contenidos y evaluaciones son preparados (y corregidos) por el equipo docente
- Los alumnos aprenden **haciendo**:
 - mediante **actividades** en clase
 - mediante **tareas** en casa
 - y nada más.
- La planificación se asegura que cada semana solo hay **una** evaluación de la cual preocuparse.

Actividades

Objetivo:

poner en práctica un contenido estudiado mediante una actividad de programación breve

Actividades

Lu	Ma	Mi	Ju	Vi	Sa	Do

Actividades

Lu	Ma	Mi	Ju	Vi	Sa	Do

Viernes: Los profesores **suben el material** de la semana para que los alumnos **estudien** y resuelvan dudas en el foro del curso.

Actividades

Lu	Ma	Mi	Ju	Vi	Sa	Do

Martes: Habrá una ayudantía para **reforzar** los contenidos y para resolver dudas de forma presencial con los ayudantes.

Actividades

Lu	Ma	Mi	Ju	Vi	Sa	Do

Miércoles: Se subirán **puestos asignados** de la sala al [syllabus](#), para la clase del siguiente día.

Actividades

Lu	Ma	Mi	Ju	Vi	Sa	Do

Jueves: ¡Día de actividad!

Actividades

Jueves: ¡Día de actividad!

Se abre sesión de **repaso** de los contenidos de la semana, para resolver dudas con el profesor.

Las actividades realizadas son **entregadas** y el profesor realizará un **cierre** discutiendo solución y aprendizajes.



Alumnos realizan **actividad** para aplicar los contenidos y resuelven dudas con ayudantes y profesor.

Actividades

- Serán de los **contenidos de la semana**.
- Los ayudantes **pasarán lista**. Si están marcados como ausentes, su actividad no será corregida.
- Tendrán hasta las **16:30** para entregar su actividad.
- Hay dos tipos de actividades: **formativas y sumativas**.

Actividades formativas

- Tienen por objetivo conseguir que el estudiante **practique** el contenido de la semana.
- **No** son parte del cálculo de nota del curso.
- Hay **6** en total durante el semestre.
- Serán corregidas **de forma colectiva** y se proveerá un *feedback* **general** detallando los resultados generales del curso.

Actividades sumativas

- Tienen por objetivo **evaluar** el aprendizaje individual de cada estudiante en un contenido esencial.
- Obtendrán una **nota** por su resultado.
- Hay **3** en total durante el semestre.
- Serán corregidas **individualmente** y se proveerá un *feedback* a **cada** alumno.
- Habrá una actividad **recuperativa opcional**.

Actividad recuperativa

- Tiene por objetivo **reemplazar** una inasistencia de actividad sumativa o mejorar una nota.
- Es una evaluación **opcional**.
- Será de temática **integradora**, acumulando varios contenidos del curso.
- La nota obtenida reemplazará **la peor nota** de actividades sumativas.
- Se realizará el **1 de julio, a las 09:00 AM**.

Tareas

Objetivo:

aplicar los contenidos
estudiados resolviendo
un problema complejo

Tareas

Lu	Ma	Mi	Ju	Vi	Sa	Do

Tareas

Lu	Ma	Mi	Ju	Vi	Sa	Do

Lunes: el enunciado de una tarea se **publica**.

Tareas

Lu	Ma	Mi	Ju	Vi	Sa	Do

Domingo (subsiguiente): termina el plazo para realizar la tarea.

Tareas

Lu	Ma	Mi	Ju	Vi	Sa	Do

Lunes (después de fin de plazo): se permite hasta **24 horas** de entrega **atrasada**. El tiempo de atraso tendrá un descuento correspondiente.

Tareas

Lu	Ma	Mi	Ju	Vi	Sa	Do

¿Y qué ocurre en las ayudantías y clase?

Tareas

Lu	Ma	Mi	Ju	Vi	Sa	Do

Clases: se realizarán **talleres o repasos** de contenido y **sala de ayuda** para realizar sus tareas.

Clases Talleres

- Clases guiadas por el profesor sobre tópicos de programación que **no son parte del programa** del curso.
- Son temas que **complementarán sus conocimientos y podrán usarlos** en sus tareas y actividades futuras.
- Estos temas también tendrán **material de estudio**, pero **no** es necesario que los revisen previo a la clase.
- Luego de la sesión de taller, se abrirá espacio de **sala de ayuda** para que realicen su **tarea** del momento.

Tareas

Lu	Ma	Mi	Ju	Vi	Sa	Do

Ayudantías: se realizarán **salas de ayuda** con los ayudantes o se revisarán temas **complementarios**.

Participación

Objetivo:

Incentivar el aprendizaje
y premiar la constancia
de trabajo

Actividades formativas: auto-evaluación

- Tienen por objetivo conseguir que el alumno **reflexione** sobre su rendimiento durante la actividad formativa.
- Tras terminar una actividad formativa, una **rúbrica** será publicada.
- El alumno **puede** enviar su auto-evaluación siguiendo tal rúbrica.
- Por **cada** auto-evaluación **enviada** (de una actividad formativa **realizada**) el alumno ganará **dos décimas**.
- El **total** de décimas ganadas se sumarán a la nota de la **peor actividad sumativa** del alumno (tras reemplazar la recuperativa).

Tareas: entrega de avance

- Tienen por objetivo incentivar la **realización anticipada** de las tareas y poder **entregar un *feedback*** rápido a los alumnos de sus avances.
- Tras la publicación del enunciado de una tarea, tendrán un plazo de unos días para enviar un **avance de su tarea**.
- Los avances entregados serán corregidos de forma **colectiva** y un *feedback general* será publicado.
- El enviar un avance **satisfactorio** agrega dos décimas a la nota de la tarea.
- El contenido del avance, plazo de envío y condiciones de satisfacción se detallarán en cada tarea.

Fechas de evaluaciones

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

Nota del curso (NC) y nota final (NF)

- **AC**: Promedio de tres actividades sumativas.
- **T**: Promedio ponderado de tareas: $(3 \times T_1 + 4 \times T_2 + 5 \times T_3) / 12$

$$NC = 0,4 \times AC + 0,6 \times T$$

$$NF = \begin{cases} NC & \text{si } AC \geq 3,95 \text{ y } T \geq 3,95 \\ \min(NC; 3,9) & \text{e.o.c.} \end{cases}$$

Aprobación

$$\mathbf{NF} = \begin{cases} \mathbf{NC} & \text{si } \mathbf{AC} \geq 3,95 \text{ y } \mathbf{T} \geq 3,95 \\ \min(\mathbf{NC}; 3,9) & \text{e.o.c.} \end{cases}$$

El alumno **aprobará** si **NF** es mayor o igual a **3,95**.

Evaluación de última instancia

En el caso extremo que **NC** sea mayor a 3,95, pero **AC** o **T** se encuentren entre **3,50 y 3,94**, el alumno tendrá opción de apelar y rendir una **última evaluación oral** para aprobar el curso.

Esta evaluación se realizará el **8 de julio** y busca identificar si el alumno logró desarrollar las competencias del curso. Será interrogado en varios contenidos del curso.

El aprobar esta evaluación permitirá que su nota final se calcule como: **NF = NC**

Detalles sobre notas

- La **inasistencia** a una actividad se interpreta como no realizada, tanto para **formativas como sumativas**.
- Solo **una nota** de actividad sumativa puede ser **reemplazada** por la Actividad Recuperativa.
- **No** habrá reemplazos de notas de **tareas**.
- Todas las notas del curso se calculan con **dos decimales**, excepto **NF** que se calcula con **un decimal**.

Correcciones y recorreciones

- Las notas de una evaluación se publicarán a **más tardar 15 días hábiles** después de haberse realizado.
- Tras la publicación de una nota, tendrán **una semana** para enviar a corregir mediante un formulario a publicar.
- La recorreción es una instancia de aprendizaje para **reflexionar sobre su desempeño**, no una oportunidad de subir nota.
- Se esperan solicitudes **bien fundamentadas y poco ambiguas, que ayude a entender** a la persona que le revise su recorreción.

Recorrecciones extremas

- De no estar de acuerdo con la respuesta de una corrección, se puede solicitar una revisión por parte de los profesores a través de un formulario.
- Puede que tome tiempo, ya que implica la revisión completa de la evaluación.
- Es la última instancia de apelación.

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.”*

También aplica la política de integridad académica del Departamento de Ciencia de la Computación (DCC), disponible como anexo en el programa del curso.

Integridad académica

- Está permitido el uso de código encontrado en internet u otra fuente de información similar, pero **debe indicarse la fuente de dicho código**.
- Lo mismo se aplica para código encontrado en el material del curso o ayudantías.
- De no hacerlo, se considera **plagio**.
- Todas las evaluaciones del curso se consideran **individuales**, a menos de que se indique lo contrario.
- Luego, compartir código para una evaluación se considera copia.

Consultas

- Consultas administrativas: **iic2233@ing.uc.cl**
- Consultas sobre los contenidos del curso, enunciados y pautas:
foro del curso, alojado en <https://github.com/IIC2233/syllabus/issues>
- Consultas personales: **correos de profesores o ayudantes**
 - Fernando: faflorenzano@uc.cl
 - Antonio: aaossa@uc.cl
 - Cristian: cruz@ing.puc.cl
 - Vicente: vidominguez@uc.cl

Recomendaciones

Recomendaciones

- Leer el enunciado apenas lo entreguen, para empezar a programar lo antes posible.
- Buscar más en Google.
- Estudiar e interactuar con 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

Recomendaciones

¡Aprovechen!

Aprovechen el **curso** para aprender lo más posible en herramientas que les servirán en el futuro.

Aprovechen a sus **profesores**, sus **ayudantes** y a sus **compañeros** (en el buen sentido).

Herramientas del curso



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

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

Python

- Es el lenguaje de programación que utilizaremos en el curso para aprender los contenidos.
- Es de alto nivel, de propósito general y sumamente popular.

Recientemente se publicó su versión 3.7, pero en el curso **nos mantendremos con la 3.6.x.**

Guía de estilo

En el curso utilizaremos una guía de estilo para programar.

```
a = 1
a2 = 2
mivar = a+ a2
aa = mivar **2
def miFUNCION(b1 , b2):
    return b1 - b2
```

```
numero_1 = 1
numero_2 = 2
suma = numero_1 + numero_2
cuadrado = suma ** 2
def resta(b1, b2):
    return b1 - b2
```

“El código se lee más veces de lo que se escribe, y es otro el que lo va a leer”

PEP8

- *Python Enhancement Proposal* 8 es la guía de estilo de Python
- Se usa para hacer más legible y consistente el código
- En las tareas se espera el respeto por algunos aspectos de esta guía de estilo
- <https://www.python.org/dev/peps/pep-0008/>

PEP8: algunas reglas

- **Nombres de variables descriptivos.**
- *Imports* al comienzo del módulo.
- Un espacio después de “ , ”.
- Espacios a cada lado de operadores.
- Líneas de largo máximo de 80 caracteres (incluyendo espacios).
- No usar *tabs*. Usar (4) espacios para indentar.

snake_case y CamelCase

- **snake_case** para nombres de variables, atributos, funciones y métodos.
- **CamelCase** para nombres de clases.

```
class ClaseDeEjemplo:

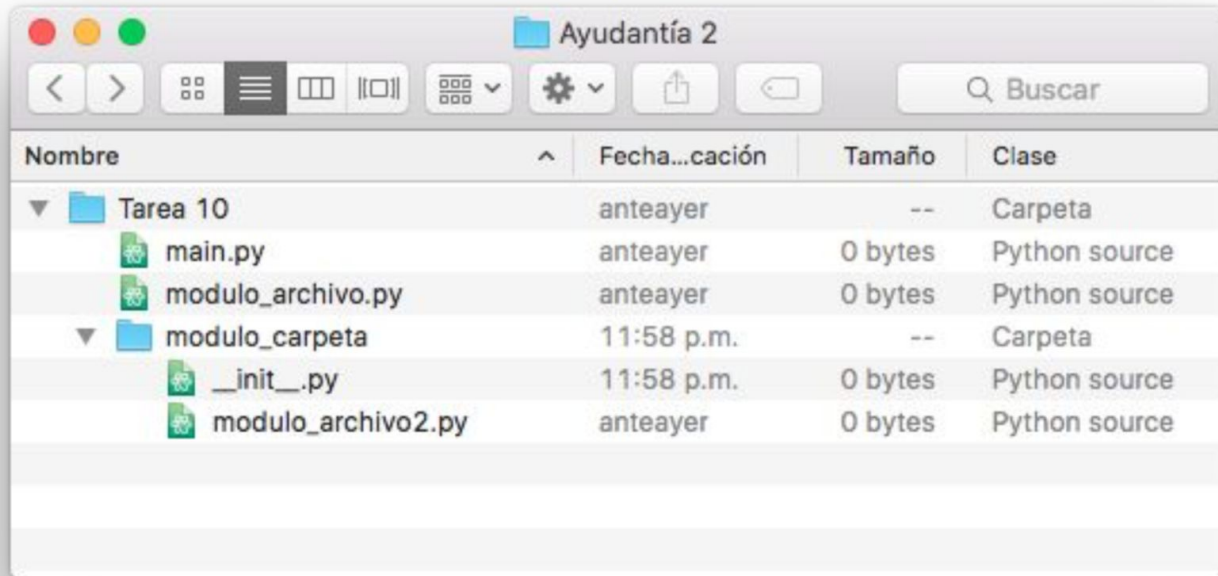
    def __init__(self, parametro):
        self.variable_de_ejemplo = parametro

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

Modularización

- Cuando un programa crece, se hace inviable mantenerlo en un solo archivo:
 - El mantenimiento es difícil
 - El trabajo en equipo es difícil
 - Es desordenado
- Un módulo es un archivo de Python normal (.py), y puede tener:
 - Variables
 - Métodos
 - Clases

Modularización



Cómo usar módulos

Importación completa

```
import modulo_archivo
```

```
if __name__ == '__main__':
```

```
    variable_tipica = modulo_archivo.VALOR_FIJO
```

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

```
    modulo_archivo.funcion()
```


Cómo usar módulos

Importación completa con un alias

```
import modulo_archivo as ma
```

```
if __name__ == '__main__':
```

```
    variable_tipica = ma.VALOR_FIJO
```

```
    objeto_tipico = ma.Clase()
```

```
    ma.funcion()
```

Cómo usar módulos

Importación parcial

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

```
if __name__ == '__main__':  
    variable_tipica = VALOR_FIJO  
  
    objeto_tipico = Clase()  
  
    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 `if` visto en los ejemplos anteriores:

```
# Código del modulo
```

```
if __name__ == '__main__':
```

```
    # Código que no se ejecuta al importar
```

Cómo **NO** usar módulos

Importación completa sin referencia al módulo

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



Cómo **NO** usar módulos

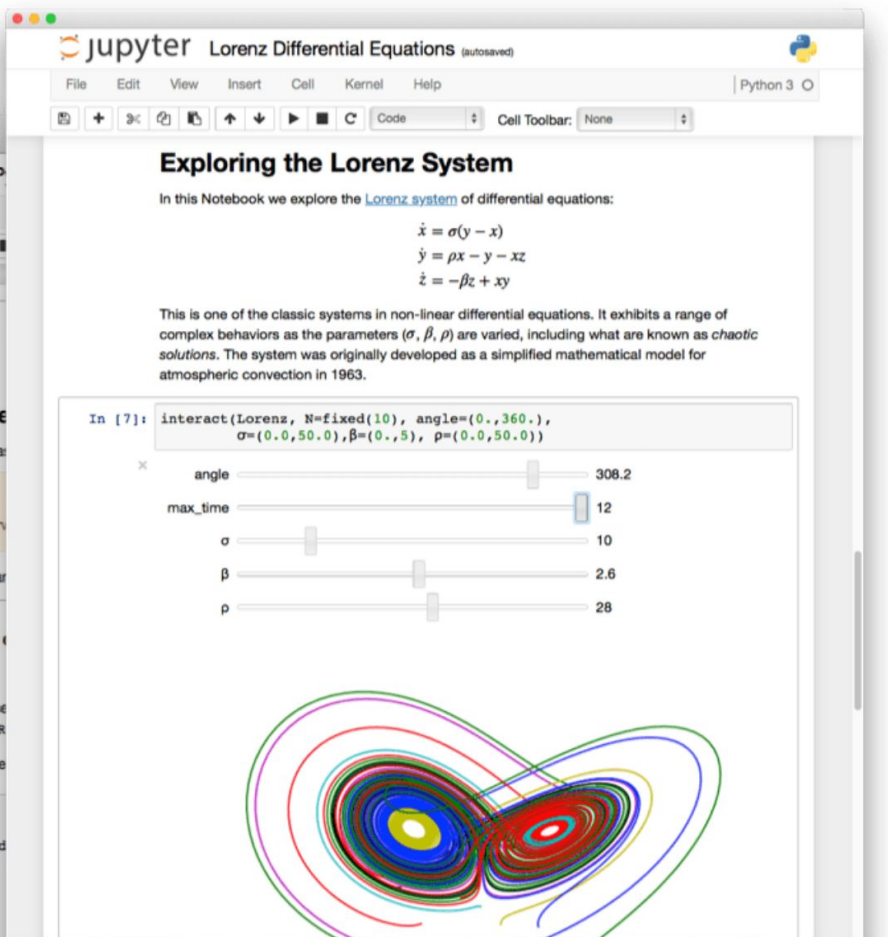
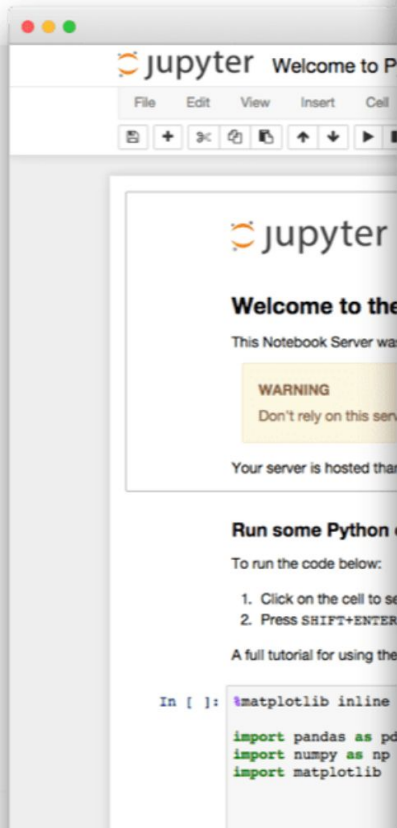
- **Evitar** crear módulos que se llamen igual a los que vienen incluidos en Python.
- ¿Cómo Python busca los módulos?
 - a. Módulo de la librería estándar
 - b. Módulo en la misma carpeta
 - c. Módulo en el directorio de instalación



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

Jupyter Notebook

- Es una aplicación web que permite crear documentos interactivos con código, gráficos y texto explicativo.
- Es el formato de los apuntes del curso.
- Se recomienda **bajar los apuntes e interactuar** con el código, no solo leerlo desde la página.
- **No deben usarlo para programar sus actividades ni tareas.**
- Instalación: <https://jupyter.org/install.html>



Google +  stackoverflow

<https://google.com/>

<https://stackoverflow.com/>

¿Cómo buscar soluciones?

¡En ingles!

python [versión] [librería] [duda]



¿Cómo imprimir una cola con Python?



Python 3.6 collections print queue



¿Cómo buscar soluciones?

¡En ingles!

python [versión] [error]



`NameError: name "variable" is not defined`



`NameError: name * is not defined`





python3.5 NameError: name * is not defined



Todos

Imágenes

Maps

Imágenes

Noticias

Más

Preferencias

Herramientas

Google encontró 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
File "Q: python NameError: name 'file' is not defined But ...

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

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

3 abr. 2014 - NameError: name 'value' is not defined ... A variable defined in a function isn't visible outside the function. ... answered Apr 5 '14 at 2:39.

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

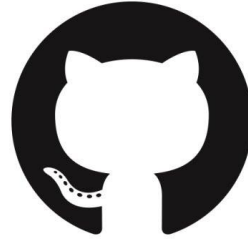
<https://stackoverflow.com/.../nameerror-global-name-unicode-is-...> ▼ Traducir esta página

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 ...



git

+



GitHub

<https://git-scm.com/>

<https://github.com/>

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.

- Permite revisar distintas versiones en cualquier momento.
- Permite controlar los cambios que se aplican sobre un proyecto.
- Permite programar en paralelo y luego juntar los resultados.
- Permite tener copias de apoyo de programas.
- Permite un trabajo en equipo mucho más fluido.

Se usa **masivamente en la vida real**,
incluso por **empresas** gigantes.

Será el medio de entrega de evaluaciones
oficial del curso.

GitHub

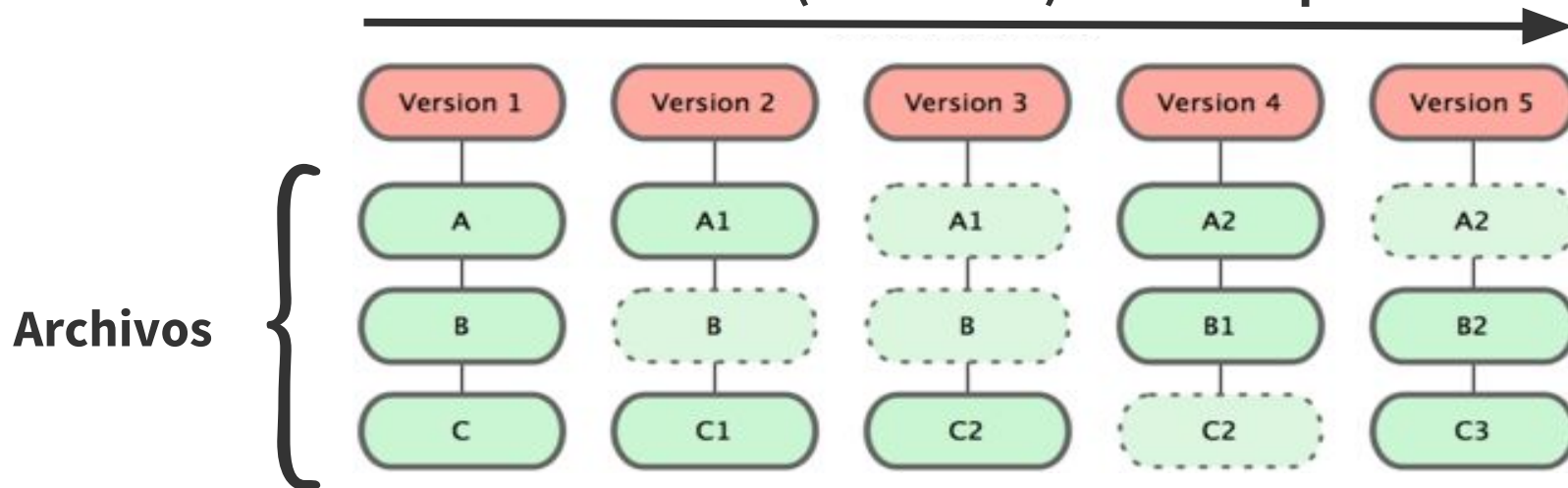
- GitHub es una plataforma para alojar proyectos, usando el sistema de control de versiones Git.
- En noviembre 2018, GitHub tenía 31 millones de usuarios y 100 millones de repositorios albergados.¹
- Provee una interfaz web que permite visualizar y administrar proyectos controlados con Git.

1: <https://github.com/about>

Conceptos

1. **Commit o versión:** un estado, que contiene archivos y carpetas con cierto contenido.

Versiones (o commits) en el tiempo



Conceptos

2. ***Working directory***: carpeta local con la que se trabaja directamente, creando, cambiando o eliminando archivos.
3. ***Staging area***: creaciones, modificaciones o eliminaciones de archivos que serán incluidas en un nuevo *commit*.
4. **Repositorio local (o repo)**: carpeta local que contiene todos los *commits*.
5. **Repositorio remoto**: servidor, en nuestro caso GitHub, que contiene todos los *commits*. **¡Este es el lugar que los ayudantes revisarán!**

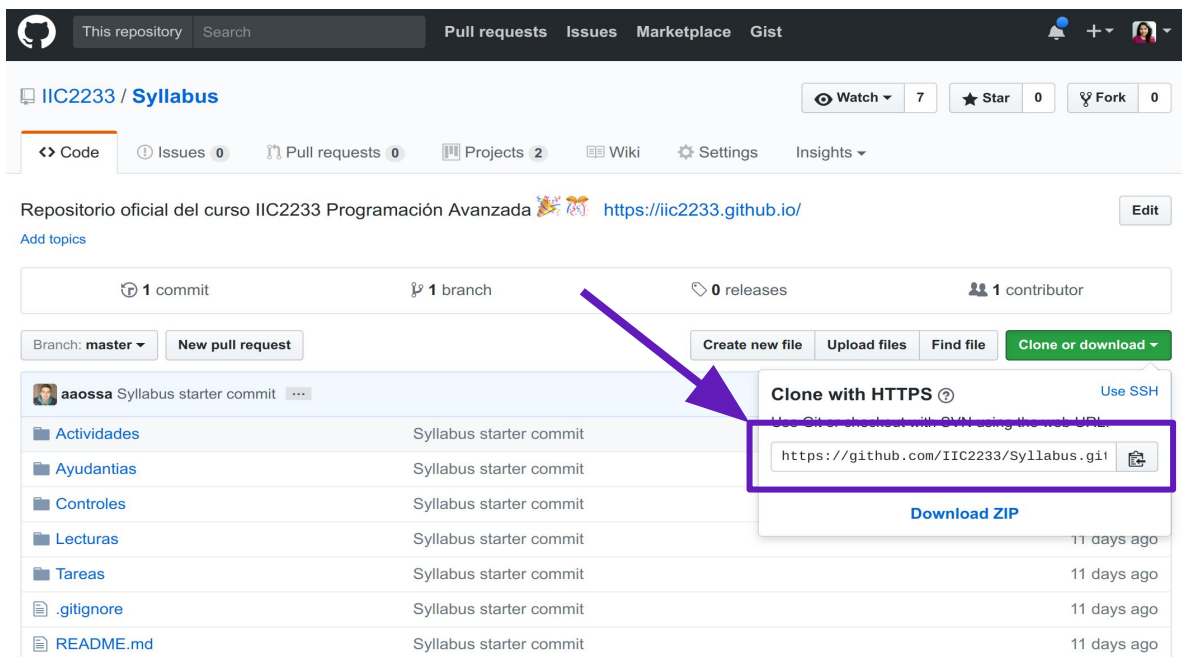
Lo primero es lo primero

Clonaremos un repositorio

- Para empezar a hacer cosas con un repositorio, hay que tener uno.
- Podríamos crear uno, pero mejor usemos el que les dieron los ayudantes.

Obtener dirección

1. Haberse registrado en el curso correctamente
2. Ir a <https://github.com/IIC2233/<mi-usuario>-iic2233-2019-1>



The screenshot shows the GitHub interface for the repository 'IIC2233 / Syllabus'. The repository has 7 watches, 0 stars, and 0 forks. The main content area shows the repository description: 'Repositorio oficial del curso IIC2233 Programación Avanzada' with a link to 'https://iic2233.github.io/'. Below this, there are statistics: 1 commit, 1 branch, 0 releases, and 1 contributor. The 'Clone or download' button is highlighted with a purple arrow pointing to the 'Clone with HTTPS' option, which is also highlighted with a purple box. The box contains the URL 'https://github.com/IIC2233/Syllabus.git' and a 'Download ZIP' button. The repository files list includes 'Actividades', 'Ayudantias', 'Controles', 'Lecturas', 'Tareas', '.gitignore', and 'README.md', all with a 'Syllabus starter commit' and a timestamp of '11 days ago'.

File	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

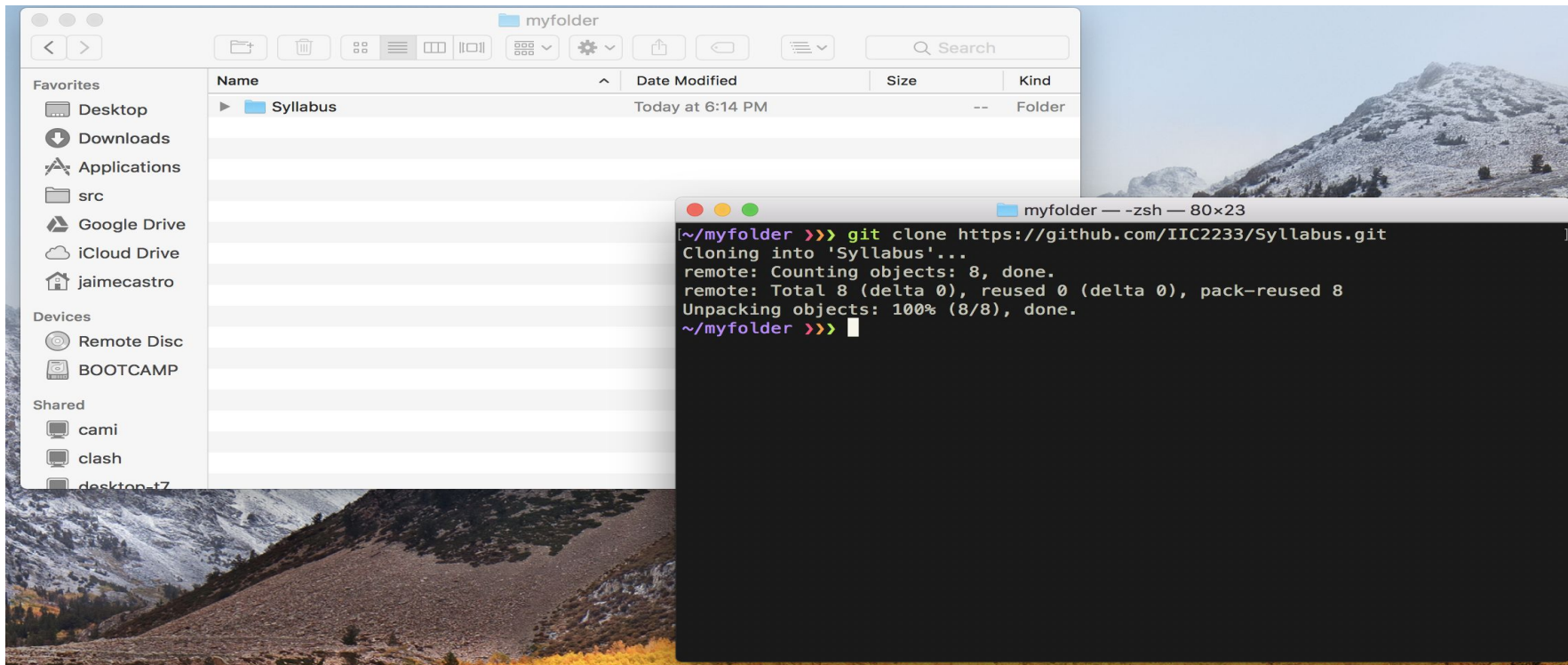
Clonar el repositorio

Escribir en la consola

```
git clone url_que_copiaron
```

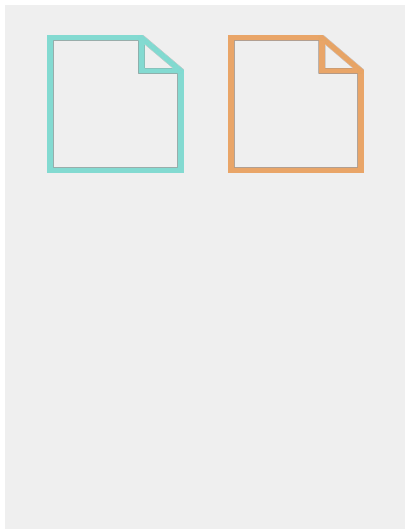
Recuerden estar en la carpeta en la que quieren mantener el repo.

Clonar el repositorio

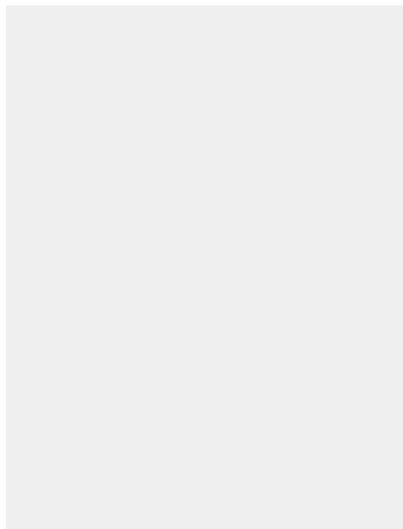


Estado inicial

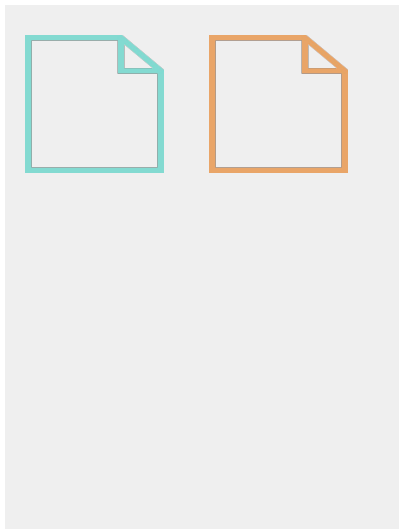
Working directory



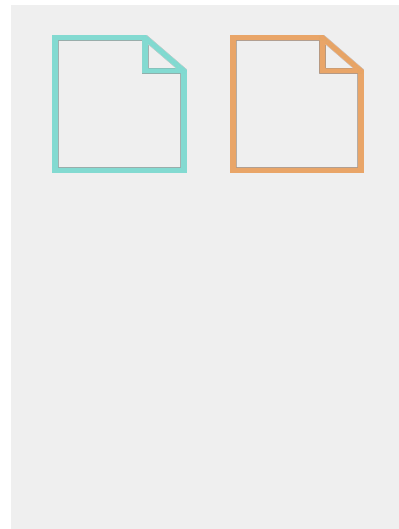
Staging area



Repositorio local
(Última versión)



Repositorio remoto
(Última versión)



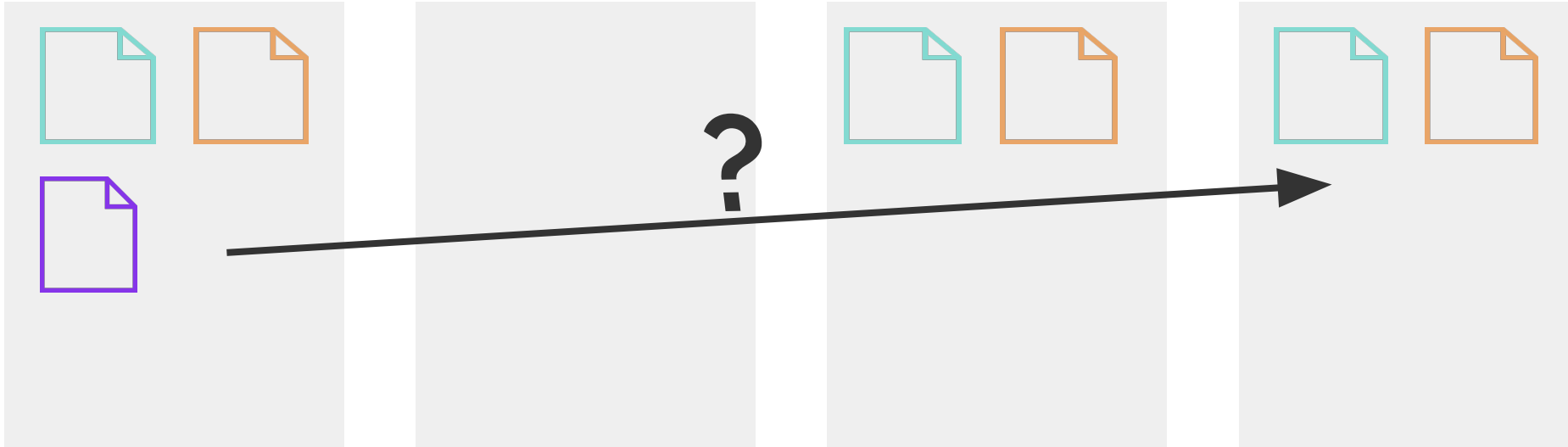
Creé un nuevo archivo. ¿Cómo lo “subo” al repositorio remoto?

Working directory

Staging area

Repositorio local
(Última versión)

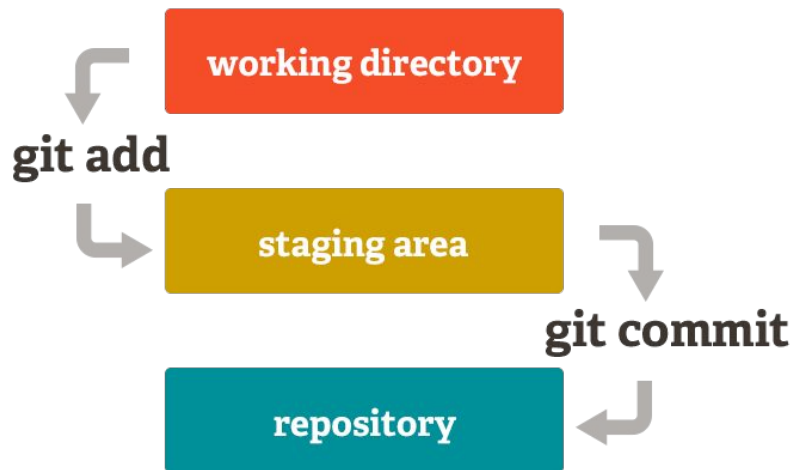
Repositorio remoto
(Última versión)



Creé un nuevo archivo. ¿Cómo lo “subo” al repositorio remoto?

1. Crear una versión con el nuevo archivo en el repositorio local.
2. Hacer que mi repositorio remoto tenga las versiones que tengo en el repositorio local.

Crear una versión



Crear nueva versión

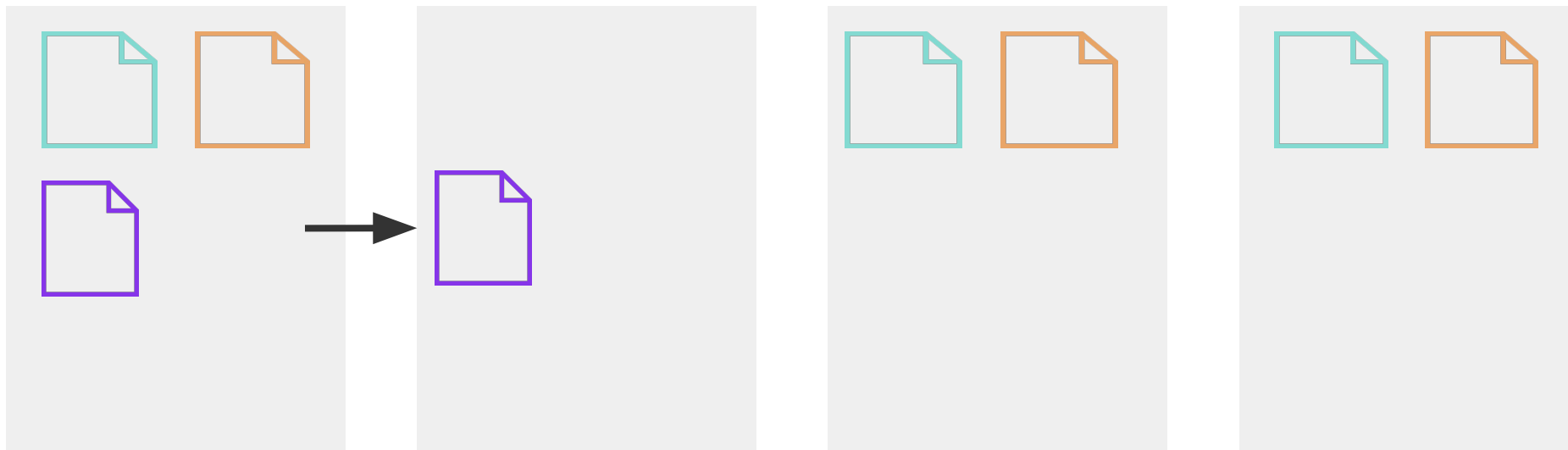
Primero, agregamos el archivo al *staging area*

Working directory

Staging area

Repositorio local
(Última versión)

Repositorio remoto
(Última versión)



`git add archivo.py`

Crear nueva versión

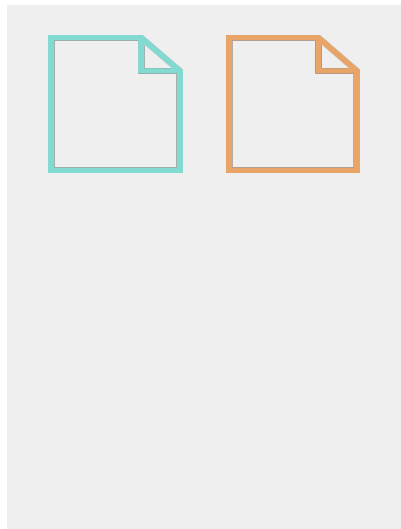
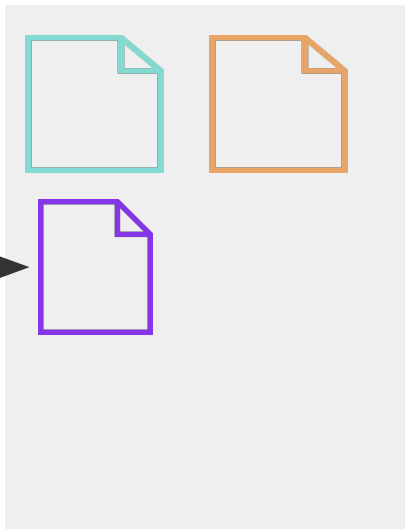
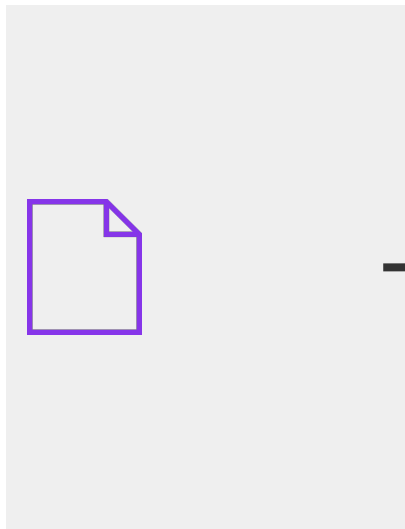
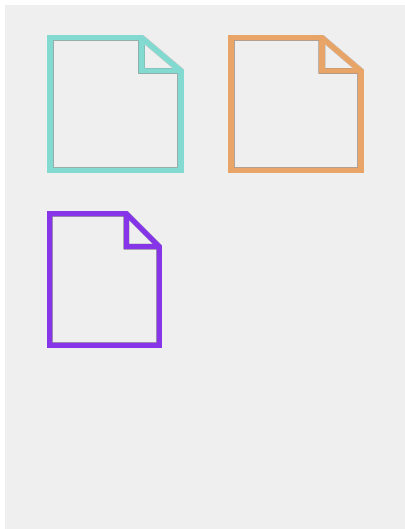
Ahora sí, creamos la versión

Working directory

Staging area

Repositorio local
(Última versión)

Repositorio remoto
(Última versión)



git commit -m “Mensaje descriptivo”

Crear nueva versión

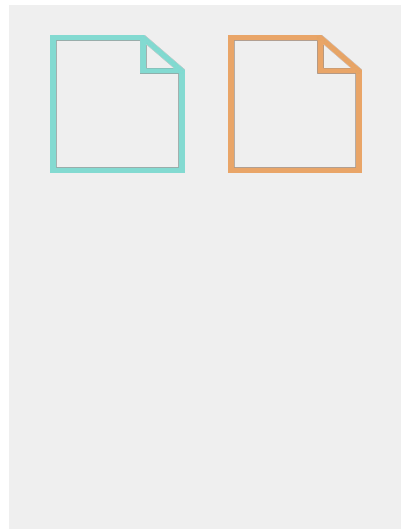
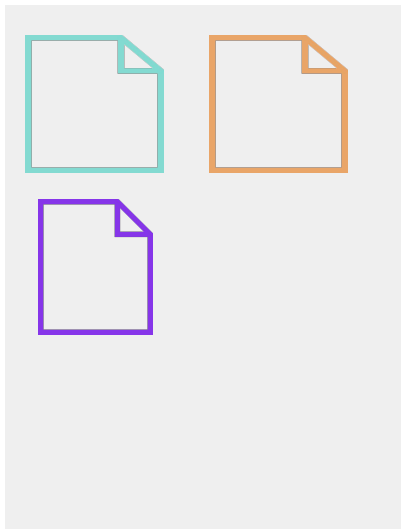
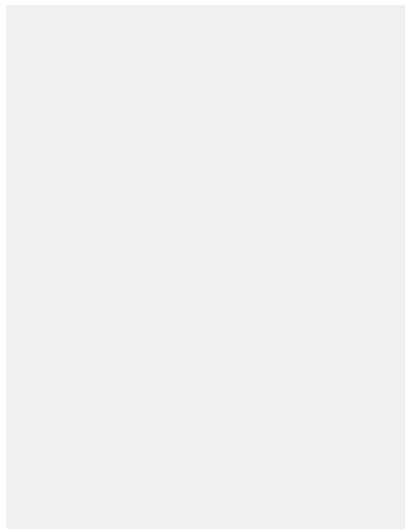
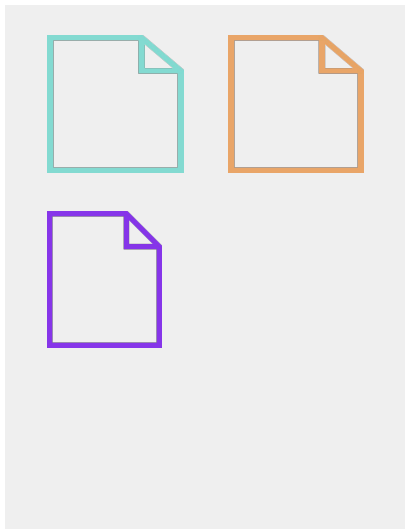
Así queda con la nueva versión creada

Working directory

Staging area

Repositorio local
(Última versión)

Repositorio remoto
(Última versión)



Los mensajes son **muy importantes**. Son una ayuda a ustedes en el futuro.

Revisen esta [guía de estilo](#).

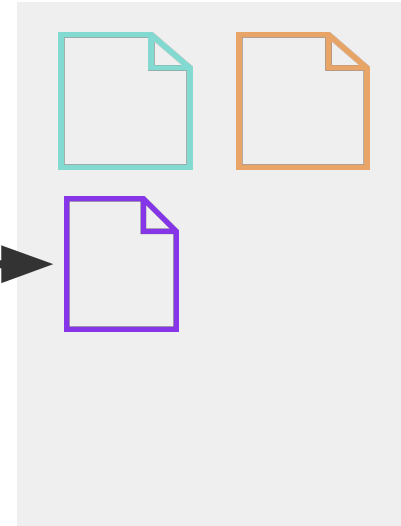
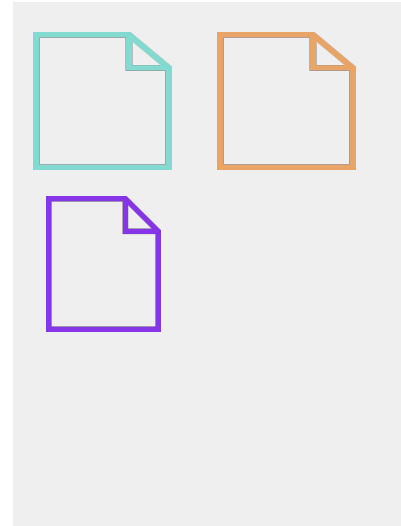
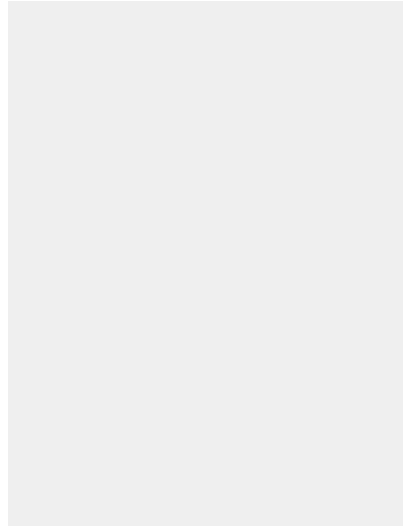
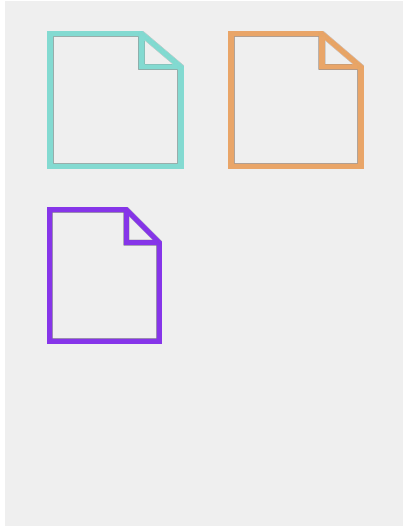
Subir lo que tengo en el repo local a GitHub

Working directory

Staging area

Repositorio local
(Última versión)

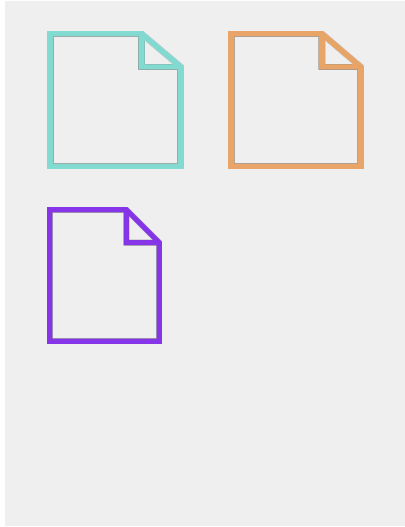
Repositorio remoto
(Última versión)



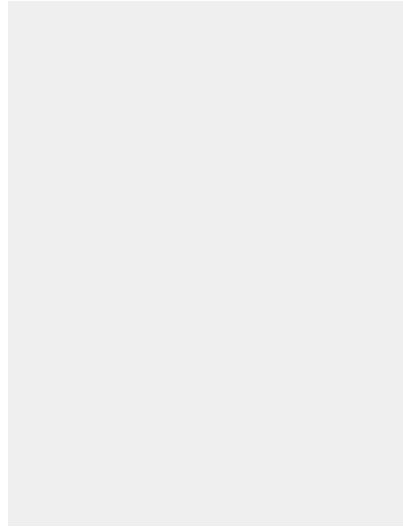
`git push`

Subir lo que tengo en el repo local a GitHub

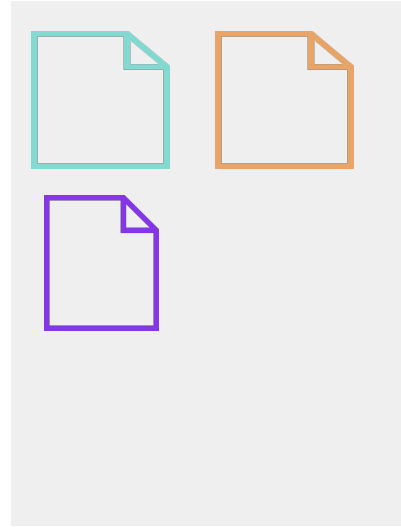
Working directory



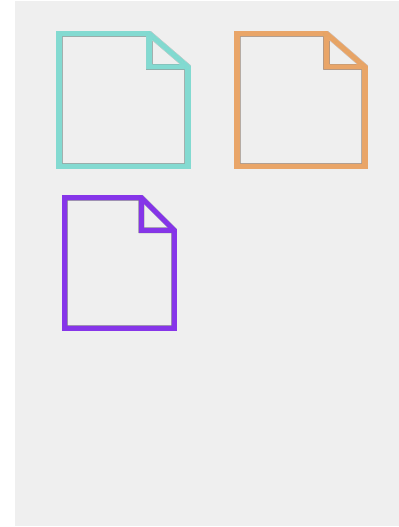
Staging area



Repositorio local
(Última versión)



Repositorio remoto
(Última versión)



Para las entregas, se toma **la hora del *push* a GitHub.**

Sacar algo del *staging area*

Son las 16:28. Las instrucciones dicen que no debo subir el archivo `VeryHeavyFile.txt` que pesa 100 MB.

Hice `git add --all` y solo me queda un minuto para poder subir la actividad.



```
git reset HEAD file_name
```

Ya hice *commit*



```
git reset HEAD~
```



IN CASE OF FIRE

 git commit

 git push

 leave building

Siempre hagan
commit y push
de su trabajo.

- Cada vez que avancen en algo importante de su actividad o tarea.
 - Si llevan programando más de media hora.
 - Cada vez que paren de programar para dedicarse a otra cosa.
-

De verdad:
Siempre hagan
commit y push
de su trabajo.

- Tener su trabajo en GitHub es una copia de seguridad.
 - *Shit happens:*
 - Accidentes con líquidos.
 - Robos en Deportes.
 - Fallas de *hardware* o *software*.
 - Cortes de internet.
 - Echar a perder la tarea.
 - Y muchas otras cosas.
-

Sitios útiles

- www.git-scm.com
- [Una metaguía de Git](#)
- [Una guía de estilo de *commits*](#)

Actividad en clase 0 (AC00)

ACoo

<https://github.com/IIC2233/syllabus>

1. En el syllabus, vayan a la carpeta “Actividades” y descarguen el enunciado de la actividad (AC00).
2. Trabajen en ella hasta las 16:30.