



Sesión 2 

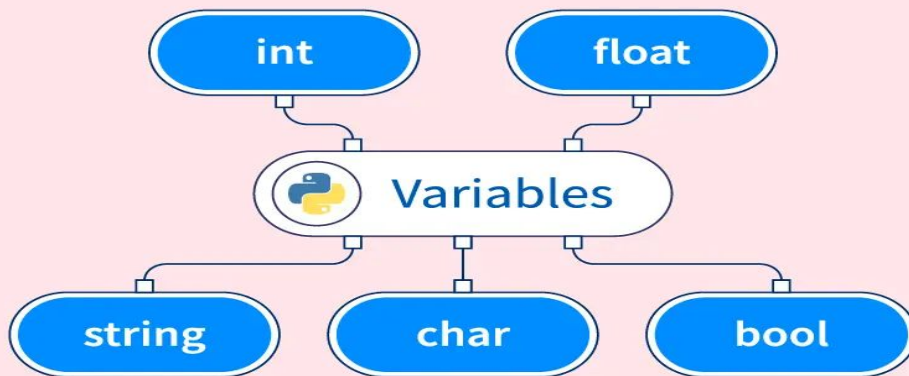
# Módulo 1 – semana 2

Descubre el Universo de  
Python:  
Tipos de Variables y Git





En programación, una **variable** es un contenedor que usamos para almacenar información o datos que pueden cambiar durante la ejecución del programa. En Python, las variables son fáciles de trabajar, ya que no es necesario declarar su tipo explícito.



# Python Data Types



Name	Type	Description
Integers	int	Whole numbers: 1,100
Floating point	float	Numbers with a decimal point: 1.1, 100.0
Strings	str	Ordered sequence of character: "hello"
Lists	list	Ordered sequence of objects: [10,"hello"]



# Python Variables



## CODE

```
a = 5  
b = "John"  
abc = {1,2,3,4,5}  
mylist = ["x","yy","zzz"]
```

```
print(a)  
print(b)  
print(abc)  
print(mylist)
```

## OUTPUT



```
x = y = z = 100  
print(x)  
print(y)  
print(z)
```



```
a, b, list1 = 10, 20, [1,2,3]  
print(a)  
print(b)  
print(list1)
```





## Assignment

age = 25

Variable      Value

```
age = 25
# print the variable age
print(age) # 25
```



```
age = 25
# variable with quotation
# prints the string
print('age')
# Output: age
```





## Utilidades

- Control de versiones
- Colaboración
- Almacenamiento remoto
- Documentación y visibilidad
- Integraciones y automatización

### **git init**



Creates a new local repository in the current directory

### **git clone**



Copies an existing remote repository to your local machine.

### **git status**



Shows the state of your working directory and staging area.

### **git add**



Adds changes in your working directory to the staging area, which is a temporary area where you can prepare your next commit.

### **git commit**



Records the changes in the staging area as a new snapshot in the local repository, along with a message describing the changes.

### **git push**



Uploads the local changes to the remote repository usually on a platform like GitHub or GitLab.

### **git pull**



Downloads the latest commits from a remote repository and merges them with your local branch.

### **git branch**



Lists, creates, renames, or deletes branches in your local repository. A branch is a pointer to a specific commit.

### **git checkout**



Switches your working directory to a different branch or commit, discarding any uncommitted changes

### **git merge**



Combines the changes from one branch into another branch, creating a new commit if there are no conflicts

### **git diff**



Shows the differences between two commits, branches, files, or the working directory and the staging area.

### **git log**



Shows the history of commits in the current branch, along with their messages, authors, and dates.

# iRally GitHub!







### Preparación del repositorio (5 min)

El líder del proyecto:

- Crea un repositorio en GitHub llamado `lista_de_tareas`.
- Agrega un archivo `README.md` con el texto:  
*"Este repositorio contiene una lista de tareas colaborativas."*
- Comparte el enlace del repositorio con los compañeros.



### Clonar el repositorio (5 min)

Cada alumno clona el repositorio en su computadora ejecutando:

```
bash
git clone
https://github.com/<usuario>/lista_de_tareas.git

cd lista_de_tareas
```



### Editar el archivo y agregar tareas (10 min)

Cada alumno:


Abre el archivo `README.md`.

Agrega su nombre y una tarea en formato de lista.

Por ejemplo:

markdown

- Ana: Lavar los platos
- Juan: Regar las plantas



Guarda los cambios y realiza un commit:

```
bash
git add README.md
git commit -m "Agrego tarea de <nombre>"
```

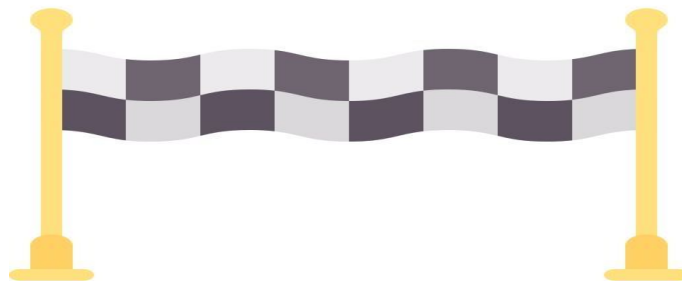


### Subir cambios al repositorio remoto (5 min)

Cada alumno sube sus cambios directamente al repositorio:

bash

```
git push origin main
```



### Verificar en GitHub (5 min)

- Cada alumno revisa en GitHub para confirmar que su tarea aparece correctamente.
- En caso de errores o conflictos, el líder del proyecto ayuda a resolverlos.

¡Nos vemos pronto!

