


Ejemplo solución tarea

PIA03 - Probando entorno de programación Colab

Vamos a notebook de introducción y ejecutamos las celdas de código

 Te damos la bienvenida a Colaboratory

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Compartir Configuración Perfil

Índice

Primeros pasos

Ciencia de datos

{x} Aprendizaje automático

Más recursos

Ejemplos destacados


+ Sección

+ Código + Texto Copiar en Drive

Conectar Editar

Te damos la bienvenida a Colab

Si ya conoces Colab, echa un vistazo a este vídeo para obtener información sobre las tablas interactivas, la vista del historial de código ejecutado y la paleta de comandos.



¿Qué es Colaboratory?

Colab, también conocido como "Colaboratory", te permite programar y ejecutar Python en tu navegador con las siguientes ventajas:

- No requiere configuración
- Acceso a GPUs sin coste adicional
- Permite compartir contenido fácilmente

analizar y visualizar datos. La celda de código de abajo utiliza **NumPy** para generar datos aleatorios y **Matplotlib** para visualizarlos. Para editar el código, solo tienes que hacer clic en la celda.

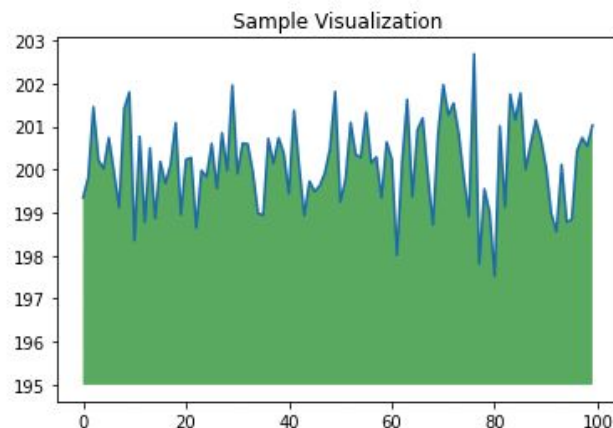
✓
0 s

```
import numpy as np
from matplotlib import pyplot as plt

ys = 200 + np.random.randn(100)
x = [x for x in range(len(ys))]

plt.plot(x, ys, '-')
plt.fill_between(x, ys, 195, where=(ys > 195), facecolor='g', alpha=0.6)

plt.title("Sample Visualization")
plt.show()
```

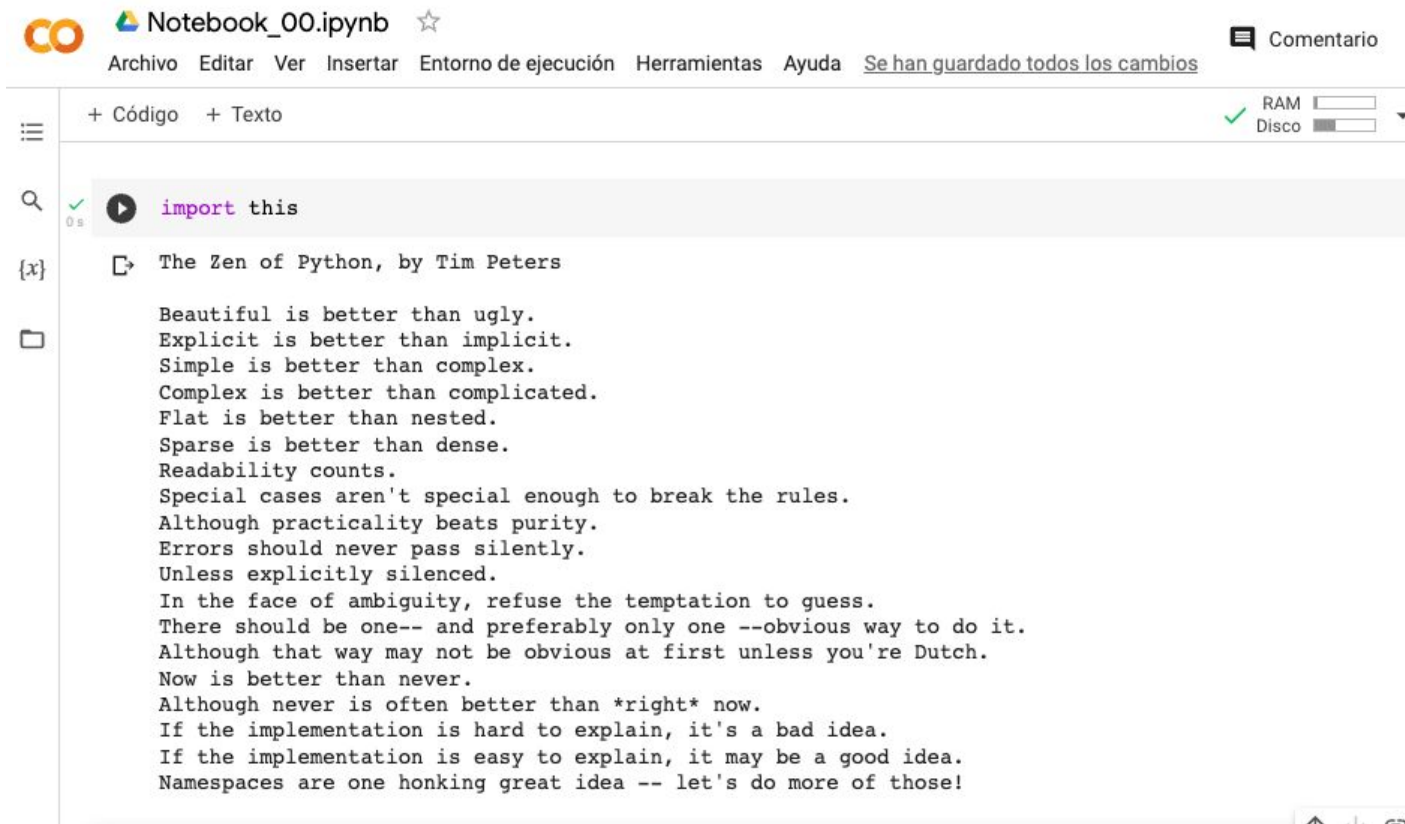


✓

0 s

completado a las 14:04

Creamos nuevo notebook, lo nombramos y ejecutamos huevo de pascua de Python:



The screenshot shows the Jupyter Notebook interface. At the top, the notebook is named "Notebook_00.ipynb". The menu bar includes "Archivo", "Editar", "Ver", "Insertar", "Entorno de ejecución", "Herramientas", "Ayuda", and a status message "Se han guardado todos los cambios". On the right, there is a "Comentario" button and resource usage indicators for RAM and Disco. The left sidebar shows icons for the table of contents, search, and file explorer. The main area displays a code cell with the text "import this", which has been executed, resulting in the output of "The Zen of Python, by Tim Peters".

```
import this
```

The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!

Generamos las primeras celdas de código

```
✓ [2] ciudades = ['Madrid', 'Barcelona', 'Valencia', 'Málaga', 'San Sebastian', 'Sevilla', 'Alicante', 'Toledo', 'Ávila', 'Gijón']  
0s print(ciudades)  
  
['Madrid', 'Barcelona', 'Valencia', 'Málaga', 'San Sebastian', 'Sevilla', 'Alicante', 'Toledo', 'Ávila', 'Gijón']  
  
✓ [3] artistas = ['Muse', 'One Direction', 'Dua Lipa', 'Rosalía', 'Ludovico Einaudi', 'Alejandro Sanz', 'Adele', 'Miley Cyrus', 'Taylor Swift', 'Maneskin']  
0s print(artistas)  
  
['Muse', 'One Direction', 'Dua Lipa', 'Rosalía', 'Ludovico Einaudi', 'Alejandro Sanz', 'Adele', 'Miley Cyrus', 'Taylor Swift', 'Maneskin']  
  
✓ [8] actuaciones = dict(zip(artistas,ciudades))  
0s print(actuaciones)  
  
{'Muse': 'Madrid', 'One Direction': 'Barcelona', 'Dua Lipa': 'Valencia', 'Rosalía': 'Málaga', 'Ludovico Einaudi': 'San Sebastian', 'Alejandro Sa  
  
✓ [19] print('Estos son los conciertos para esta semana: ')  
0s for artista in actuaciones:  
    print(artista, 'actua en: ',actuaciones[artista])
```

```
Estos son los conciertos para esta semana:  
Muse actua en: Madrid  
One Direction actua en: Barcelona  
Dua Lipa actua en: Valencia  
Rosalía actua en: Málaga  
Ludovico Einaudi actua en: San Sebastian  
Alejandro Sanz actua en: Sevilla  
Adele actua en: Alicante  
Miley Cyrus actua en: Toledo  
Taylor Swift actua en: Ávila  
Maneskin actua en: Gijón
```

Y programamos la rotación de artistas por ciudades:

```
0 s  ▶ ciudades2 = ciudades[-1:] + ciudades[:-1]
      print(ciudades2)
```

```
['Gijón', 'Madrid', 'Barcelona', 'Valencia', 'Málaga', 'San Sebastian', 'Sevilla', 'Alicante', 'Toledo', 'Ávila']
```

```
0 s  [16] actuaciones2 = dict(zip(artistas,ciudades2))
      print(actuaciones2)
```

```
{'Muse': 'Gijón', 'One Direction': 'Madrid', 'Dua Lipa': 'Barcelona', 'Rosalía': 'Valencia', 'Ludovico Einaudi': 'Málaga', 'Alejandro Sanz': 'San Sebastian', 'Adele': 'Sevilla', 'Miley Cyrus': 'Alicante', 'Taylor Swift': 'Toledo', 'Måneskin': 'Ávila'}
```

```
0 s  [18] print('Estos son los conciertos para esta semana: ')
      for artista in actuaciones2:
          print(artista, 'actua en: ',actuaciones2[artista])
```

```
Estos son los conciertos para esta semana:
Muse actua en:  Gijón
One Direction actua en:  Madrid
Dua Lipa actua en:  Barcelona
Rosalía actua en:  Valencia
Ludovico Einaudi actua en:  Málaga
Alejandro Sanz actua en:  San Sebastian
Adele actua en:  Sevilla
Miley Cyrus actua en:  Alicante
Taylor Swift actua en:  Toledo
Måneskin actua en:  Ávila
```

Pasamos a añadir las celdas de texto usando lenguaje de marcación:

¶ T B I < > 🔗 🖼️ ☰ ☷ ☸ ⋯ ψ 😊 ☰

```
# Código para generar agenda de conciertos

```

¶ T B I < > 🔗 🖼️ ☰ ☷ ☸ ⋯ ψ 😊 ☰

```
## 1.- Huevo de Pascua de Python "import this"
Este sencillo poema habla sobre las principales características de estilo de
Python, destacando la **simplicidad** pero también la **legibilidad** del
código.
puedes conocer más sobre el estilo pythonista en la [web de PEP 8](https://peps.
python.org/pep-0008/).
```

Que va quedando así:

- ▾ Código para generar agenda de conciertos



- ▾ 1.- Huevo de Pascua de Python "import this"

Este sencillo poema habla sobre las principales características de estilo de Python, destacando la **simplicidad** pero también la **legibilidad** del código. puedes conocer más sobre el estilo pythonista en la [web de PEP 8](#).

```
import this
```

The Zen of Python, by Tim Peters

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
```


Vamos a notebook de introducción y ejecutamos las celdas de código

2.- Agenda de la segunda semana

Hacemos rotar a los artistas por las ciudades utilizando el **slicing** de listas

▼ 2.- Agenda de la segunda semana

Hacemos rotar a los artistas por las ciudades utilizando el *slicing* de listas

```
ciudades2 = ciudades[-1:] + ciudades[:-1]
print(ciudades2)
```

['Gijón', 'Madrid', 'Barcelona', 'Valencia', 'Málaga', 'San Sebastian', 'Sevilla', 'Alicante', 'Toledo', 'Ávila']

Enlace al notebook:

https://colab.research.google.com/drive/1jcM8BdlO3psc_oV-Ub-_9k9-YNSLotZw?usp=sharing