

- METODOS NUMERICOS
- JUAN FRNANCISCO PINTO ANDRANGO
- Documentación en Github
- Ecuaciones!
- En Notebook de python
- Viñeta 1
- Viñeta 2

## Documentar en lenguaje de *programación*

- identificar un lenguaje

```
def suma(x1:int, x2:int):
    return x1 + x2

function y = suma(x1, x2)
    y = x1+ x2;
```

## Micelánea

- Cambia distribución de teclado windows + spacio
- mi computador trabaja con teclado Ingles Estados Unidos y Español Ecuador

## Redacción de ecuaciones en Markdown

- Comando para previsualizar: ctrl+shift+v

## Ejemplo

Fórmula para calcular las raices de la ecuación cuadrática.  $x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$

$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$

$x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$

$\pi, \alpha, \beta, \hat{y}, \leftarrow$

# Ejercicio

Escribir la ecuación Navier Stokes.

$$\rho \left( \frac{\partial u}{\partial t} + v \cdot \nabla v \right) = -\nabla p + \nabla \cdot T + f$$

\*ejemplos de caracteres especiales

$$\rho \partial \rho \delta$$

$$\rho \frac{D \overrightarrow{V}}{Dt} =$$

$$\sum_i^4 \square$$

\*adjuntarlo al taller final .

## Taller!

- Se debe presentar:
  - Notebook
  - Enlace al repositorio

## Ejercicio 1

La sumatoria  $1 + 1/2 + 1/4 + 1/8 \dots$  tal que el error absoluto  $e_{abs} < 10^{-1}$ .

```
sumatoria = 0
n = 0
error_abs = float('inf')

while error_abs > 1e-6:
    termino = 1 / (2 ** n)
    sumatoria += termino
    error_abs = abs(termino / 2)
    n += 1

print(f"La sumatoria es: {sumatoria}") # hacemos un print para mostrar el resultado
```

## Ejercicio 2 (Bubble sort)

```
import random
```

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        swapped = False
        for j in range(0, n - i - 1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
            swapped = True
        if not swapped:
            break
    return arr
```

```
v2 = [-1, 0, 4, 5, 6, 7] sorted_v2 = bubble_sort(v2) print("v2_sorted:", sorted_v2)
v3 = [random.randint(-200, 145) for _ in range(100.000)] sorted_v3 = bubble_sort(v3)
print("v3_sorted:", sorted_v3)
```

## Corrida de escritorio

$v_1 = [3, 2, 5, 8, 4, 1]$

| i | Vector                   |
|---|--------------------------|
| 0 | \$ [3, 2, 5, 8, 4, 1] \$ |
| 1 | \$ [3, 2, 5, 4, 1, 8] \$ |
| 2 | \$ [3, 2, 4, 1, 5, 8] \$ |
| 3 | \$ [3, 2, 1, 4, 5, 8] \$ |

...

Resultado final:

$v_{sorted} = [1, 2, 3, 4, 5, 8]$

Casos de prueba:

- $v_2 = [-1, 0, 4, 5, 6, 7]$
- $v_3$  100\_000 número aleatorios entre -200 y 145.

## Insertar Fotografia

Pseudocode:

```

1  procedure iterative          (n: nonnegative integer)
2  if n=0 then
3      return 0
4  else
5      x := 0
6      y := 1
7      for i := 1 to n - 1
8          z := x + y
9          x := y
10         y := z
11  return y

```

```
import random

def bubble_sort(arr):
    n = len(arr)
```

```

counter = 0
for i in range(n):
    swapped = False
    for j in range(0, n - i - 1):
        counter += 1
        if arr[j] > arr[j+1]:
            arr[j], arr[j+1] = arr[j+1], arr[j]
            swapped = True
    if not swapped:
        break
return arr, counter

v2 = [-1, 0, 4, 5, 6, 7]
sorted_v2, counter_v2 = bubble_sort(v2)
print("v2_sorted:", sorted_v2)
print("Comparaciones en v2:", counter_v2)

v3 = [random.randint(-200, 145) for _ in range(100000)]
sorted_v3, counter_v3 = bubble_sort(v3)
print("v3_sorted:", sorted_v3[:10])
print("Comparaciones en v3:", counter_v3)

v2_sorted: [-1, 0, 4, 5, 6, 7]
Comparaciones en v2: 5

```

## Corrida de escritorio

$v_1 = [3, 2, 5, 8, 4, 1]$

| i | Vector                   |
|---|--------------------------|
| 0 | \$ [3, 2, 5, 8, 4, 1] \$ |
| 1 | \$ [3, 2, 5, 4, 1, 8] \$ |
| 2 | \$ [3, 2, 4, 1, 5, 8] \$ |
| 3 | \$ [3, 2, 1, 4, 5, 8] \$ |

...

Resultado final:

$v_{sorted} = [1, 2, 3, 4, 5, 8]$

Casos de prueba:

- $v_2 = [-1, 0, 4, 5, 6, 7]$
- $v_3$  100\_000 número aleatorios entre -200 y 145.

## Algoritmo 3

Pseudocode:

```
1 procedure iterative          (n: nonnegative integer)
2 if n=0 then
3     return 0
4 else
5     x := 0
6     y := 1
7     for i := 1 to n - 1
8         z := x + y
9         x := y
10        y := z
11 return y
```

## Algoritmo 3

XSDF

| n            | fib(n) |
|--------------|--------|
| 0            | 0      |
| 1            | 1      |
| 2            | 1      |
| 3            | 2      |
| 4            | 3      |
| 5            | 5      |
| 6            | 8      |
| 6            | 13     |
| ...          | ...    |
| \$n = 11\$   | ?      |
| \$n = 84\$   | ?      |
| \$n = 1531\$ | ?      |

## Graficar!

- El valor de la serie  $fib(n)$
- El valor del cociente

$$\phi \rightarrow \frac{fib(n)}{fib(n-1)} \approx 1.618 \text{ número áureo.}$$

| n        | $\text{fib}(n) / \text{fib}(n-1)$                        |
|----------|--|
| 2        | $1/1=1$  |
| 3        | $2/1=2$  |
| 4        | $3/2=1.5$  |
| 5        | $5/3=1.66667$  |
| 6        | $8/5=1.6$  |
| 7        | $13/8=1.625$   |
| 8        | $21/13=1.615$  |
| ...      | ...  |
| $\infty$ | $\frac{1 + \sqrt{5}}{2} \approx 1.618$<br>(número áureo) |

## Breve revisión *git*

- Inicializar repositorio

```
git init
```

- ¿Qué significa la U?
  - No están añadidos
  - Se añaden con ``bash git add file1 file2... import plotly.graph\_objects as go from plotly.subplots import make\_subplots import numpy as np

```
def fibonacci(n): if n < 0: raise ValueError("n debe ser un entero no negativo.") if n == 0: return 0
x, y = 0, 1 for _ in range(1, n): x, y = y, x + y return y
```

```
def graficar_analisis_fibonacci(n_max=20): n_valores = list(range(1, n_max + 1)) fib_valores =
[fibonacci(n) for n in n_valores] n_cociente = list(range(2, n_max + 1)) cociente_valores =
[fib_valores[i] / fib_valores[i-1] for i in range(1, len(fib_valores))] phi = (1 + np.sqrt(5)) / 2
```

```
fig = make_subplots(
    rows=2, cols=1,
    subplot_titles=(
        "Crecimiento de la Serie de Fibonacci",
        "Convergencia al Número Áureo ( $\phi$ )"
    )
)

fig.add_trace(go.Scatter(
    x=n_valores,
    y=fib_valores,
    mode='lines+markers',
    name='fib(n)',
    line=dict(dash='dash')
), row=1, col=1)
```

```

fig.add_trace(go.Scatter(
    x=n_cociente,
    y=cociente_valores,
    mode='lines+markers',
    name='Cociente',
    line=dict(dash='dot')
), row=2, col=1)

fig.add_hline(y=phi, line_dash="dash", line_color="red",
annotation_text=f"Número Áureo  $\phi \approx \{phi:.4f\}$ ", row=2, col=1)

fig.update_layout(
    height=800,
    title_text="Análisis Gráfico de la Serie de Fibonacci",
    template="plotly_dark",
    showlegend=False
)
fig.update_yaxes(title_text="Valor de fib(n)", row=1, col=1)
fig.update_yaxes(title_text="Cociente fib(n)/fib(n-1)", row=2, col=1)
fig.update_xaxes(title_text="n", row=1, col=1)
fig.update_xaxes(title_text="n", row=2, col=1)

fig.show()

```

```

if name == "main": graficar_analisis_fibonacci(n_max=20) * Nunca!  Añade **TODO**!
bash git add . * git commit bash git commit -m "actualización de código" ``

```