

# Python 3.10: Welcome to pattern matching 🔥

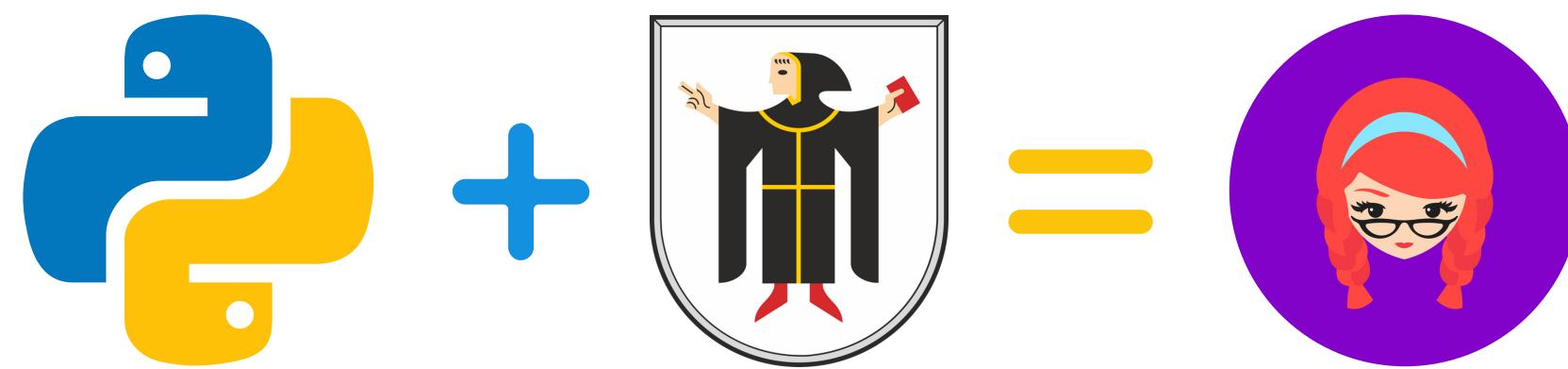


@laysauchoa

# self.intro()



@laysauchoa



@aiven\_io

# Structure

**Part 1:** Python 3.10

**Part 2:** Pattern Matching 

**Part 3:** Final thoughts

# Python 3.10 Party



Pablo Galindo  
@pyblogsal

...

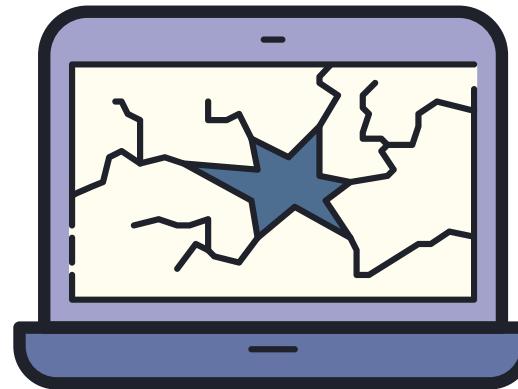
Join us at the Python 3.10 release party 🎉 that we are organising with the good people of [@PythonDiscord](#). We will have several core devs 💬 speaking about new Python features and secrets and you can see me making mistakes live! 🚀 [#python\\_release\\_party](#)

[youtube.com/watch?v=AHT2l3...](https://youtube.com/watch?v=AHT2l3...)



@laysauchoa

# Python 3.10 Breaking Changes



**Leah Neukirchen** @LeahNeukirchen · Sep 27

...

You won't believe how much software breaks because **Python 3.10** has a two-digit minor version.

51

194

1.3K



[Show this thread](#)

## Python 3.9.7

@laysauchoa

# Python 3.10

## Meaningful messages



Anthony Shaw   
@anthonypjshaw

...

Loving Python 3.10 so far.  
All these "little"  
improvements like  
suggesting which attribute  
you meant are going to  
make life much easier

```
try:  
    1/0  
except ZeroDivisionError, KeyError:  
    print("Something is wrong")  
  
File "<ipython>", line 3  
except ZeroDivisionError, KeyError:  
^  
SyntaxError: multiple exception types must be parenthesized
```

@laysauchoa



**laysa** ✨  
@laysauchoa

...

54.7% of the 360 most downloaded packages on PyPI  
supports Python 3.10 now!



: [pyreadiness.org/3.10/](https://pyreadiness.org/3.10/)

#Python

@laysauchoa

# Python 3.10

## mypy



The Mypy Project  
@mypyproject

...

Mypy 0.940 is out, with support for the Python 3.10  
match statement, and much more! [mypy-  
lang.blogspot.com/2022/03/mypy-0...](https://mypy-lang.blogspot.com/2022/03/mypy-0...)

4:40 PM · Mar 11, 2022 · Twitter Web App

@laysauchoa



**PEP 634 -- Structural Pattern Matching: Specification**  
**PEP 622 -- Structural Pattern Matching**  
**PEP 636 -- Structural Pattern Matching: Tutorial**

<https://github.com/python/cpython/blob/main/Doc/whatsnew/3.10.rst>

@laysauchoa

# Python 3.10

## What is new?



Pablo Galindo @pyblogsal · Jun 2

Which Python 3.10 feature are you most excited about?

...

Pattern matching

38.8%

Context managers with ()

9%

Better error messges

41.2%

Type union operator

11%

691 votes · Final results

15

25

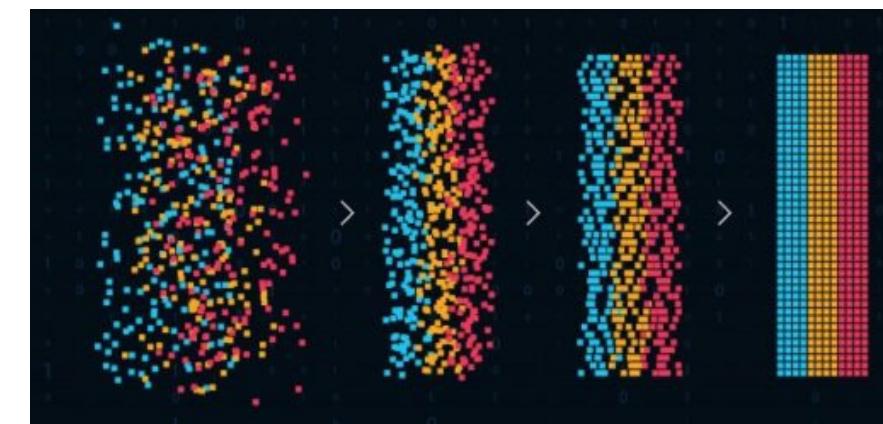
47



@laysauchoa

# Pattern matching

**"either it will or will not be a match"**

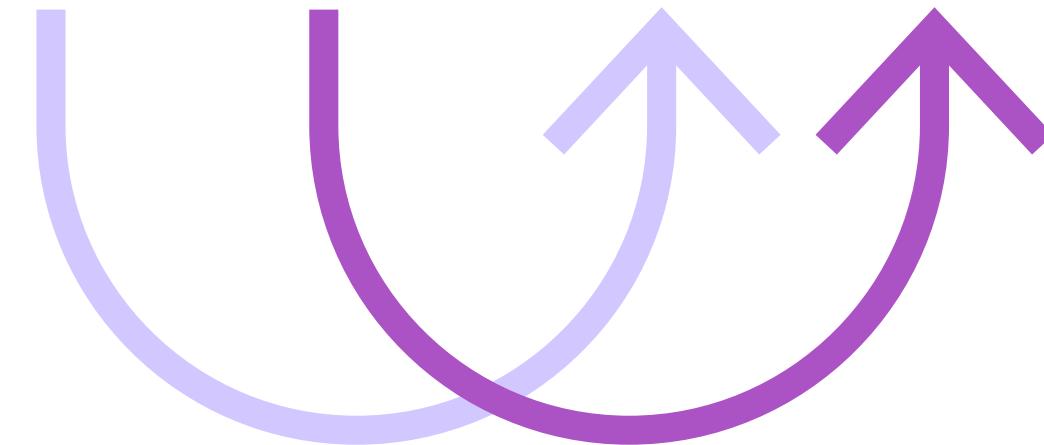


not a dating app  
not a pattern recognition

@laysauchoa

# Primitive Pattern matching

**x, y = 1, 2**



# Primitive Pattern matching



```
1  try:  
2      # do something that may fail  
3  except ILikeYouException:  
4      # do something about it  
5  except YouAreBeingExceptional:  
6      # do something about it
```

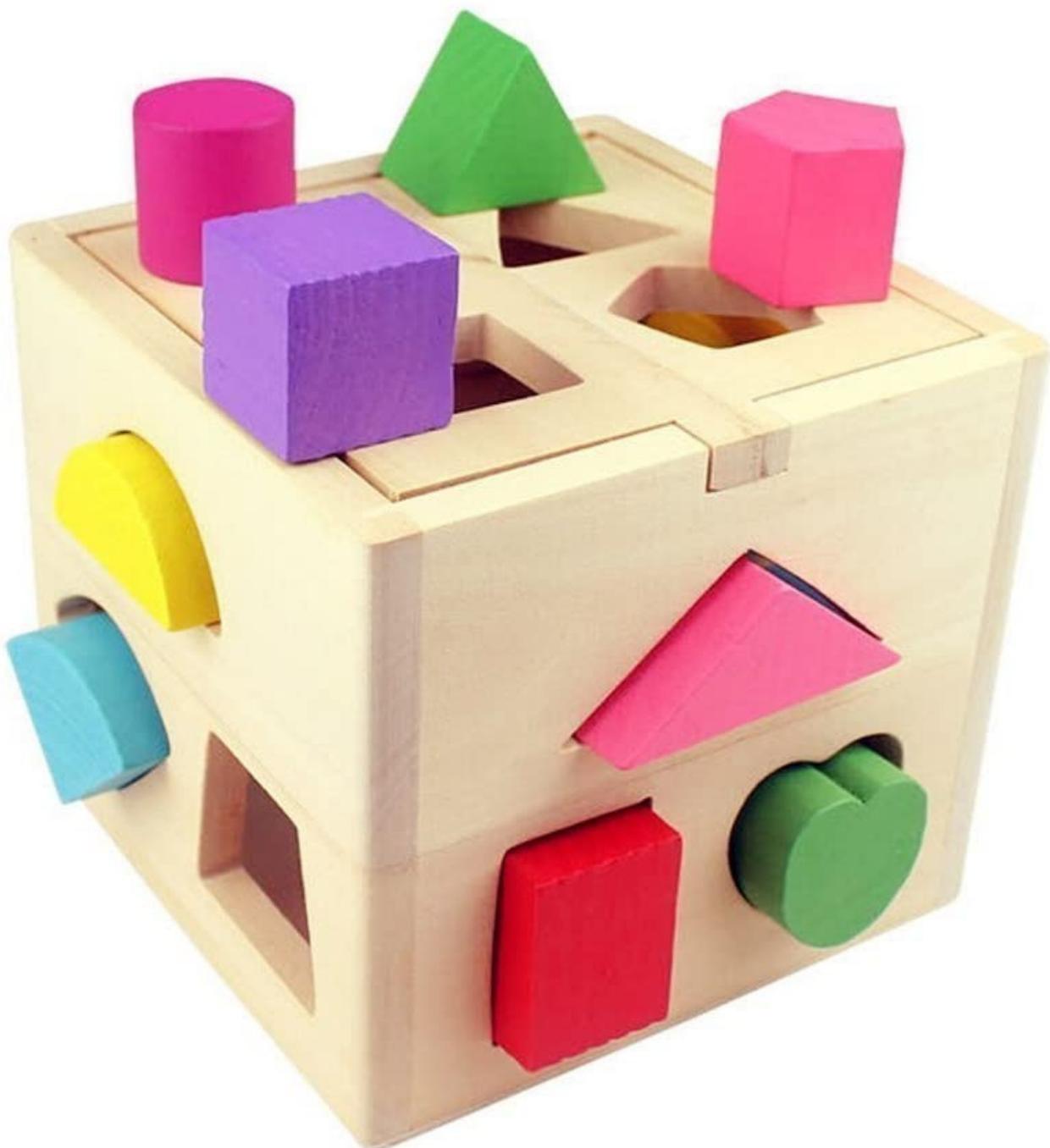
# Game Pattern matching



source: [@TodoDiaGatinhos](#)

@laysauchoa

# Steckspielzeug



@laysauchoa

# First Game

# Pattern matching

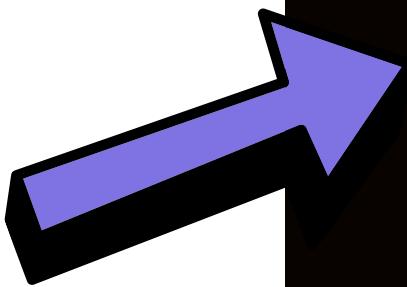
```
In [1]: form = "triangle"
.....
....: match form:
....:     case "circle":
....:         print("Circle: ●")
....:     case "square":
....:         print("Square: ■")
....:     case "triangle":
....:         print("Triangle: ▲")
....:     case _:
....:         print("Any other form!")
```



@laysauchoa

# Pattern matching

pattern



```
● ● ●  
form = "triangle"  
match form:  
    case "circle":  
        print("Circle: ")  
    case "square":  
        print("Square: ")  
    case "triangle":  
        print("Triangle: ")  
    case _:  
        print("Any other form")
```

# Pattern matching

pattern

```
● ● ●  
form = "triangle"  
match form:  
    case "circle":  
        print("Circle: ")  
    case "square":  
        print("Square: ")  
    case "triangle":  
        print("Triangle: ")  
    case _:  
        print("Any other form")
```

subject

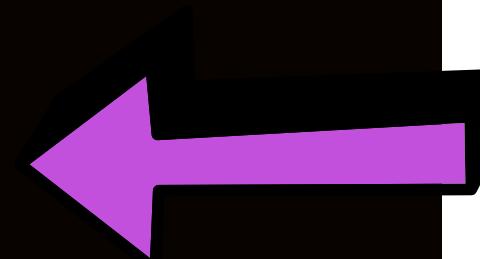
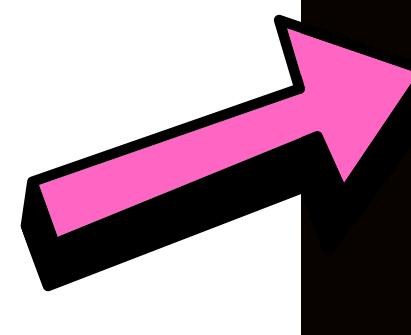
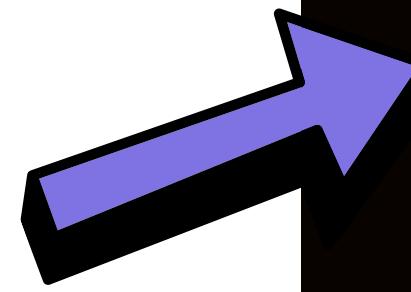
# Pattern matching

pattern

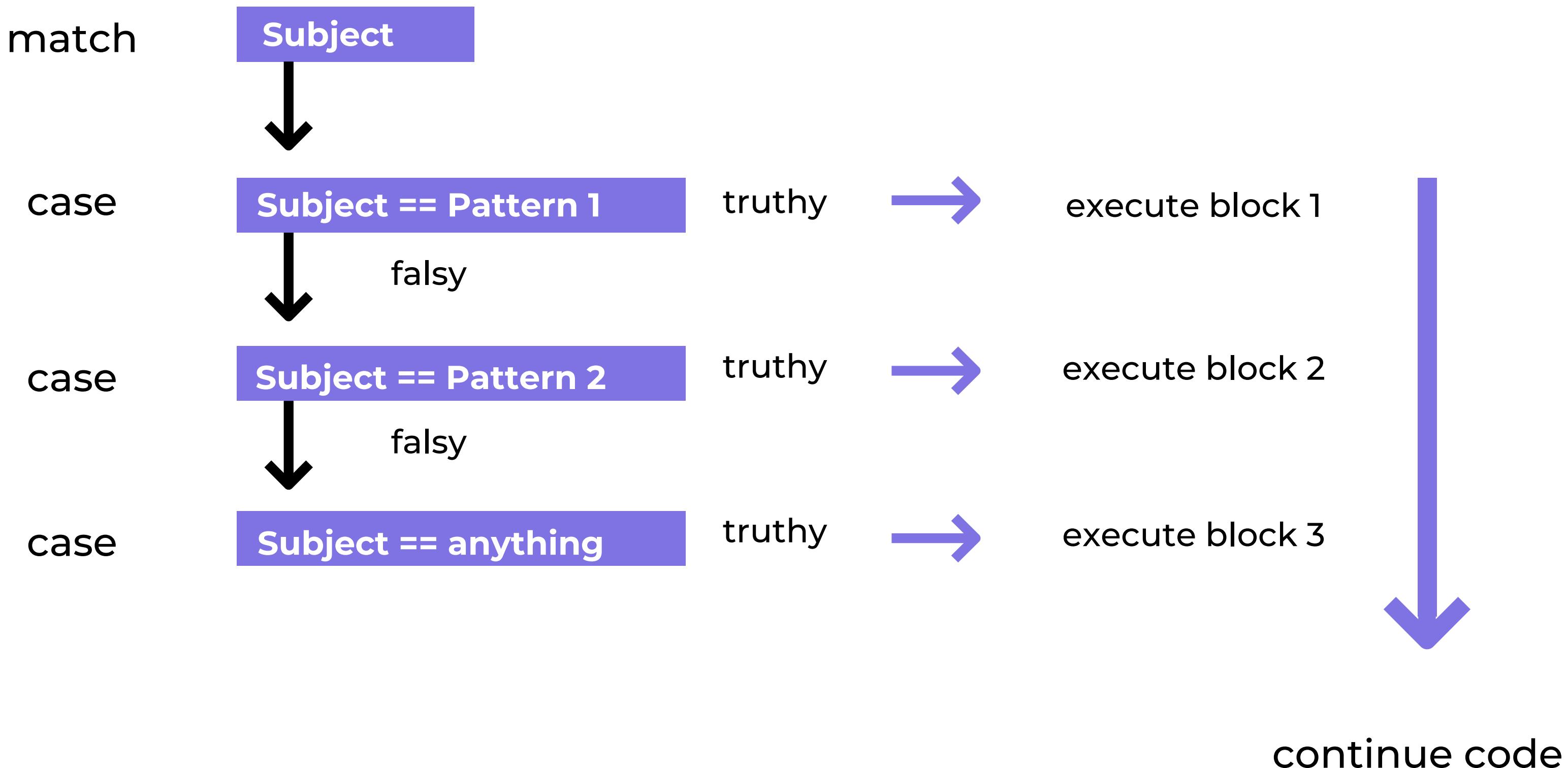
wildcard

subject

```
● ● ●  
form = "triangle"  
match form:  
    case "circle":  
        print("Circle: ")  
    case "square":  
        print("Square: ")  
    case "triangle":  
        print("Triangle: ")  
    case _:  
        print("Any other form")
```



# Pattern matching



# Don'ts

## Pattern matching

```
In [1]: form = "triangle"
....:
....: match form:
....:     case _:
....:         print("Any other form!")
....:     case "circle":
....:         print("Circle: ●")
....:     case "square":
....:         print("Square: ■")
....:     case "triangle":
....:         print("Triangle: ▲")
```



@laysauchoa

# With and Without Pattern matching

```
def store_elements(elements: List[str])-> str:  
    match elements:  
        case ["triangle"]:  
            print("triangle is stored")  
        case ["triangle", obj]:  
            print(f"triangle and {obj} are stored")  
        case _:  
            print("no matches found")
```



# With and Without Pattern matching

```
def store_elements(elements: List[str]) -> str:  
    match elements:  
        case ["triangle"]:  
            print("triangle is stored")  
        case ["triangle", obj]:  
            print(f"triangle and {obj} are stored")  
        case _:  
            print("no matches found")
```



```
def store_elements(elements: List[str]) -> str:  
    if elements == ["triangle"]:  
        print("triangle is stored")  
    elif len(elements) == 2 and elements[0] == "triangle":  
        obj = elements[1]  
        print(f"triangle and {obj} are stored")  
    else:  
        print("no matches found")
```



@laysauchoa

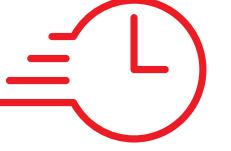
**%timeit** ⏱

# Pattern matching



```
In [46]: %timeit store_elements_with_match(["circle", "triangle", "square"])
367 ns ± 5.11 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)
```

```
In [47]: %timeit store_elements_without_match(["circle", "triangle", "square"])
380 ns ± 5.39 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)
```

**%timeit** 

## Pattern matching



```
In [48]: %timeit store_elements_without_match([])
239 ns ± 12 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)
```

```
In [49]: %timeit store_elements_with_match([])
177 ns ± 9.92 ns per loop (mean ± std. dev. of 7 runs, 10000000 loops each)
```

# With and Without Pattern matching



```
1 def count_elements(number: int) -> str:  
2     match number:  
3         case 0:  
4             print("No elements!")  
5         case 1:  
6             print("One element!")  
7         case 2:  
8             print("Two elements!")  
9         case _:  
10            print("wildcard!")
```



```
1 def count_elements(number: int) -> str:  
2     if number == 0:  
3         print("No elements!")  
4     elif number == 1:  
5         print("One element!")  
6     elif number == 2:  
7         print("Two elements!")  
8     else:  
9         print("wildcard!")
```



@laysauchoa

# Guard

## Pattern matching



```
from typing import List
def match_guard(form: List):
    exists = ["triangle", "circle", "heart", "square", "rectangle"]
    match form:
        case ["form", element] if element in exists:
            print(f"Element is of form {element}.")
        case _:
            print("No matches found.")
```



# Composable Patterns **with |**

## Pattern matching

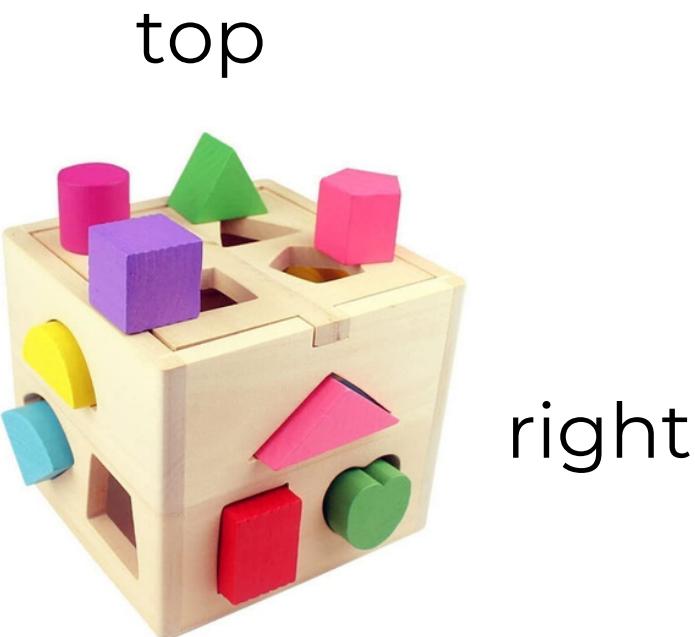
top



@laysauchoa

# Composable Patterns **with |**

## Pattern matching



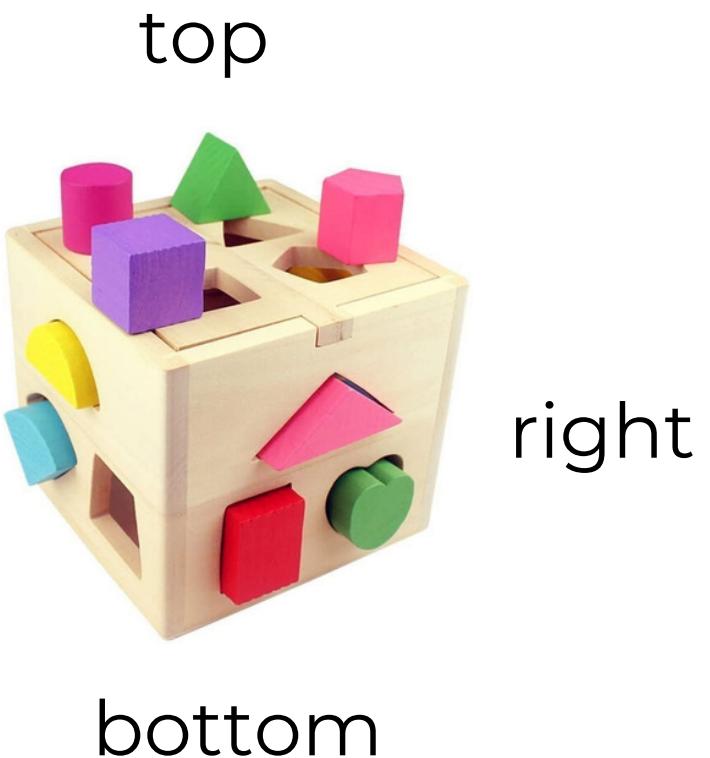
top

right

@laysauchoa

# Composable Patterns **with |**

## Pattern matching



top

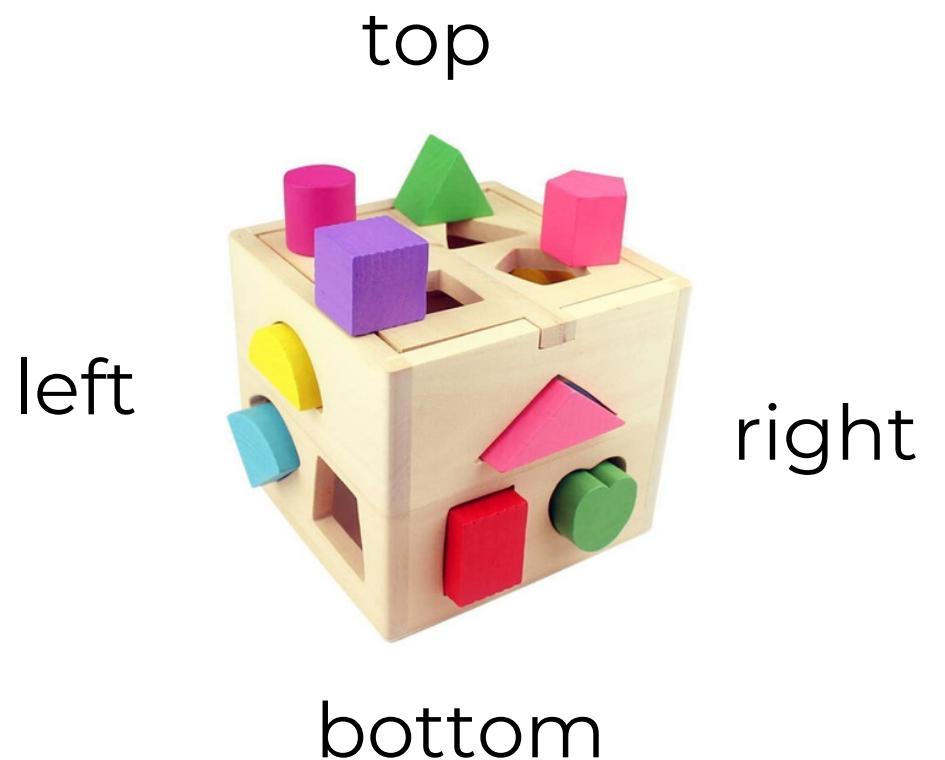
right

bottom

@laysauchoa

# Composable Patterns **with |**

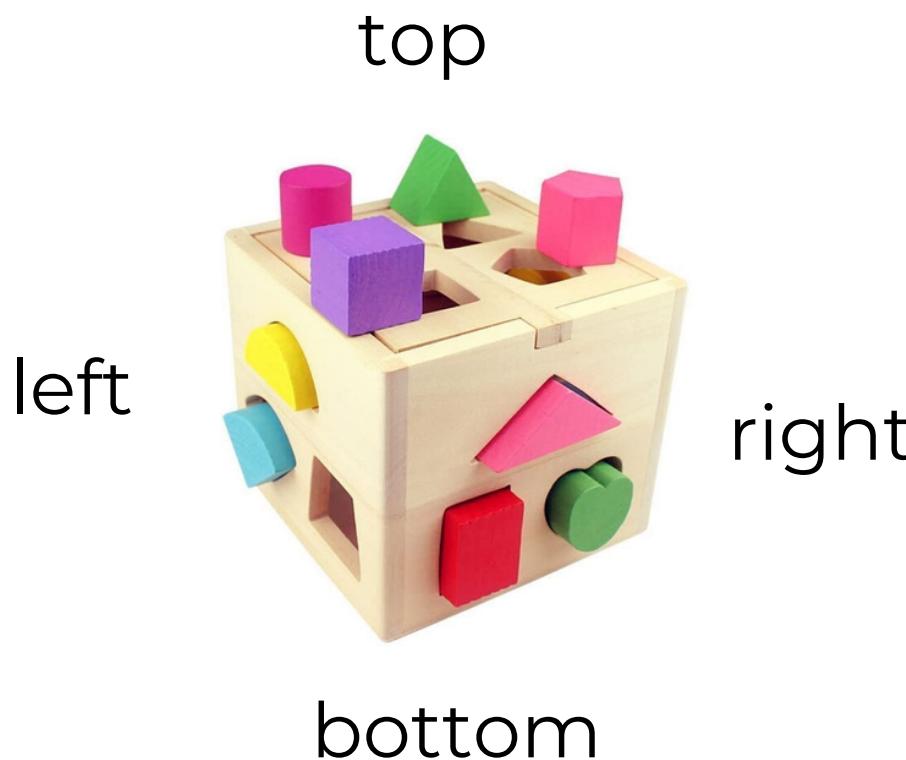
## Pattern matching



@laysauchoa

# Composable Patterns **with |**

## Pattern matching



```
● ● ●  
def store_elements(elements: List[str]) -> str:  
    match elements:  
        case ["direction", ("bottom" | "top" | "left" | "right") as side]:  
            print(f"The side is {side}")
```

# Matching Objects

## Pattern matching

```
In [1]:  
....: @dataclass  
....: class Circle:  
....:     color: str  
....:     radius: float  
....:  
....: element = Circle("pink", 10.0)  
....:  
....: match element:  
....:     case Circle(color="red"):  
....:         print(f"Red circle found.")  
....:     case Circle(color="pink"):  
....:         print(f"Pink circle found.")  
....:     case Circle(color=some_color, radius=value):  
....:         print(f"Circle found with color: {some_color} and radius: {value}")  
....:  
....:  
Pink circle found.
```



# Python 3.10

## Dict Pattern matching



```
○ ○ ○  
  
pikachu_data = {  
    "name": "Pikachu",  
    "type": "Electric",  
    "evolve": "Raichu"  
}  
  
match pikachu_data  
case {"type": "Psychic"}  
    print("🔍🧠")  
case {"type": "Water"}  
    print("🌊🌊🌊")  
case {"type": "Electric"}  
    print("⚡⚡⚡⚡")  
case _  
    print("Unknown type")
```

@laysauchoa

# Problem

Match all "matchme-



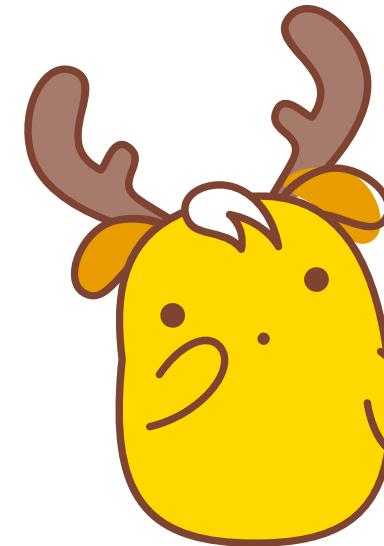
```
data = [{  
    "id": "matchme-foo",  
    "message": "hallo this is a message",  
}, {  
    "id": "matchme-bar",  
    "message": "goodbye",  
}, {  
    "id": "anotherid",  
    "message": "completely diffrent event"  
}, ...]
```

<https://stackoverflow.com/questions/70111401/python-3-10-pattern-matching-pep-634-wildcard-in-string>

@laysauchoa

# Problem

Match all "matchme-



```
for event in data:  
    match event:  
        case {'id': x} if x.startswith("matchme"):  
            print(event["message"])  
        case {'id':'anotherid'}:  
            print(event["message"])
```

<https://stackoverflow.com/questions/70111401/python-3-10-pattern-matching-pep-634-wildcard-in-string>

@laysauchoa

# Python 3.10

## Dict Pattern matching



laysa 🇧🇷 @laysauchoa · 2h

Python 3.10.0rc1: help 😕, why is this happening?

...

```
>>> event = {"size":3}
>>> match event:
...     case {}:
...         print("Empty dict!")
...
Empty dict!
```

```
>>> event = (1, 2, 3)
>>> match event:
...     case []:
...         print("Empty tuple!")
...
>>>
```

@laysauchoa

# Thoughts

## Pattern matching



@laysauchoa

# Questions

If you want to know anything more **Python 3.10** or share your ideas, **feel free to talk to me @laysauchoa or @aiven\_io**

---



**Thank you!**