

# Python **3.10**: Welcome to pattern matching



@laysauchoa

# self.intro()



**Laysa Uchoa**



@laysauchoa



[github.com/laysauchoa](https://github.com/laysauchoa)

- backend developer at e-mobility (Munich)
- dabbling in Python for 4+ years
- Python communities <3
- dystopian books, tea and carnival

@laysauchoa

# Structure

**Part 1:** Python 3.10

**Part 2:** Pattern Matching 

**Part 3:** Final thoughts

# 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

```
/Users/anthonyshaw/projects/cpython-3.10/python.exe -X dev Tests/run_libregr_tests.py test_buffer -v
Traceback (most recent call last):
  File "/Users/anthonyshaw/CLionProjects/pyjion/Tests/run_libregr_tests.py", line 8, in <module>
    pyjion.enable_graph()
AttributeError: module 'pyjion' has no attribute 'enable_graph'. Did you mean: 'enable_graphs'?
```

2:20 AM · Sep 30, 2021 · Twitter  
Web App

4 Retweets 62 Likes

@laysauchoa

# Python 3.10 OSS

⤳ You Retweeted



Hugo @hugovk · 23h

Python 3.10 is due out on Monday!

...

Right now only 9.7% of the top 360 packages on [@PyPI](#) explicitly support 3.10:

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

Maintainers! Is your library ready? Add `3.10-dev` to GitHub Actions or Travis CI to check tests pass and your dependencies are ready.

2

⤳ 30

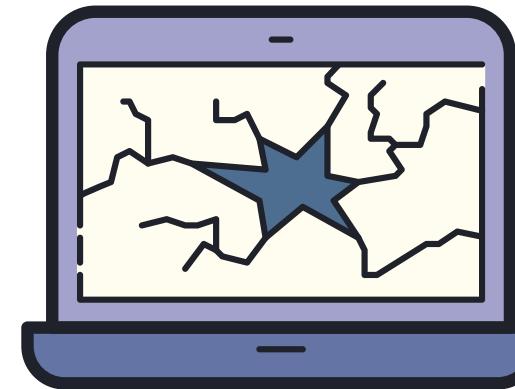
83



[Show this thread](#)

@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](#)

**Download the latest source release**

[Download Python 3.9.7](#)

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [macOS](#), [Other](#)

Want to help test development versions of Python? [Prereleases](#), [Docker images](#)

Looking for Python 2.7? See below for specific releases

@laysauchoa

# 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=AHT2I3...](https://youtube.com/watch?v=AHT2I3...)



@laysauchoa



<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 messages**

**41.2%**

Type union operator

11%

691 votes · Final results

15

25

47

↑

@laysauchoa

# Primitive Pattern matching

**x, y = 1, 2**



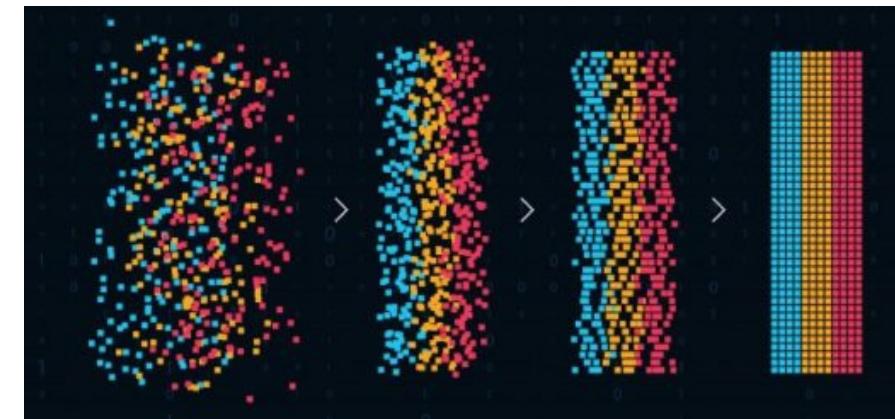
# Primitive Pattern matching



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

# Pattern matching

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



not a dating app  
not a pattern recognition

# Game Pattern matching



or

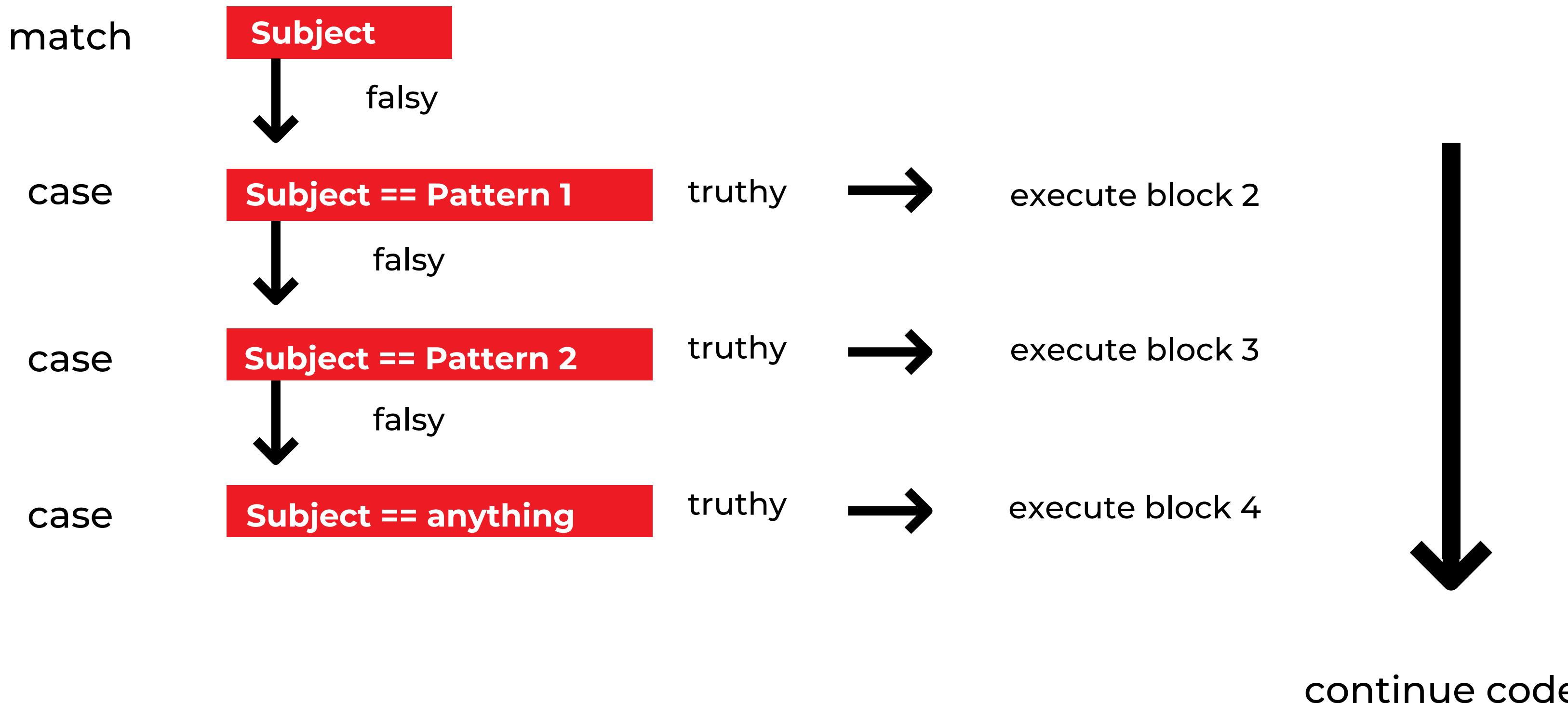
# First Game

# 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!")
```



# 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: ▲")
```



# With and Without Pattern matching

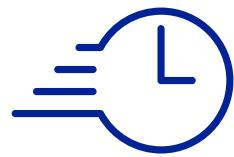


```
1 def store_elements(elements: List[str]) -> str:  
2     match elements:  
3         case ["triangle"]:  
4             print("triangle is stored")  
5         case ["triangle", obj]:  
6             print(f"triangle and {obj} are stored")  
7         case ["circle", *objects]:  
8             print(f"circle stored with: {objects}")  
9         case _:  
10            print("no matches found")
```



```
1 def store_elements(elements: List[str]) -> str:  
2     if elements == ["triangle"]:  
3         print("triangle is stored")  
4     elif len(elements) == 2 and elements[0] == "triangle":  
5         obj = elements[1]  
6         print(f"triangle and {obj} are stored")  
7     elif len(elements) >= 2 and elements[0] == "circle":  
8         obj = elements[1:]  
9         print(f"circle stored with: {obj}")  
10    else:  
11        print("no matches found")
```



**%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)
```

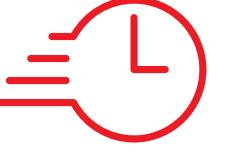


# Pattern matching



```
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)
```

%timeit 

# Pattern matching



```
# Python 3.10
In [48]: %timeit store_elements_without_match([])
239 ns ± 12 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)

# Python 3.7
In [3]: %timeit store_elements_without_match([])
272 ns ± 18.7 ns per loop (mean ± std. dev. of 7 runs, 1000000 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!")
```



# Guard

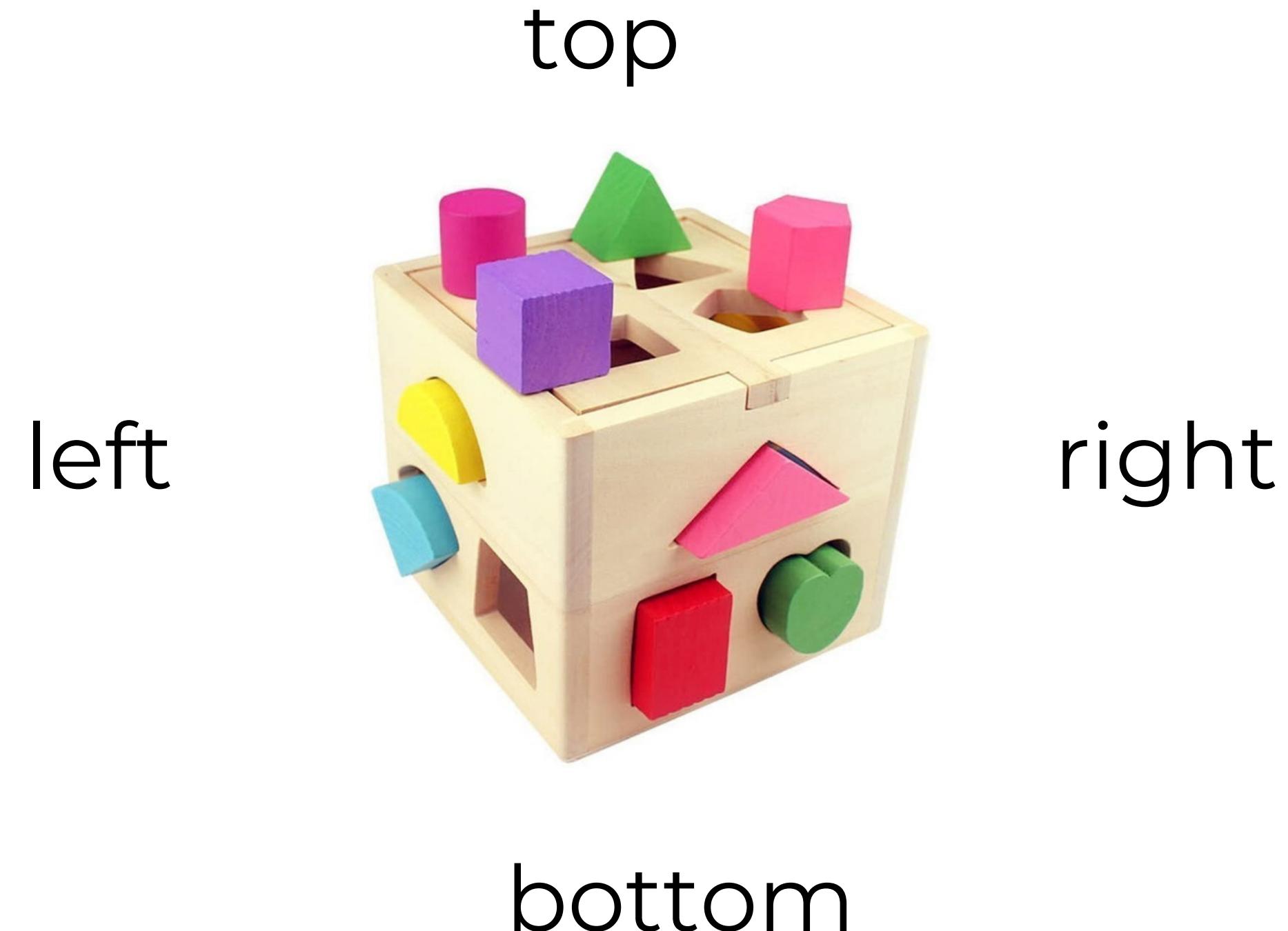
## Pattern matching



```
1 def store_elements(elements: List[str], exists: List[str]) -> str:
2     match elements:
3         case ["direction", side] if side in exists:
4             print(f"going {side}")
5         case ["direction", _]:
6             print(f"direction does not exist")
```

# Composable Patterns

## Pattern matching



# Composable Patterns

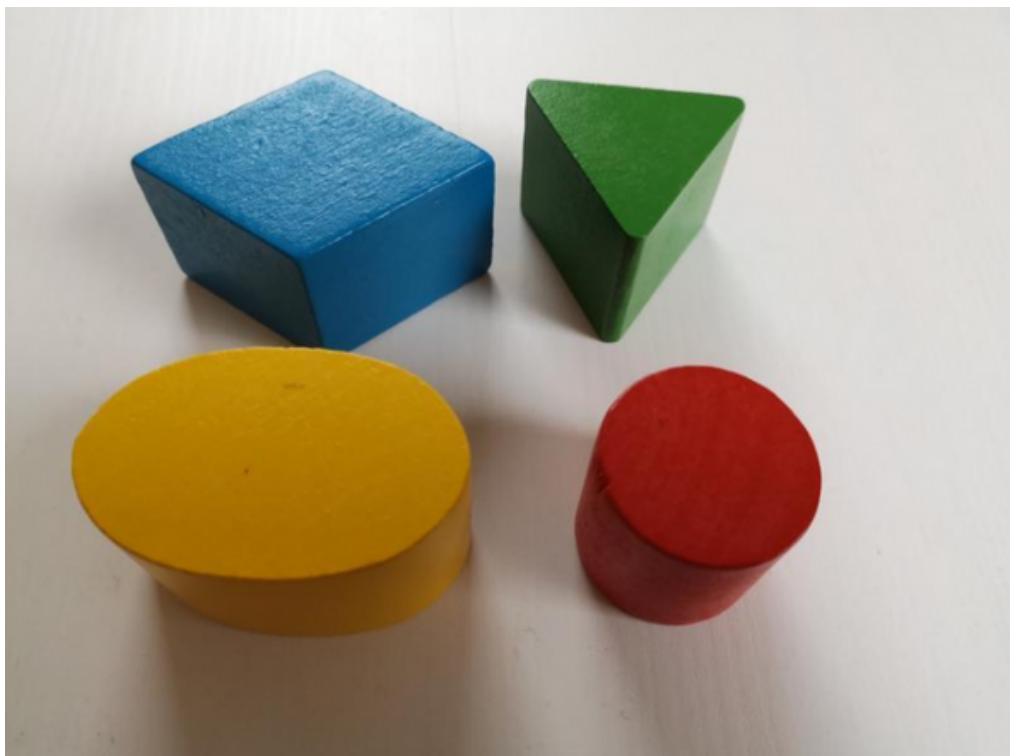
## Pattern matching



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

# Matching Objects

## Pattern matching



```
● ● ●  
1 @dataclass  
2 class Circle:  
3     color: str  
4     radius: float  
5  
6 element = Circle("pink", 10.0)  
7  
8 match element:  
9     case Circle(color="red", radius=value):  
10        print(f"Red circle stored with radius={value}")  
11     case Circle(color=element_color, radius=value) if value >= 10.0:  
12        print(f"Circle with radius={value}")  
13     case _:  
14        print("no match found.")
```

>>> Circle with radius=10.0

# **Dictionary**

## **Pattern matching**

# 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



# Questions

If you want to know anything more about my work in communities or share your ideas, **feel free to talk to me.**

---



**Thank you!**

---