```
nums = Series([7,8,9], index=[-1,0,1])
x = Series({"A":1, "B":2, "C":3})
y = Series({"A":2, "C":12, "D":4})
```

①

Expression	Result(s)
`nums[0]`	
`nums.at[0], nums.iat[0]`	
`nums.loc[-1], nums.iloc[-1]`	
`x / y`	

- -

```
s = Series(["A", "B", "C", "D"])
letters = Series(["x", "y", "z"], index=[1, 0, 3])
```

②

Expression	Result(s)
`s[-1]`	
`s[-2:]`	
`s + s`	
`letters[0]`	
`s + letters`	
`s[1:] + s[:-1]`	

- -

```
v = Series([-1, 1, 200, 191, 4])
```

③

Expression	Result(s)
`v < 0`	
`v * v == 1`	
`v[v > 100]`	
`v[v % 2 == 0]`	
`v[(v>0) & (v<100)]`	

note: `Series.loc[X]` looks for label X in the **index**. `Series.iloc[X]` looks for the **int position** X. These names are confusing. `iloc` supports negative indexing.

Code:	storms.csv:
| ```python
path = "storms.csv"
tab = pd.read_csv(path)

df = DataFrame({
 "code": ["o","p","a"],
 "where": ["other","Pacific","Atlantic"]
})
``` | name,year,type,speed,place<br>alice,2016,tornado,100,o<br>bob,2016,hurricane,200,p<br>cindy,2017,tornado,150,o<br>dan,2018,tornado,300,o<br>eve,2018,hurricane,250,a |

----

④

Expression	Result(s)
`df["code"]`	
`df.code`	
`type(df.code), type(df.where)`	
`tab.year.mean()`	
`tab.year == 2018`	
`tab.name[tab.year == 2018]`	
`df["where"] == "Atlantic"`	

`b = df["where"] == "other"`
`code = df.code[b].item()`
`nms = tab.name[tab.place==code]`
	# what are b, code, nms?

----

⑤

Expression	Result(s)
`tab.loc[0]`	
`tab.at[4, "type"]`	
`df.at[0,"where"] = "mainland"` `place = df["where"][0]`	# what is place?
`tab.loc[:, "speed"] += 1` `col = tab.speed`	# what is col?

note: s.COL is a shortcut for s["COL"], unless COL collides with a method name
also: when a Series s contains exactly one one item, s.item() extracts it