```
domains = []
for sale in sales_data:
    domains.append(sale.email.split("@")[1])
```

```
domains = []
for sale in sales_data:
    domains.append(sale.email.split("@")[1])
```

```
domains = []
```

```python
domains = []
for sale in sales_data:
    domains.append(sale.email.split("@")[1])
```

```python
domains = [sale.email.split("@")[1]]
domains = []
for sale in sales_data:
    domains.append(sale.email.split("@")[1])
```

```python
domains = [sale.email.split("@")[1] for sale in sales_data]
```

```python
ratings = []
for sale in sales_data:
    if sale.rating:
        ratings.append(int(sale.rating))
```

```
ratings = []
for sale in sales_data:
    if sale.rating:
        ratings.append(int(sale.rating))
```

```
ratings = []
ratings = []
for sale in sales_data:
    if sale.rating:
        ratings.append(int(sale.rating))
```

```
ratings = [int(sale.rating)]
```

```python
ratings = []
for sale in sales_data:
    if sale.rating:
        ratings.append(int(sale.rating))
```

```python
ratings = [int(sale.rating) for sale in sales_data]
ratings = []
for sale in sales_data:
    if sale.rating:
        ratings.append(int(sale.rating))
```

```
ratings = [int(sale.rating) for sale in sales_data if sale.rating]
```

Comprehending Comprehensions

by Rodrigo Girão Serrão

PyOhio 2022

@mathsppblog

List comprehensions

List comprehensions

```
[func(elem) for elem in iterable if cond(elem)]
```

List comprehensions

```
[
    func(elem)
    for elem in iterable
    if cond(elem)
]
```

List comprehensions

```
[
    func(elem)            # data transformation
    for elem in iterable
    if cond(elem)
]
```

List comprehensions

```
[
    func(elem)            # data transformation
    for elem in iterable  # data fetching
    if cond(elem)
]
```

List comprehensions

```
[
    func(elem)              # data transformation
    for elem in iterable    # data fetching
    if cond(elem)           # data filtering
]
```

List comprehensions

```
[
    func(elem)
    for elem in iterable
    if cond(elem)
]
```

```
... = []

for elem in iterable:
    if cond(elem):
        append(func(elem))
```

List comprehensions

```
... = []

for elem in iterable:
    if cond(elem):
      append(func(elem))
```

```
[func(elem) for elem in iterable if cond(elem)]
```

List comprehensions – how?

- Element transformation goes first
- "Top to bottom" becomes "left to right"
- Conditions filter data
- Arbitrary nesting
    - Don't get too carried away!

List comprehensions – why?

- Simplifies useful pattern ⚒
- Data transformation is highlighted 🔦
- Gotta go fast 🐌

~~List~~ Set comprehensions

~~List~~ Set comprehensions

- Replace [ ] with { }
- Other rules and caveats apply

~~List~~ ~~Set~~ Dict comprehensions

~~List~~ ~~Set~~ Dict comprehensions

- Separate keys and values with `:`
- Other rules and caveats apply

Generator expressions

Generator expressions

- Replace [ ] with ( )
- "Lazy list comprehensions"
- Other rules and caveats apply

Comprehending Comprehensions
[ ]
From the international Python speaker
RODRIGO GIRÃO SERRÃO

mathspp.com/comprehensions

*book mockup – digital book only 🌲