# PRO TIP: DICTIONARY COMPREHENSIONS



Comprehensions can also **create dictionaries** from other iterables

Syntax:

```
new_dict = {key: value for key, value in other_iterable(if condition)}
```

*These can be expressions (including function calls!)*

**EXAMPLE** *Creating a dictionary of inventory costs per item (stock quantity * price)*

```
items = ['skis', 'snowboard', 'goggles', 'boots']
status = [[5, 249.99], [0, 219.99], [0, 99.99], [12, 79.99]]

inventory_costs = {k: round(v[0] * v[1], 2) for k, v in zip(items, status)}

inventory_costs
```

```
{'skis': 1249.95, 'snowboard': 0.0, 'goggles': 0.0, 'boots': 959.88}
```

*This is creating a dictionary by using **items** as keys, and the product of **status[0]** and **status[1]** as values*

*zip() is being used to stitch the two lists together into a single iterable*

```
inventory_costs = {
    k: round(v[0] * v[1], 2) for k, v in zip(items, status) if v[0] > 0
}

inventory_costs
```

```
{'skis': 1249.95, 'boots': 959.88}
```

*You can still use conditional logic!*

# ASSIGNMENT: DICTIONARY COMPREHENSIONS

**NEW MESSAGE**

*February 11, 2022*

From: **Jerry Slush** *(IT Manager)*

Subject: **Final Tax Calculator**

Hey there,

We're getting close to having a final ETL process for our European dictionary. Can you create a function that applies our tax calculator to a list of subtotals and matches the output up with customer_IDs?

Store them in a dictionary, with customer IDs as keys, and the transactions as values.

Thanks – we're getting close to implementing this!

*final_tax_calculator.ipynb*

Reply | Forward

## *Results Preview*

```
from tax_calculator import tax_calculator

customer_ids = ['C00004', 'C00007', 'C00015', 'C00016', 'C00020', 'C00010']

subtotals = [15.98, 899.97, 799.97, 117.96, 5.99, 599.99]
```

```
transaction_dict_creator(customer_ids, subtotals, .08)
```

```
{'C00004': [15.98, 1.28, 17.26],
 'C00007': [899.97, 72.0, 971.97],
 'C00015': [799.97, 64.0, 863.97],
 'C00016': [117.96, 9.44, 127.4],
 'C00020': [5.99, 0.48, 6.47],
 'C00010': [599.99, 48.0, 647.99]}
```

# ASSIGNMENT: DICTIONARY COMPREHENSIONS

**NEW MESSAGE**

*February 11, 2022*

From: **Jerry Slush** *(IT Manager)*

Subject: **Final Tax Calculator**

Hey there,

We're getting close to having a final ETL process for our European dictionary. Can you create a function that applies our tax calculator to a list of subtotals and matches the output up with customer_IDs?

Store them in a dictionary, with customer IDs as keys, and the transactions as values.

Thanks – we're getting close to implementing this!

*final_tax_calculator.ipynb*

Reply | Forward

## *Solution Code*

```
from tax_calculator import tax_calculator

customer_ids = ['C00004', 'C00007', 'C00015', 'C00016', 'C00020', 'C00010']

subtotals = [15.98, 899.97, 799.97, 117.96, 5.99, 599.99]
```

```
def transaction_dict_creator(customer_ids, subtotals, tax_rate):
    # docstring omitted for screenshot
    customer_dict = {
        customer_id: tax_calculator(subtotal, tax_rate)
        for customer_id, subtotal in zip(customer_ids, subtotals)
    }
    return customer_dict
```

```
transaction_dict_creator(customer_ids, subtotals, .08)
```

```
{'C00004': [15.98, 1.28, 17.26],
 'C00007': [899.97, 72.0, 971.97],
 'C00015': [799.97, 64.0, 863.97],
 'C00016': [117.96, 9.44, 127.4],
 'C00020': [5.99, 0.48, 6.47],
 'C00010': [599.99, 48.0, 647.99]}
```