

List Comprehensions - Part A Reference

Basic Transforms & Filtering

The Pattern You're Replacing

```
# OLD WAY (4 lines)
result = []
for item in some_list:
    result.append(do_something(item))

# NEW WAY (1 line)
result = [do_something(item) for item in some_list]
```

Basic Comprehension Syntax

[expression	FOR	variable	IN	iterable]
↑		↑		↑
WHAT		EACH item		WHERE items
to create		is called		come from

Read it as: "Give me [expression] for each [variable] in [iterable]"

Examples: Basic Transforms

```
# Math operations
numbers = [1, 2, 3, 4, 5]
doubled = [n * 2 for n in numbers]      # [2, 4, 6, 8, 10]
squares = [n ** 2 for n in numbers]     # [1, 4, 9, 16, 25]

# String operations
names = ["alice", "bob", "charlie"]
upper = [name.upper() for name in names] # ["ALICE", "BOB", "CHARLIE"]
lengths = [len(name) for name in names]  # [5, 3, 7]

# Formatting
prices = [9.99, 14.99, 4.99]
display = [f"${p:.2f}" for p in prices]  # ["$9.99", "$14.99", "$4.99"]

# Time formatting
seconds = [180, 210, 195]
formatted = [f"{s//60}:{s%60:02d}" for s in seconds] # ["3:00", "3:30", "3:15"]
```

Filter Syntax

```
[ expression FOR variable IN iterable IF condition ]
                                ↑
                                only keep items
                                where this is True
```

Key point: Filter REMOVES items that don't pass the test!

Examples: Filtering

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Keep only even numbers
evens = [n for n in numbers if n % 2 == 0]
# [2, 4, 6, 8, 10]

# Keep only numbers greater than 5
big = [n for n in numbers if n > 5]
# [6, 7, 8, 9, 10]

# Filter AND transform
doubled_evens = [n * 2 for n in numbers if n % 2 == 0]
# [4, 8, 12, 16, 20]
```

```
# String filtering
usernames = ["", "gamer_pro", "x", "music_fan", ""]

# Keep non-empty (empty string is falsy)
valid = [name for name in usernames if name]
# ["gamer_pro", "x", "music_fan"]

# Keep long usernames
long = [name for name in usernames if len(name) >= 5]
# ["gamer_pro", "music_fan"]
```

Multiple Conditions

Use `and` / `or` to combine conditions:

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Keep numbers between 3 and 8
middle = [n for n in numbers if n >= 3 and n <= 8]
# [3, 4, 5, 6, 7, 8]

# Keep numbers less than 3 OR greater than 8
edges = [n for n in numbers if n < 3 or n > 8]
# [1, 2, 9, 10]
```

Common Mistakes

✗ Forgetting `for` :

```
# WRONG
result = [n * 2 in numbers]

# RIGHT
result = [n * 2 for n in numbers]
```

✗ Wrong order:

```
# WRONG
result = [for n in numbers n * 2]

# RIGHT
result = [n * 2 for n in numbers]
```

✗ Forgetting the list stays unchanged:

```
original = [1, 2, 3]
doubled = [n * 2 for n in original]
# original is still [1, 2, 3]
# doubled is [2, 4, 6]
```

Quick Reference Table

Goal	Syntax
Transform all items	<code>[expr for x in list]</code>
Filter items	<code>[x for x in list if condition]</code>
Filter then transform	<code>[expr for x in list if condition]</code>

Remember

1. **Comprehensions create NEW lists** - original is unchanged
2. **Filter (`if` at end) removes items** - result may be smaller
3. **Read right-to-left:** "for x in list" then "what to do with x"