## 1. `join()`
Syntax: `separator.join(list)`
```python
list3 = ['1', '2', '3', '4']
result = "-".join(list3)
print(result)    # Output: 1-2-3-4
```

## 2. Removing Empty Items from a List
Syntax: `[item for item in list if item]`
```python
my_list = ["Hello", "", "World", "", "Python"]
filtered_list = [item for item in my_list if item]
print(filtered_list)   # Output: ["Hello", "World",
"Python"]
```

## 3. `max()`
Syntax: `max(list)`
```python
lst = [1, 3, 4, 2]
print(max(lst))   # Output: 4
```

### 4.1. List Methods `append()`
Syntax: `list.append(item)`
```python
my_list = [1, 2, 3]
my_list.append(4)
print(my_list)    # Output: [1, 2, 3, 4]
```

### 4.2. `insert()`
Syntax: `list.insert(index, item)`
```python
my_list = [1, 2, 3]
my_list.insert(1, 4)
print(my_list)    # Output: [1, 4, 2, 3]
```

### 4.3. `remove()`
Syntax: `list.remove(item)`
```python
my_list = [1, 2, 3, 2, 4]
my_list.remove(2)
print(my_list)    # Output: [1, 3, 2, 4]
```

### 4.4. `index()`
Syntax: `list.index(element)`
```python
my_list = [1, 2, 3, 2, 4]
index = my_list.index(2)
print(index)      # Output: 1
```

### 4.5. `pop()`
Syntax: `list.pop(index)`
```python
my_list = [1, 2, 3, 4, 5]
element = my_list.pop(2)
print(element)    # Output: 3
print(my_list)    # Output: [1, 2, 4, 5]
```

### 4.6. `find()`
Syntax: `string.find(substring)`
```python
my_string = "Hello, World!"
index = my_string.find("World")
print(index)      # Output: 7
```

### 4.7. `count()`
Syntax: `list.count(element)`
```python
my_list = [1, 2, 3, 2, 4, 2]
count = my_list.count(2)
print(count)      # Output: 3
```

### 5.1. List Operations - Slicing
Syntax: `list[start:end]`
```python
my_list = [1, 2, 3, 4, 5]
my_slice = my_list[1:4]
print(my_slice)   # Output: [2, 3, 4]
my_slice = my_list[-2:]
print(my_slice)   # Output: [4, 5]
```

## 6. Finding All Occurrences of an Element in a List
```python
my_list = [1, 2, 3, 2, 4]
element = 2
indices = [i for i, x in enumerate(my_list) if x ==
element]
print(indices)    # Output: [1, 3]
```

## 7. Checking Neighbors of a Given Element in a List of Lists
Syntax: `[(y + dy, x + dx) for dy, dx in directions if conditions]`
```python
def find_neighbors(point, grid):
    y, x = point
    max_y, max_x = len(grid), len(grid[0])
    directions = [(-1, 0), (1, 0), (0, -1), (0, 1),
(-1, -1), (-1, 1), (1, -1), (1, 1)]

    neighbors = [(y + dy, x + dx) for dy, dx in
directions if 0 <= y + dy < max_y and 0 <= x + dx <
max_x]

    return neighbors
```

## 8. Comparing Dates
Syntax: `compare_date(date1, date2)`
```python
def compare_date(date1, date2):
    month1, year1 = date1
    month2, year2 = date2

    if year1 < year2:
        return -1
    elif year1 > year2:
        return 1
    else:  # years are equal
        if month1 < month2:
            return -1
        elif month1 > month2:
            return 1
        else:
            return 0
```

Example usage:
```python
date1 = [10, 1995]
date2 = [8, 1995]
result = compare_date(date1, date2)
print(result)     # Output: 1
```

### 9.1. Additional Operators - Floor Division (`//`)
Syntax: `dividend // divisor