1. Conditions

Ternary Condition

```
max_value = numbers[row][column] if numbers[row][column] > max_value else max_value
```

2. Lists

List_Operations1

```
# append: Adding 10 to end of list
l1.append(10)

# insert: Inserting 5 at index 0
l1.insert(0, 5)

# extend: Adding multiple elements  [15, 20, 25] at the end
l1.extend([15, 20, 25])

# max: find the max value in the list
max_val = max(l1)

# min: find the min value in the list
min_val = min(l1)

# sum: Sum all items of the list
summed_list = sum(l1)

# ----------------------------------------------------------------------
l2 = [10, 20, 30, 40, 50]

# slicing
l2_new = l2[1:3] # l2 from index 1 to 3 (not included!), or 1 to 2 included. (= [20, 30])

# remove: Removes the first occurrence of 30
l2.remove(30)
print("After remove(30):", l2)

# pop: Removes the element at index 1 (20)
popped_val = l2.pop(1)

# delete: Deletes the first element (10)
del l1[0]

# ----------------------------------------------------------------------
fruits = ['orange', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']

# count: Return the number of occurences of an item
fruits.count('apple') # Output: 2

# index: Find index of next item from index 2 to index 6
fruits.index('apple', 3, 6)  # Output: 5

# reverse: Reverse the order of the list
fruits.reverse()

# sort: Sort the list A-Z or 0-9
fruits.sort()

# sorted: Create a NEW sorted list
fruits_new = fruits.sorted()
```

Get item from list

```
1d_list | num = numbers[i]
```
```
2d_list | num = numbers[row][column]
```

List Comprehension

A short and powerful way to construct a new list.

```
List | new_list = [item for item in list1 if (var satisfies this condition)]
```

The following list comprehension will transpose rows and columns:

```
Input | # Input: matrix = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]
```
```
Transpose | transposed = [[row[i] for row in matrix] for i in range(4)]
```
```
Output | # Output: [[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]
```