Python Reference Guide

List Basics

```
# Creating a list
my_list = [1, 2, 3, 4, 5]

# Accessing elements (indexing starts at 0)
first_element = my_list[0]          # 1
last_element = my_list[-1]          # 5

# Slicing
subset = my_list[1:4]               # [2, 3, 4]

# Modifying elements
my_list[2] = 10                     # [1, 2, 10, 4, 5]

# Adding elements
my_list.append(6)                   # [1, 2, 10, 4, 5, 6]
my_list.insert(1, 7)                # [1, 7, 2, 10, 4, 5, 6]

# Removing elements
removed_element = my_list.pop()     # Removes and returns 6
my_list.remove(10)                  # Removes the first occurrence of 10

# List concatenation
new_list = my_list + [8, 9]         # Combines two lists
```

Dictionary Basics

```
# Creating a dictionary
flavors = {'chocolate': 'rich', 'vanilla': 'sweet', 'strawberry': 'fruity'}

# Accessing elements
chocolate_desc = flavors['chocolate']  # 'rich'
vanilla_desc = flavors.get('vanilla')  # 'sweet'
mint_desc = flavors.get('mint', 'refreshing')

# Modifying elements
flavors['vanilla'] = 'creamy'

# Adding new key-value pairs
flavors['mint'] = 'cool'

# Removing key-value pairs
del flavors['vanilla']              # Removes key 'vanilla' and its value

# Checking if a key exists
if 'chocolate' in flavors:
    print("We have chocolate ice cream!")
```

String Methods

Method	Description	Example
`str.lower()`	Convert string to lowercase	"Hello".lower() → "hello"
`str.upper()`	Convert string to uppercase	"Hello".upper() → "HELLO"
`str.strip()`	Remove leading and trailing whitespace	" Hello ".strip() → "Hello"
`str.split()`	Split string into a list	"Hello World".split() → ["Hello", "World"]
`str.join()`	Join list elements into a string	",".join(["a", "b", "c"]) → "a,b,c"
`str.replace()`	Replace substring	"Hello".replace("l", "w") → "Hewwo"

List Methods

Method	Description	Example
`list.append()`	Add an item to the end of the list	[1, 2].append(3) → [1, 2, 3]
`list.extend()`	Add all items from another list	[1, 2].extend([3, 4]) → [1, 2, 3, 4]
`list.insert()`	Insert an item at a given position	[1, 3].insert(1, 2) → [1, 2, 3]
`list.remove()`	Remove first occurrence of an item	[1, 2, 2, 3].remove(2) → [1, 2, 3]
`list.index()`	Return index of first occurrence of an item	[1, 2, 3].index(2) → 1
`list.count()`	Count occurrences of an item	[1, 2, 2, 3].count(2) → 2
`list.sort()`	Sort the list in-place	[3, 1, 2].sort() → [1, 2, 3]
`list.reverse()`	Reverse the list in-place	[1, 2, 3].reverse() → [3, 2, 1]

Dictionary Methods

Method	Description	Example
`dict.get()`	Get value for key, with optional default	{'a': 1}.get('b', 0) → 0
`dict.keys()`	Return a view of dictionary's keys	{'a': 1, 'b': 2}.keys() → ['a', 'b']
`dict.values()`	Return a view of dictionary's values	{'a': 1, 'b': 2}.values() → [1, 2]
`dict.items()`	Return a view of dictionary's (key, value) pairs	{'a': 1, 'b': 2}.items() → [('a', 1), ('b', 2)]

Built-in Functions

Function	Description	Example
`len()`	Return the length of an object	len([1, 2, 3]) → 3
`range()`	Generate a sequence of numbers	list(range(3)) → [0, 1, 2]
`sorted()`	Return a new sorted list	sorted([3, 1, 2]) → [1, 2, 3]
`enumerate()`	Return an enumerate object	list(enumerate(['a', 'b'])) → [(0, 'a'), (1, 'b')]

Looping Constructs

List Iteration

```
fruits = ['apple', 'banana', 'cherry']
for fruit in fruits:
    print(f"Fruit: {fruit}")
```

Enumerate

```
fruits = ['apple', 'banana', 'cherry']
for index, fruit in enumerate(fruits):
    print(f"Index: {index}, Fruit: {fruit}")

# Enumerate is helpful when you need both the index and the value in a loop.
# It's more efficient and readable than using a separate counter variable.
# Use cases: tracking position in a list, creating numbered lists, etc.
```

Range

```
# Printing numbers from 0 to 4
for i in range(5):
    print(f"Number: {i}")

# Printing even numbers from 2 to 8
for i in range(2, 9, 2):
    print(f"Even number: {i}")
```

Dictionary Iteration

```
# Iterating over key-value pairs
my_dict = {'a': 1, 'b': 2, 'c': 3}
for key, value in my_dict.items():
    print(f"Key: {key}, Value: {value}")
```