

## TOP 30 Python Important Functions:

for interview preparation as well as for company use.

### 1. `len()`

**Explanation:** Returns the length (number of items) of an object such as a list, string, or dictionary.

**Example:**

```
# List Example
fruits = ['apple', 'banana', 'cherry']
length = len(fruits)
print(length) # Output: 3
```

```
# String Example
text = "Hello, World!"
length = len(text)
print(length) # Output: 13
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

### 2. `print()`

**Explanation:** Prints the specified message to the console.

**Example:**

```
message = "Hello, World!"
print(message) # Output: Hello, World!
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

### 3. `type()`

**Explanation:** Returns the type of the specified object.

**Example:**

```
number = 10
print(type(number)) # Output: <class 'int'>
```

```
text = "Hello"  
print(type(text)) # Output: <class 'str'>
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

#### 4. `input()`

**Explanation:** Allows user input from the console.

**Example:**

```
name = input("Enter your name: ")  
print(f"Hello, {name}!") # Output: Hello, [user input]!
```

**Rating: Interview Prep: 8/10 | Company Use: 8/10**

---

#### 5. `str()`

**Explanation:** Converts an object to a string.

**Example:**

```
number = 123  
string_number = str(number)  
print(string_number) # Output: '123'
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

#### 6. `int()`

**Explanation:** Converts a string or float to an integer.

**Example:**

```
text = "123"  
integer_number = int(text)  
print(integer_number) # Output: 123
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 7. float()

**Explanation:** Converts a string or integer to a float.

**Example:**

```
text = "123.45"  
float_number = float(text)  
print(float_number) # Output: 123.45
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 8. list()

**Explanation:** Converts an iterable (e.g., tuple, string) into a list.

**Example:**

```
# Converting a tuple to a list  
tuple_data = (1, 2, 3)  
list_data = list(tuple_data)  
print(list_data) # Output: [1, 2, 3]
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 9. dict()

**Explanation:** Creates a dictionary object.

**Example:**

```
# Creating a dictionary from key-value pairs  
data = dict(name="John", age=30, city="New York")  
print(data) # Output: {'name': 'John', 'age': 30, 'city': 'New York'}
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 10. range()

**Explanation:** Generates a sequence of numbers.

**Example:**

```
# Generating a list of numbers from 0 to 4
numbers = list(range(5))
print(numbers) # Output: [0, 1, 2, 3, 4]
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

## 11. `sum()`

**Explanation:** Sums the items of an iterable (e.g., list).

**Example:**

```
numbers = [1, 2, 3, 4, 5]
total = sum(numbers)
print(total) # Output: 15
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

## 12. `min()`

**Explanation:** Returns the smallest item in an iterable or the smallest of two or more arguments.

**Example:**

```
numbers = [10, 20, 5, 40]
minimum = min(numbers)
print(minimum) # Output: 5
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 13. `max()`

**Explanation:** Returns the largest item in an iterable or the largest of two or more arguments.

**Example:**

```
numbers = [10, 20, 5, 40]
maximum = max(numbers)
print(maximum) # Output: 40
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 14. `sorted()`

**Explanation:** Returns a sorted list of the specified iterable.

**Example:**

```
numbers = [3, 1, 4, 2]
sorted_numbers = sorted(numbers)
print(sorted_numbers)  # Output: [1, 2, 3, 4]
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 15. `append()`

**Explanation:** Adds an item to the end of the list.

**Example:**

```
fruits = ['apple', 'banana']
fruits.append('cherry')
print(fruits)  # Output: ['apple', 'banana', 'cherry']
```

**Rating: Interview Prep: 9/10 | Company Use: 10/10**

---

## 16. `extend()`

**Explanation:** Extends a list by appending elements from another iterable.

**Example:**

```
fruits = ['apple', 'banana']
more_fruits = ['cherry', 'date']
fruits.extend(more_fruits)
print(fruits)  # Output: ['apple', 'banana', 'cherry', 'date']
```

**Rating: Interview Prep: 8/10 | Company Use: 9/10**

---

## 17. `insert()`

**Explanation:** Inserts an item at a specified position in the list.

**Example:**

```
fruits = ['apple', 'banana']  
fruits.insert(1, 'cherry')  
print(fruits) # Output: ['apple', 'cherry', 'banana']
```

**Rating: Interview Prep: 8/10 | Company Use: 9/10**

---

## 18. `remove()`

**Explanation:** Removes the first occurrence of an element in a list.

**Example:**

```
fruits = ['apple', 'banana', 'cherry']  
fruits.remove('banana')  
print(fruits) # Output: ['apple', 'cherry']
```

**Rating: Interview Prep: 9/10 | Company Use: 9/10**

---

## 19. `pop()`

**Explanation:** Removes and returns the element at the specified position. If no position is specified, it removes and returns the last item.

**Example:**

```
fruits = ['apple', 'banana', 'cherry']  
fruit = fruits.pop(1)  
print(fruit) # Output: 'banana'  
print(fruits) # Output: ['apple', 'cherry']
```

**Rating: Interview Prep: 9/10 | Company Use: 9/10**

---

## 20. `clear()`

**Explanation:** Removes all items from the list.

**Example:**

```
fruits = ['apple', 'banana', 'cherry']
```

```
fruits.clear()
print(fruits) # Output: []
```

**Rating: Interview Prep: 8/10 | Company Use: 9/10**

---

## 21. `index()`

**Explanation:** Returns the index of the first occurrence of a specified value.

**Example:**

```
fruits = ['apple', 'banana', 'cherry']
position = fruits.index('banana')
print(position) # Output: 1
```

**Rating: Interview Prep: 8/10 | Company Use: 9/10**

---

## 22. `count()`

**Explanation:** Returns the number of occurrences of a specified value.

**Example:**

```
numbers = [1, 2, 2, 3, 2]
count_of_twos = numbers.count(2)
print(count_of_twos) # Output: 3
```

**Rating: Interview Prep: 8/10 | Company Use: 9/10**

---

## 23. `join()`

**Explanation:** Joins all items in a tuple or list into

40

**You said:**

Continue

**ChatGPT said:**

ChatGPT

a single string, with a specified separator.

**Example:**

```
words = ['Hello', 'World']
sentence = ' '.join(words)
print(sentence) # Output: 'Hello World'
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

## 24. `split()`

**Explanation:** Splits a string into a list of substrings based on a specified separator.

**Example:**

```
sentence = "Hello World"
words = sentence.split(' ')
print(words) # Output: ['Hello', 'World']
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

## 25. `map()`

**Explanation:** Applies a specified function to each item of an iterable (e.g., list) and returns a map object (which can be converted to a list).

**Example:**

```
# Function to square a number
def square(x):
    return x * x

numbers = [1, 2, 3, 4]
squared_numbers = list(map(square, numbers))
print(squared_numbers) # Output: [1, 4, 9, 16]
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

## 26. `filter()`

**Explanation:** Filters the elements of an iterable, returning only those that match a specified condition.

**Example:**

```
# Function to check if a number is even
def is_even(x):
```



```
return x % 2 == 0
```

```
numbers = [1, 2, 3, 4, 5, 6]
even_numbers = list(filter(is_even, numbers))
print(even_numbers) # Output: [2, 4, 6]
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

## 27. `reduce()`

**Explanation:** Applies a rolling computation to sequential pairs of values in a list. It's part of the `functools` module.

**Example:**

```
from functools import reduce

# Function to multiply two numbers
def multiply(x, y):
    return x * y

numbers = [1, 2, 3, 4]
product = reduce(multiply, numbers)
print(product) # Output: 24
```

**Rating: Interview Prep: 8/10 | Company Use: 8/10**

---

## 28. `lambda`

**Explanation:** A small anonymous function that can have any number of arguments but only one expression.

**Example:**

```
# Lambda function to double a number
double = lambda x: x * 2
result = double(5)
print(result) # Output: 10
```

**Rating: Interview Prep: 10/10 | Company Use: 10/10**

---

## 29. `enumerate()`

**Explanation:** Adds a counter to an iterable and returns it as an enumerate object.

**Example:**

```
fruits = ['apple', 'banana', 'cherry']
for index, fruit in enumerate(fruits):
    print(index, fruit)
# Output:
# 0 apple
# 1 banana
# 2 cherry
```

**Rating: Interview Prep: 9/10 | Company Use: 9/10**

---

### 30. `zip()`

**Explanation:** Combines two or more iterables (e.g., lists) element-wise, returning an iterator of tuples.

**Example:**

```
names = ['John', 'Jane', 'Doe']
ages = [25, 22, 35]
combined = list(zip(names, ages))
print(combined) # Output: [('John', 25), ('Jane', 22), ('Doe', 35)]
```

**Rating: Interview Prep: 9/10 | Company Use: 9/10**

---