File Handling in Python: A Comprehensive Guide

1. Basics of File Handling

Python provides the open() function for file operations. Modes include:

- 'r': Read (default)

- 'w': Write (creates file if not exists or truncates)

- 'a': Append (add content to the end)

- 'b': Binary mode (e.g., images, executables)

- 'x': Create (fails if file exists)

Example:

```
with open('example.txt', 'r') as file:
    for line in file:
        print(line.strip())
```

2. Reading Files

Functions available:

- read(size): Reads entire file or specified number of bytes

- readline(): Reads one line at a time

- readlines(): Reads all lines as a list

Examples:

```
# Read entire file
with open('example.txt', 'r') as file:
    content = file.read()
```

```
    print(content)


# Read one line
with open('example.txt', 'r') as file:
    line = file.readline()
    print(line.strip())


# Read all lines into a list
with open('example.txt', 'r') as file:
    lines = file.readlines()
    print(lines)
```

3. Writing to Files

Functions available:

- write(string): Writes a string to the file
- writelines(list): Writes a list of strings

Examples:

```
# Write a single string
with open('output.txt', 'w') as file:
    file.write('Hello, World!\n')


# Write multiple lines
with open('output.txt', 'w') as file:
```

```
    file.writelines(['First Line\n', 'Second Line\n'])
```

4. Moving the File Pointer

Functions available:

- tell(): Returns the current file pointer position

- seek(offset, whence): Moves the pointer (0=start, 1=current, 2=end)

Examples:

```
# Using tell and seek
with open('example.txt', 'r') as file:
    print(file.tell())  # Initial position
    file.seek(5)        # Move pointer to 5th byte
    print(file.read())
```

5. Working with Binary Files

Binary mode allows handling files like images or executables.

Examples:

```
# Read binary file
with open('image.jpg', 'rb') as file:
    content = file.read()

# Write binary data
with open('output.bin', 'wb') as file:
```

```
    file.write(b'Binary Data')
```

6. File Operations (os and shutil modules)

Perform operations like renaming, deleting, or copying files.

Examples:

```
# Rename a file
import os
os.rename('old_name.txt', 'new_name.txt')

# Delete a file
os.remove('file.txt')

# Move a file
import shutil
shutil.move('source.txt', 'destination/')
```

7. Exception Handling in File Operations

Always handle exceptions like FileNotFoundError.

Example:

```
try:
    with open('nonexistent.txt', 'r') as file:
        print(file.read())
except FileNotFoundError:
```

```
    print('File not found. Please check the path!')
```

8. Working with JSON Files

JSON is useful for structured data storage.

Examples:

```
# Write JSON data
import json
data = {'name': 'Alice', 'age': 25}
with open('data.json', 'w') as file:
    json.dump(data, file)


# Read JSON data
with open('data.json', 'r') as file:
    loaded_data = json.load(file)
    print(loaded_data)
```

9. Working with CSV Files

CSV is great for tabular data.

Examples:

```
# Write to CSV
import csv
with open('data.csv', 'w', newline='') as file:
    writer = csv.writer(file)
```

```
    writer.writerow(['Name', 'Age'])
    writer.writerow(['Alice', 25])

# Read CSV
with open('data.csv', 'r') as file:
    reader = csv.reader(file)
    for row in reader:
        print(row)
```