Data Handling Class 11 Python Notes

Introduction to Data Handling

Data handling in Python refers to the process of storing, accessing, and manipulating data efficiently using Python's built-in data structures. This is an essential skill for Class 11 students, as it provides a foundation for future programming.

Python Data Types

Python offers several types of data, categorized into mutable and immutable types.

1. **Mutable Data Types**: Mutable data types can be changed after creation. These include:

- **Lists**: Ordered collections that can be modified.

  Example:

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

- **Dictionaries**: Key-value pairs that are mutable.

  Example:

  ```python
  student = {'name': 'John', 'age': 17}
  student['age'] = 18
  print(student)
  # Output: {'name': 'John', 'age': 18}
  ```

2. **Immutable Data Types**: Immutable data types cannot be changed after creation. These include:

- **Tuples**: Ordered collections that cannot be changed.

  Example:

  ```python
  coordinates = (10, 20)
  print(coordinates)
  # Output: (10, 20)
  ```

- **Strings**: Immutable sequences of characters.

  Example:

  ```python
  name = 'John'
  new_name = name.replace('o', 'a')
  print(new_name)
  # Output: 'Jahn'
  ```

Data Structures in Python

Python offers various data structures, and each has its unique characteristics.

1. **Lists**: An ordered collection of items that can be changed.

  Example:

  ```python
  numbers = [1, 2, 3, 4]
  numbers.append(5)
  print(numbers)
  # Output: [1, 2, 3, 4, 5]
  ```

2. **Tuples**: Similar to lists but immutable.

Example:

```python

days = ('Monday', 'Tuesday', 'Wednesday')

print(days)

# Output: ('Monday', 'Tuesday', 'Wednesday')

```

3. **Dictionaries**: Store data as key-value pairs.

Example:

```python

person = {'name': 'Alice', 'age': 25}

print(person['name'])

# Output: Alice

```

4. **Sets**: Unordered collections that do not allow duplicates.

Example:

```python

unique_numbers = {1, 2, 3, 3, 2}

print(unique_numbers)

# Output: {1, 2, 3}

```

File Handling in Python

File handling allows you to store data in external files. Python uses the built-in `open()` function to work with files.

1. **Opening and Closing Files**

  Example:

```python
file = open('data.txt', 'w')
file.write('This is a test.')
file.close()
```

2. **Reading from Files**

  Example:

```python
file = open('data.txt', 'r')
content = file.read()
print(content)
file.close()
# Output: This is a test.
```

Exception Handling in Python

Exception handling is crucial when working with data to avoid crashes. Python provides a `try-except` block for managing exceptions.

Example:

```python
```

```
try:

    file = open('non_existent.txt', 'r')

except FileNotFoundError:

    print('File not found.')
```