Simple Character Matches

It seems like you might be referring to simple character matches in the context of string manipulation or pattern matching in Python. Let me provide you with some information and examples.

In Python, you can perform simple character matches using various methods and functions. Here are a few common ways to achieve this:

1. Using the `in` Operator:

The `in` operator is used to check if a particular character or substring is present in a given string.

```python
text = "Hello, World!"

# Check if a specific character is present
print('H' in text)  # Output: True

# Check if a substring is present
print('Hello' in text)  # Output: True
print('Python' in text)  # Output: False
```

2. Using the `find` Method:

The `find` method returns the lowest index of the substring if it is found, and -1 otherwise.

```python
text = "Hello, World!"

# Find the index of a substring
index = text.find('World')
print(index)  # Output: 7

# If not found, find returns -1
index = text.find('Python')
print(index)  # Output: -1
```

3. Using Regular Expressions:

Python's `re` module provides powerful tools for pattern matching using regular expressions.

```python
import re

text = "Hello, World!"
```

```
# Search for a specific pattern
pattern = re.compile(r'\bWorld\b')
match = pattern.search(text)

if match:
    print("Pattern found:", match.group())
else:
    print("Pattern not found")
```

In this example, `\b` is a word boundary, ensuring that "World" is a whole word and not part of another word.

4. Using the `startswith` and `endswith` Methods:

These methods check if a string starts or ends with a specified substring.

```python
text = "Hello, World!"

# Check if a string starts with a specific substring
print(text.startswith('Hello'))  # Output: True

# Check if a string ends with a specific substring
print(text.endswith('World!'))  # Output: True
```