Regular Expressions (Regex) in Python

Regular expressions are powerful tools for searching, matching, and manipulating strings based on patterns.

1. **Introduction to Regular Expressions**

- The `re` module in Python provides functions for working with regular expressions.
- You can use regular expressions to:
  - Search for specific patterns within a string.
  - Replace parts of a string that match a pattern.
  - Split strings based on patterns.

2. **Basic Functions in the `re` module**

- `re.match()`: Checks if the regular expression matches the beginning of the string.

```
import re
result = re.match(r'Hello', 'Hello, world!')
print(result)  # Output: <re.Match object; span=(0, 5), match='Hello'>
```

- `re.search()`: Searches for the first location where the regular expression matches.

```
result = re.search(r'world', 'Hello, world!')
print(result)  # Output: <re.Match object; span=(7, 12), match='world'>
```

- `re.findall()`: Returns a list of all non-overlapping matches of the regular expression in the string.

```
result = re.findall(r'\d+', 'There are 12 apples and 34 oranges')
print(result)  # Output: ['12', '34']
```

- `re.sub()`: Replaces occurrences of the pattern with a specified string.

```
result = re.sub(r'apples', 'bananas', 'There are 12 apples and 34 oranges')
print(result)  # Output: There are 12 bananas and 34 oranges
```

3. **Special Characters in Regular Expressions**

- `.`: Matches any character except a newline.
- `^`: Matches the start of the string.
- `$`: Matches the end of the string.
- `[]`: Matches any single character inside the brackets.

- `|`: Acts as a logical OR operator.
- `\d`: Matches any digit (equivalent to `[0-9]`).
- `\w`: Matches any alphanumeric character (equivalent to `[a-zA-Z0-9_]`).
- `+`: Matches 1 or more occurrences of the preceding character or group.
- `*`: Matches 0 or more occurrences of the preceding character or group.

4. **Examples**

- **Extracting digits from a string**:

```
text = "The price is 50 dollars"
numbers = re.findall(r'\d+', text)
print(numbers)  # Output: ['50']
```

- **Checking if a string starts with a certain word**:

```
text = "Hello, world!"
if re.match(r'^Hello', text):
    print("String starts with 'Hello'")
```

- **Validating an email address**:

```
email = "test@example.com"
if re.match(r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$', email):
    print("Valid email")
else:
    print("Invalid email")
```

5. **Exercise:**

- **Exercise 1**: Write a function that extracts all phone numbers from a given string. Assume phone numbers follow the format `XXX-XXX-XXXX`.
- **Exercise 2**: Create a program that validates a given username. The username must start with a letter and contain only letters, digits, and underscores.

Would you like to work on these exercises, or would you like a deeper dive into any specific part of regular expressions?