String Formatting & Manipulation

1. Write a function that capitalizes the first letter of each word in a string.

```python
string = input("Enter a string: ")
print(f"The string is '{string.title()}'!")
```

2. Create a program that finds all substrings of a given string.

```python
string = input("Enter a string: ")

substrings = []
for i in range(len(string)):
    for j in range(i + 1, len(string) + 1):
        substrings.append(string[i:j])
print(substrings)
```

3. Write a function that replaces all vowels in a string with '*' symbol.

```python
string = input("Enter a string: ")

vowels = "aeiouAEIOU"
for v in vowels :
    string = string.replace(v, "*")
print(f"New String: {string}")
```

8. Create a function to remove HTML tags from a string.

```python
import re

def remove_tags(s):
    return re.sub(r'<[^>]+>', '', s)

html = "<p>Hello <i>World</i><p>"
print(remove_tags(html))
```

4. Develop a function that counts words, characters, and lines in a string.

5.4.py > string

```python
def word_count(s) :
    '''words = s.split()
    return len(words)'''
    count = 1
    for char in s :
        if char == ' ' : count += 1
    return count

def char_count(s) :
    '''characters = len(s)
    return len(characters)'''
    count = 0
    for char in s :
        if char != ' ' : count += 1
    return count

def line_count(s) :
    '''lines = s.split('\n')
    return len(lines)'''
    count = 1
    for char in s :
        if char == '\n' : count += 1
    return count

string = input("Enter a sentence: ")
string.strip()

print(f"The word count is {word_count(string)}")
print(f"The character count is {char_count(string)}")
print(f"The line count is {line_count(string)}")
```

5. Write a script to format a number as currency (e.g., 1000000 -> 1,000,000).

```python
def currency_format(number) :
    number_str = str(number)
    result = ""
    while len(number_str) > 3 :
        result = "," + number_str[-3:] + result
        number_str = number_str[:-3]
    result = number_str + result
    return result

number = int(input("Enter a number: "))
#print("{:,}".format(number))
print(currency_format(number))
```

6. Implement a function that validates a strong password based on given criteria.

```python
import re

def is_strong(password):
    if (len(password) >= 8 and
        re.search(r"[A-Z]", password) and
        re.search(r"[a-z]", password) and
        re.search(r"[0-9]", password) and
        re.search(r"[!@#$%^&*(),.?\":{}|<>]", password)):
        return True
    return False

password = input("Enter the password: ")
print(is_strong(password))
```

7. Write a script that encodes a string using Caesar cipher.

5.7.py > ...

```python
def cipher(text, shift):
    result = ""
    for char in text:
        if char.isalpha():
            base = ord('A') if char.isupper() else ord('a')
            result += chr((ord(char) + shift - base) % 26 + base)
        else:
            result += char
    return result

shift = int(input("Enter the shift: "))
text = input("Enter the text: ")
print(f"Text converted to cipher: {cipher(text, shift)}")
```

9. Develop a function that finds the longest palindromic substring.

5.9.py > ...

```python
def is_palindrome(s):
    return s == s[::-1]

string = input("Enter a string: ")
max_palindrome = ""

for i in range(len(string)):
    for j in range(i + 1, len(string) + 1):
        substring = string[i:j]
        if is_palindrome(substring) and len(substring) > len(max_palindrome):
            max_palindrome = substring

print(f"Longest palindromic string is: {max_palindrome}")
```

10. Implement a string compression algorithm (e.g., aabccccaaa -> a2b1c5a3).

5.10.py > ...

```python
def compressed_string(s):
    result = ""
    count = 1

    for i in range(1, len(s)):
        if s[i] == s[i-1]:
            count += 1
        else:
            result += (s[i-1] + str(count))
            count = 1
    result += (s[-1] + str(count))

    return result


string = input("Enter a string: ")
print(f"Compressed string is: {compressed_string(string)}")
```