Given the string `s = "Programming"`, retrieve the character at index 6.
s[6]
Given the string `s = "Python"`, retrieve the last character using slicing.
S[-1], s[len(s)-1]
Given the string `s = "Language"`, retrieve the substring `"Lang"`.
s[0:4]
Given the string `s = "PythonProgramming"`, retrieve the substring `"Pro"`.
s[6:9]
Given the string `s = "PythonProgrammingLanguage"`, retrieve the substring `"Programming"`.
s[6:17]
Given the string `s = "abcdefghijklm"`, retrieve every second index starting from the second index.
s[2::2]
Given the string `s = "abcdefabcdefabcdef"`, retrieve every third index from the entire string.
s[::3]
Given the string `s = "PythonProgramming"`, retrieve the substring `"nohtyP"` using slicing in reverse order.
s[-12:-18:-1]
Given the string `s = "HelloWorld"`, retrieve all characters from the start up to but not including index 5.
Given the string `s = "PythonLanguage"`, retrieve the substring starting from index 6 to the end.
Given the string `s = "Palindrome"`, reverse the entire string using slicing.
Given the string `s = "abcdefgh"`, retrieve every second character in reverse order.
Given the string `s = "abcdefghi"`, retrieve the substring `"def"` using both positive and negative indices.
Given the string `s = "Complete"`, retrieve the entire string using slice notation.
Given the string `s = "Programming"`, retrieve the substring `"gramm"` using only negative indices.
Given the string `s = "123456789"`, retrieve every second character in reverse order using a step of -2.
Given the string `s = "University"`, retrieve the first four characters and reverse them.
Given the string `s = "PythonLanguage"`, retrieve every alternate character starting from the third character.
Given the string `s = "DataScience"`, retrieve all characters from index -6 to the end of the string.
Given the string `s = "ArtificialIntelligence"`, retrieve the substring from the start to index -4.

Extract a single character:

- Given `s = "Programming"`, extract the character at index 4.
- S [4]

Extract the last character:

- Given `s = "Python"`, extract the last character using negative indexing.

Extract a substring from the start:

- Given `s = "Language"`, extract the substring "Lang".

Extract a substring from the middle:

- Given `s = "PythonProgramming"`, extract the substring "Pro".

Extract a substring using both positive and negative indices:

- Given `s = "PythonProgrammingLanguage"`, extract the substring "Programming" using both positive and negative indices.

Extract a substring using a step:

- Given `s = "abcdefghijklm"`, extract every second character starting from the second character.

Extract every third character:

- Given `s = "abcdefabcdefabcdef"`, extract every third character from the entire string.

Extract a substring in reverse:

- Given `s = "PythonProgramming"`, extract the substring "nohtyP" using slicing in reverse order.

Extract from the start to a specific index:

- Given `s = "HelloWorld"`, extract all characters from the start up to but not including index 5.

Extract from a specific index to the end:

- Given `s = "PythonLanguage"`, extract the substring starting from index 6 to the end.

Reverse the entire string:

- Given `s = "Palindrome"`, reverse the entire string using slicing.

Extract using a negative step:

- Given `s = "abcdefgh"`, extract every second character in reverse order.

Extract a middle part of the string using both positive and negative slicing:

- Given `s = "abcdefghi"`, extract "def" using both positive and negative indices.

Slice the entire string without specifying start and end:

- Given `s = "Complete"`, extract the entire string using slice notation.

Extract using only negative indices:

- Given `s = "Programming"`, extract the substring "gramm" using only negative indices.

Slice with step of -2:

- Given `s = "123456789"`, extract every second character in reverse order using a step of -2.

Extract a prefix and reverse it:

- Given `s = "University"`, extract the first four characters and reverse them.

Extract an alternating character sequence:

- Given `s = "PythonLanguage"`, extract every alternate character starting from the third character.

Extract from a negative index to the end:

- Given `s = "DataScience"`, extract all characters from index -6 to the end of the string.

Extract from start to a negative index:

- Given `s = "ArtificialIntelligence"`, extract the substring from the start to index -4.

1. **Reverse a String:**
    - Write a function `reverse_string(s)` that takes a string `s` and returns the string reversed.
2. **Check for Palindrome:**
    - Write a function `is_palindrome(s)` that checks if the given string `s` is a palindrome (reads the same forwards and backwards).

    Example:
    ```
    Input: "racecar"
    Output: True
    ```

3. **Count Vowels:**
    - Write a function `count_vowels(s)` that counts the number of vowels (`a, e, i, o, u`) in a given string `s`.

    Example:
    ```
    Input: "Programming"
    Output: 3
    ```

    -
4. **Remove Duplicates:**
    - Write a function `remove_duplicates(s)` that removes all duplicate characters from a given string `s`.

    Example:
    ```
    Input: "mississippi"
    Output: "misp”
    ```

5. **Find the Longest Word:**
    - Write a function `longest_word(s)` that takes a sentence `s` and returns the longest word in the sentence.

    Example:
    ```
    Input: "I love programming in Python"
    Output: "programming”
    ```

    -
6. **Count Substring Occurrences:**

- Write a function `count_substring(s, sub)` that takes a string `s` and a substring `sub` and counts how many times `sub` appears in `s`.

Example:

```
Input: "banana", "ana"
Output: 1
```

-

7. **Capitalize Every Word:**
   - Write a function `capitalize_words(s)` that takes a sentence `s` and capitalizes the first letter of each word.

Example:

```
Input: "hello world"
Output: "Hello World"
```

-

8. **Find the First Non-Repeating Character:**
   - Write a function `first_non_repeating(s)` that returns the first non-repeating character in a string `s`. If all characters are repeated, return `None`.

Example:

```
Input: "swiss"
Output: "w”
```

9. **Count Words in a Sentence:**
   - Write a function `count_words(s)` that takes a sentence `s` and counts the number of words in it. A word is defined as a sequence of characters separated by spaces.

Example:

```
Input: "The quick brown fox"
Output: 4
```