Pseudocode for BearING Search Engine Lab

Function: `is_vowel(character1: char) -> bool`

- Description: Checks if a given character is a vowel.
- Parameters:
  - `character1`: A single character to be checked.
- Returns:
  - `true` if the character is a vowel.
  - `false` otherwise.
- Logic:

```
    if character1 is 'a' or character1 is 'e' or character1 is 'i' or
character1 is 'o' or character1 is 'u':
        return true
    else:
        return false
```

Function: `is_consonant(character1: char) -> bool`

- Description: Checks if a given character is a consonant.
- Parameters:
  - `character1`: A single character to be checked.
- Returns:
  - `true` if the character is a consonant.
  - `false` otherwise.
- Logic:

```
    if ch is not a vowel:
        return true
    else:
        return false
```

Function: `ends_with_double_consonant(str: string) -> bool`

- Description: Checks if a string ends with two consecutive consonants.
- Parameters:
  - `str`: The input string to be checked.
- Returns:
  - `true` if the string ends with double consonants.
  - `false` if the string is less than 2 characters.
- Logic:

```
    if length of str is less than 2:
        return false
    else if last character of str is a consonant and second to last
character of str is a consonant:
        return true
```

```
    else:
        return false
```

Function: `ends_with_cvc(str: string) -> bool`

- Description: Checks if a string ends with a consonant, followed by a vowel, followed by another consonant.
- Parameters:
  - `str`: The input string to be checked.

- Returns:
  - `true` if the string ends with CVC pattern.
  - `false` if the string is less than 3 characters.

- Logic:

```
    if length of str is less than 3:
        return false
    else if last character of str is a consonant, second to last character
of str is a vowel, and third to last character of str is a consonant:
        return true
    else:
        return false
```

Function: `contains_vowel(str: string) -> bool`

- Description: Checks if a string contains at least one vowel.
- Parameters:
  - `str`: The input string to be checked.
- Returns:
  - `true` if the string contains a vowel.
  - `false` otherwise.
- Logic:

```
    for each character ch in str:
        if ch is a vowel:
            return true
    return false
```

Function: `count_consonants_at_front(str: string) -> int`

- Description: Counts the number of consecutive consonants at the beginning of the string.
- Parameters:
  - `str`: The input string to be checked.

- Returns:
  - The count of consecutive consonants at the front of the string.
- Logic:

```
declare count and initialize to 0

for each character ch in str:
    if ch is a consonant:
        increment count
    else:
        break

return count
```

Function: `count_vowels_at_back(str: string) -> int`

- Description: Counts the number of consecutive vowels at the end of the string.
- Parameters:
  - `str`: The input string to be checked.
- Returns:
  - The count of consecutive vowels at the back of the string.
- Logic:

```
declare count and initialize to 0

for each character ch in str (from the end):
    if ch is a vowel:
        increment count
    else:
        break

return count
```

Function: `ends_with(candidate: string, suffix: string) -> bool`

- Description: Checks if the inputted string ends with a specified suffix.
- Parameters:
  - `candidate`: The input string to be checked.
  - `suffix`: The suffix string to check for at the end of `candidate`.
- Returns:
  - `true` if `candidate` ends with `suffix`, `false` otherwise.
- Logic:

```
if candidate is empty and suffix is empty:
    return true
else if candidate is empty and suffix is not empty:
    return false

return candidate ends with suffix
```

Function: `new_ending(candidate: string, suffix_length: int, replacement: string) -> string`

- Description: Replaces the ending of the input string with a specified replacement.
- Parameters:
  - `candidate`: The input string to be modified.
  - `suffix_length`: The length of the suffix to be replaced.
  - `replacement`: The string to replace the last `suffix_length` characters of `candidate`.
- Returns:
  - A new string formed by replacing the last `suffix_length` characters of `candidate` with `replacement`.
- Logic:

```
    if length of candidate is less than suffix_length:
        return candidate

    return candidate without the last suffix_length characters, concatenated
with replacement
```