

Longest Substring Without Repeating Characters

Solving the classic Longest Substring Without Repeating Characters problem - best tackled using the sliding window technique.

Strategy: Sliding Window with Hash Map:

1. Use a hash map to store the last seen index of each character.
2. Maintain a window [left, right] of unique characters.
3. For each character $s[right]$:
 - If it's already in the map and its last seen index $\geq left$, move left to $map[c] + 1$ to avoid duplicates.
 - Update the character's last seen index.
 - Update the max length: $maxLen = \max(maxLen, right - left + 1)$

Time Complexity: $O(n)$ - each character is visited at most twice (once by right, once by left)
 $O(1)$ space if fixed alphabet (ASCII), else $O(n)$

Example Walkthrough:

- For "pwwkew":
- $p \rightarrow$ window: $p \rightarrow$ length: 1
 - $w \rightarrow pw \rightarrow$ length: 2
 - another $w \rightarrow$ move left to skip first $w \rightarrow$ window: w continue
 - $k \rightarrow wk \rightarrow$ length: 2
 - $e \rightarrow wke \rightarrow$ length: 3 ✓