4.7 LONGEST SUBSTRING WITHOUT REPEATING CHARACTERS

Question:

Given a string s, find the length of the longest substring without repeating characters.

AIM

To implement a Python program that computes the length of the longest substring without repeating characters using the sliding window technique.

ALGORITHM

1. Initialize a set seen to store characters in the current window.
2. Use two pointers left and right to define the sliding window.
3. Move right forward and add characters to seen until a duplicate is found.
4. When a duplicate is encountered, remove characters from the left until the duplicate is removed.
5. Track the maximum window size during traversal.
6. Return the maximum length found.

PROGRAM

```
def length_of_longest_substring(s):
    seen = {}
    left = max_len = 0
    for right, char in enumerate(s):
        if char in seen and seen[char] >= left:
            left = seen[char] + 1
        seen[char] = right
        max_len = max(max_len, right - left + 1)
    return max_len

s = input("Enter a string: ")
print("Length of longest substring without repeating characters:", length_of_longest_substring(s))
```

Input:

Enter a string: bifbfbinbi

Output:

```
Enter a string: bifbfbinbi
Length of longest substring without repeating characters: 4
>>>
```

RESULT:

Thus the program is successfully executed and the output is verified.

PERFORMANCE ANALYSIS:

- Time Complexity: O(n), where *n* is the length of the string.
- Space Complexity: O(min(n, m)), where *m* is the size of the character set.