1) String s = "abcabc bb"
   s.length();

   (String s) { Hash set<Character> set = new Hash set<>();
   int left = 0;
   int maxlength = 0;
   For (int right = 0; right < s.length(); right++) {
   while (set.contains (s.charAt(right))) {
   set.remove (s.Char M ('left)); left++;}
   set.add (s.charAt (right));
   max. length = math. max (maxlength, right. left+i); }
   return maxlength; }}

Output. length = 3

a) String s = "()[]{}";
   is valid (s);

   stack <character> stack = new stack <> ();
   for (char ch:s.toChar Array()) {
   if (ch == 'c' || ch = '{' || ch --'ʃ') stack push (ch)}
   else { if (stack. is Empty()) { return False; }
   char top = stack. pop();
   if ((ch == ')' & top! = 'c'] || (ch== '}' || top!=
   (ch :"]') || top!= 'ʃ') { return False ; }}}
   stack. is empty();

   Output : True.

Programiz
Online Java Compiler

Main.java

```java
import java.util.Stack;
public class Main {
    public static void main(String[] args) {
        String s = "()[]{}";
        System.out.println("Valid: " + isValid(s));
    }
    public static boolean isValid(String s) {
        Stack<Character> stack = new Stack<>();
        for (char ch : s.toCharArray()) {
            if (ch == '(' || ch == '{' || ch == '[') {
                stack.push(ch);
            } else {
                if (stack.isEmpty()) {
                    return false;
                }
                char top = stack.pop();
                if ((ch == ')' && top != '(') ||
                    (ch == '}' && top != '{') ||
                    (ch == ']' && top != '[')) {
                    return false;
                }
            }
        }
        return stack.isEmpty();
    }
}
```
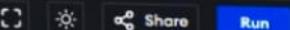
Output

Clear

```
Valid: true

--- Code Execution Successful ---
```

```java
import java.util.HashSet;
class LongestSubstring {
    public static void main(String[] args) {
        String s = "abcabcbb";
        System.out.println("Length: " + lengthOfLongestSubstring(s));
    }
    public static int lengthOfLongestSubstring(String s) {
        HashSet<Character> set = new HashSet<>();
        int left = 0;
        int maxLength = 0;
        for (int right = 0; right < s.length(); right++) {
            while (set.contains(s.charAt(right))) {
                set.remove(s.charAt(left));
                left++;
            }
            set.add(s.charAt(right));
            maxLength = Math.max(maxLength, right - left + 1);
        }
        return maxLength;
    }
}
```

Output

Length: 3

--- Code Execution Successful ---