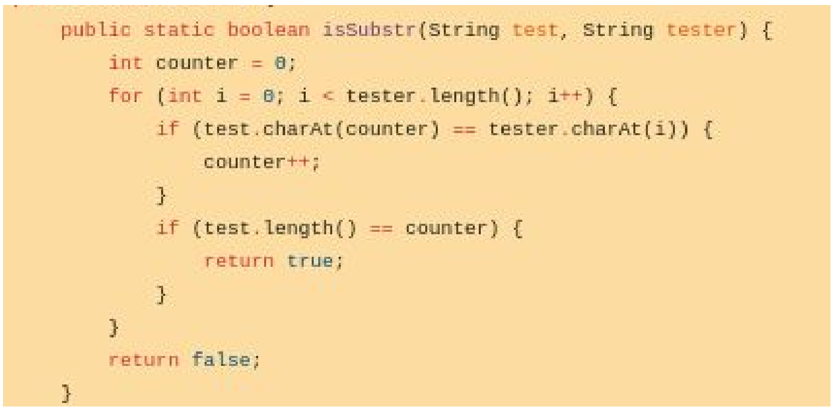
ICCS200: Assignment 4

Vikrom Narula
vikrom.nar@gmail.com

31/05/2019

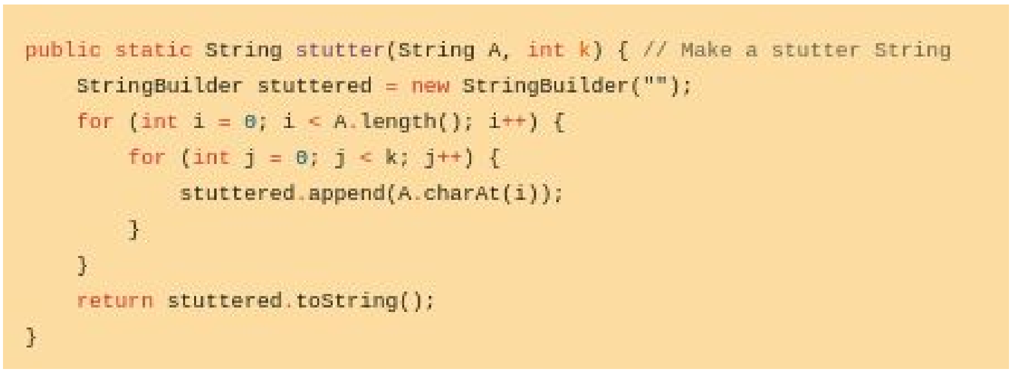
People in 1408 & 1409

Exercise 2:



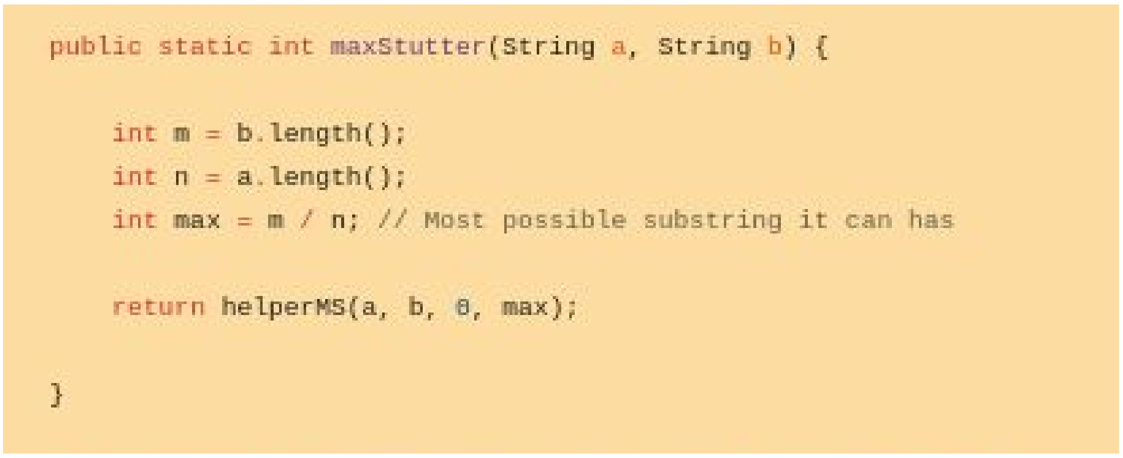
```java
    public static boolean isSubstr(String test, String tester) {
        int counter = 0;
        for (int i = 0; i < tester.length(); i++) {
            if (test.charAt(counter) == tester.charAt(i)) {
                counter++;
            }
            if (test.length() == counter) {
                return true;
            }
        }
        return false;
    }
```

This is $isSubstr$ takes $O(n)$ but due to we have to check all of the test but we also have to test to the tester length which means it will take $O(m+n)$ { m = tester.length() , n = test.length()}



```java
public static String stutter(String A, int k) { // Make a stutter String
    StringBuilder stuttered = new StringBuilder("");
    for (int i = 0; i < A.length(); i++) {
        for (int j = 0; j < k; j++) {
            stuttered.append(A.charAt(i));
        }
    }
    return stuttered.toString();
}
```

This stutter take $O(n \cdot k)$ times due to first loop takes n times and the second takes k times hence $n \cdot k$ { n = A.length }

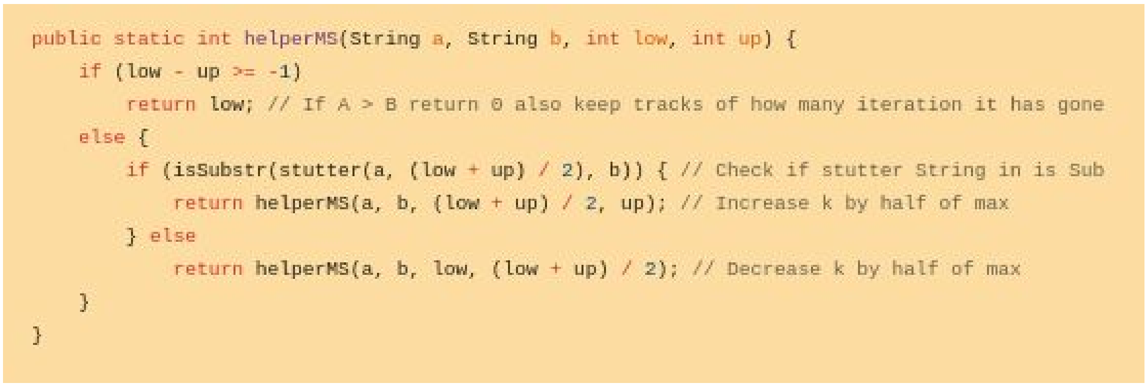


```
public static int maxStutter(String a, String b) {

    int m = b.length();
    int n = a.length();
    int max = m / n; // Most possible substring it can has

    return helperMS(a, b, 0, max);

}
```



```
public static int helperMS(String a, String b, int low, int up) {
    if (low - up >= -1)
        return low; // If A > B return 0 also keep tracks of how many iteration it has gone
    else {
        if (isSubstr(stutter(a, (low + up) / 2), b)) { // Check if stutter String in is Sub
            return helperMS(a, b, (low + up) / 2, up); // Increase k by half of max
        } else
            return helperMS(a, b, low, (low + up) / 2); // Decrease k by half of max
    }
}
```

helperMS cuts the max possible substring in half every time so we know that it take $log\ max$ and hence we know max is $\frac{m}{n}$ this function also uses stutter which as established that it costed $O(m+n)$ and due to this helperMS is $O((m+n)\cdot log\ (\frac{m}{n}))$ Hence all the maxStutter uses helperMS and others is O(1) the run times is $O((m+n)\cdot log\ (\frac{m}{n}))$.

Exercise 4:

We a let $g(n) = \frac{f(n)}{n+1}$

$$\frac{n*f(n)}{n*(n+1)} = \frac{2n+(n+1)f(n-1)}{n*(n+1)} \Rightarrow g(n) = \frac{2}{n+1} + \frac{(n+1)f(n-1)}{n*(n+1)} \Rightarrow g(n) = \frac{2}{n+1} + g(n-1)$$

Now it and recurrence and if we keep going we will get …

$g(n) = g(1) + \frac{2}{3} + \frac{2}{4} + \frac{2}{5} + ... + \frac{2}{n+1} = 2(\frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + ... + \frac{1}{n+1})$

$\frac{f(n)}{n+1} = 2\ H_n + \frac{2}{n+1} \Rightarrow f(n) = (n+1)(2\ H_n + \frac{2}{n+1})$

We have to find rung time we know $H_n \leq O(ln\ n)$, $ln\ n\ > \frac{1}{n}$
$f(n) = O(n) * O(ln\ n) \Rightarrow O(n * ln(n))$