

Lecture 22 - Nov. 26

Recursion

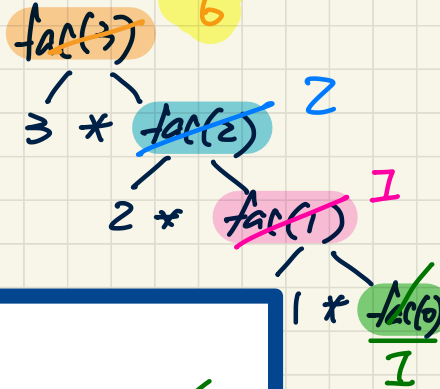
***Tracing: Factorial, Fibonacci Sequence
Recursion on Str.: Palindrome, Reversal
Exam Info***

Announcements/Reminders

- **Lab5** released
 - + Required study: **Abstract Classes & Interfaces**
- Learning resources on **Recursion**
- **Exam** Review Sessions eClass Polling

Recursive Solution in Java: factorial

$$n! = \begin{cases} 1 & \text{if } n = 0 \\ n \cdot (n-1)! & \text{if } n \geq 1 \end{cases}$$



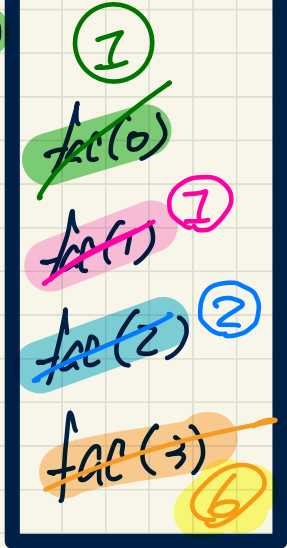
```
int factorial(int n) {  
    int result;  
    if (n == 0) { /* base case */ result = 1; }  
    else { /* recursive case */  
        .. result = n * factorial(n - 1);  
    }  
    return result;  
}
```

Annotations:

- Handwritten numbers: 3, 2, 1, 0 next to the recursive calls.
- Handwritten numbers: 6, 2, 1, 1 next to the return values.
- Handwritten text: "assigned and available for use" with an arrow pointing to the recursive call.

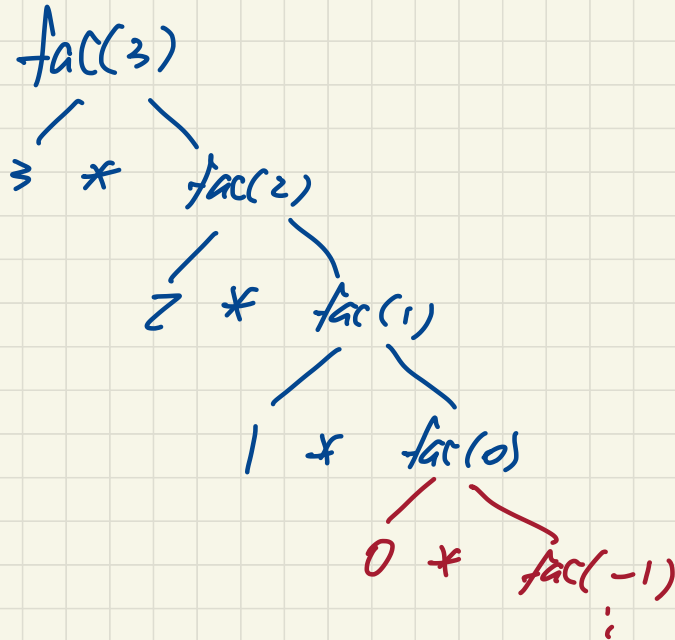
Example: factorial(3)

Runtime Stack



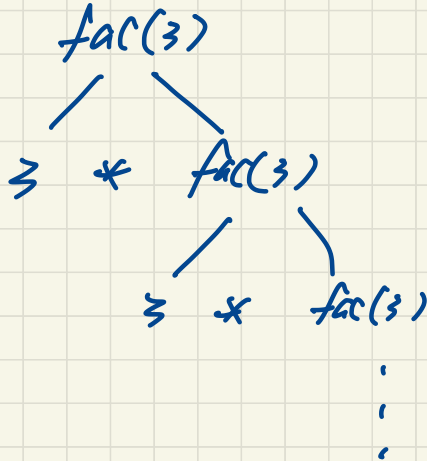
Common **Errors** of Recursion (1)

```
int factorial(int n) {  
    return n * factorial(n - 1);  
}
```



Common **Errors** of Recursion (2)

```
int factorial(int n) {  
    if(n == 0) { /* base case */ return 1; }  
    → else { /* recursive case */ return n * factorial(n); }  
}
```



Recursive Solution: Fibonacci Numbers

$F = 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \dots$

1 2 3 4

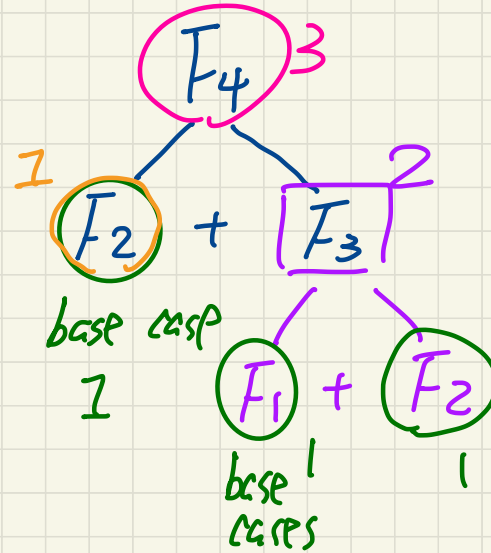
base
case

$F_6 + F_7$

F_8

F_1

F_2



Recursive Solution in Java: Fibonacci Numbers

$$F_n = \begin{cases} 1 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \\ F_{n-1} + F_{n-2} & \text{if } n > 2 \end{cases}$$

```
int fib(int n) {  
    int result;  
    if(n == 1) { /* base case */ result = 1; }  
    else if(n == 2) { /* base case */ result = 1; }  
    else { /* recursive case */  
        result = fib(n - 1) + fib(n - 2);  
    }  
    return result;  
}
```

two recursive calls

Example: fib(4)

Runtime Stack

Use of String

$s.length()$

$s \leadsto \text{"Hello"}$

$s \leadsto$

0	1	2	3	4
H	e	l	l	o

```
public class StringTester {  
    public static void main(String[] args) {  
        String s = "abcd";  
        System.out.println(s.isEmpty()); /* false */  
        /* Characters in index range [0, 0) */  
        String t0 = s.substring(0, 0);  $[0, -1]$   $i > j \rightarrow$  Empty  
        System.out.println(t0); /* "" */  
        /* Characters in index range [0, 4) */  
        String t1 = s.substring(0, 4);  
        System.out.println(t1); /* "abcd" */  
        /* Characters in index range [1, 3) */  
        String t2 = s.substring(1, 3);  
        System.out.println(t2); /* "bc" */  
        String t3 = s.substring(0, 2) + s.substring(2, 4);  
        System.out.println(s.equals(t3)); /* true */  
        for(int i = 0; i < s.length(); i++) {  
            System.out.print(s.charAt(i));  
        }  
        System.out.println();  
    }  
}
```

$s.substring(i, j)$
 $\hookrightarrow$
 $[i, j)$
"
 $[i, j-1]$

For any given string S,

given int. i s.t. $0 \leq i \leq s.length() - 1$

✓ S. equals (s.substring(0, i) + s.substring(i, s.length()))

Recursions on Strings

ex. "abcd"

Palindrome

"racecar"

"aracecars"

"racecar"

→ strictly smaller problem

!= no need to recur further.

Reversal

rev("abcd")

rev("bcd") + a
d c b

Number of Occurrences

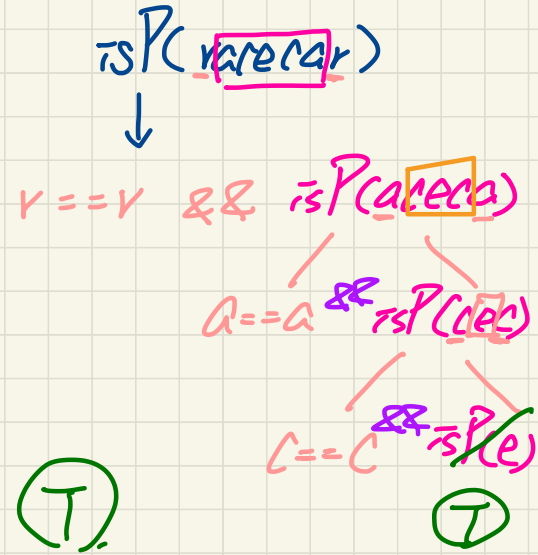
"abca"

'a'

'b'

Problem: Palindrome

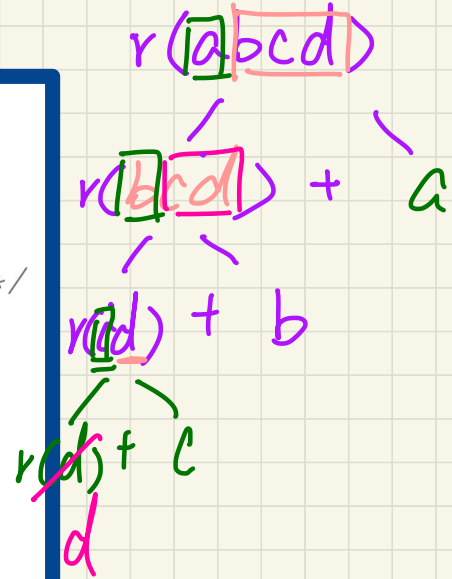
```
boolean isPalindrome(String word) {  
    if(word.length() == 0 || word.length() == 1) {  
        /* base case */  
        return true;  
    }  
    else {  
        /* recursive case */  
        char firstChar = word.charAt(0);  
        char lastChar = word.charAt(word.length() - 1);  
        String middle = word.substring(1, word.length() - 1);  
        return  
            firstChar == lastChar  
            /* See the API of java.lang.String.substring. */  
            && isPalindrome(middle);  
    }  
}
```



ex. `isP(racecar)`

Problem: Reverse of a String

```
String reverseOf (String s) {  
    if(s.isEmpty()) { /* base case 1 */  
        return "";  
    }  
    else if(s.length() == 1) { /* base case 2 */  
        return s;  
    }  
    else { /* recursive case */  
        String tail = s.substring(1, s.length());  
        String reverseOfTail = reverseOf (tail);  
        char head = s.charAt(0);  
        return reverseOfTail + head;  
    }  
}
```



Exam Info

- When: 10am to 1pm, Sunday, December 8
- Where: TC Sobeys
- Format: Some Multiple Choice & Mostly Written
- Coverage: **Everything** (lecture materials & labs)
 - + slides, iPad notes, code examples
 - + <https://codingbat.com/java/Recursion-1>
- Restrictions:
 - + No data sheet
 - + No sketch paper (Exam booklet includes it)
- What you should bring:
 - + Valid, Physical Photo ID (strict)
 - + Water/Snack

1. explanation
2. output
3. justification
4. code