# Data Structure

(Java programming)


***Chapter 03.***

# Contents

- String
- Binary Recursion
- Practice

- ***String***

# String

https://docs.oracle.com/javase/7/docs/api/java/lang/String.html

### Method Summary

Methods

| Modifier and Type | Method and Description |
|---|---|
| char | **charAt**(int index)<br>Returns the char value at the specified index. |
| int | **codePointAt**(int index)<br>Returns the character (Unicode code point) at the specified index. |
| int | **codePointBefore**(int index)<br>Returns the character (Unicode code point) before the specified index. |
| int | **codePointCount**(int beginIndex, int endIndex)<br>Returns the number of Unicode code points in the specified text range of this String. |
| int | **compareTo**(String anotherString)<br>Compares two strings lexicographically. |
| int | **compareToIgnoreCase**(String str)<br>Compares two strings lexicographically, ignoring case differences. |
| String | **concat**(String str)<br>Concatenates the specified string to the end of this string. |
| boolean | **contains**(CharSequence s)<br>Returns true if and only if this string contains the specified sequence of char values. |
| boolean | **contentEquals**(CharSequence cs)<br>Compares this string to the specified CharSequence. |
| boolean | **contentEquals**(StringBuffer sb)<br>Compares this string to the specified StringBuffer. |
| static String | **copyValueOf**(char[] data)<br>Returns a String that represents the character sequence in the array specified. |
| static String | **copyValueOf**(char[] data, int offset, int count)<br>Returns a String that represents the character sequence in the array specified. |
| boolean | **endsWith**(String suffix)<br>Tests if this string ends with the specified suffix. |
| boolean | **equals**(Object anObject)<br>Compares this string to the specified object. |

# • String

```java
public class StringTest {
    public static void main(String[] args) {

        String msg = "Hello Smart Computing Laboratory";
        String word = "datastructure";

        System.out.println("Character at index 1 is " + msg.charAt(1));
        System.out.println("Length of msg is " + msg.length());
        if(word.equals("datastructure")) {
            System.out.println("String word is same with \"datastructure\"");
        }

        String subMsg = msg.substring(0, 4);

        System.out.println("Substring of msg is " + subMsg);

        int num = 10;
        String strNum = String.valueOf(num);

        System.out.println("Value \"num\" is Integer");
        System.out.println("Value \"strNum\" is String");

        System.out.println("Length of strNum is " + strNum.length());

    }

}
```

```
Character at index 1 is e
Length of msg is 32
String word is same with "datastructure"
Substring of msg is Hell
Value "num" is Integer
Value "strNum" is String
Length of strNum is 2
```

- ***Binary Recursion***

# • Binary Recursion

- Binary recursion occurs whenever there are two recursive calls for each non-base case
  - E.g., adding all the numbers in an integer array ***A*** using binary recursion

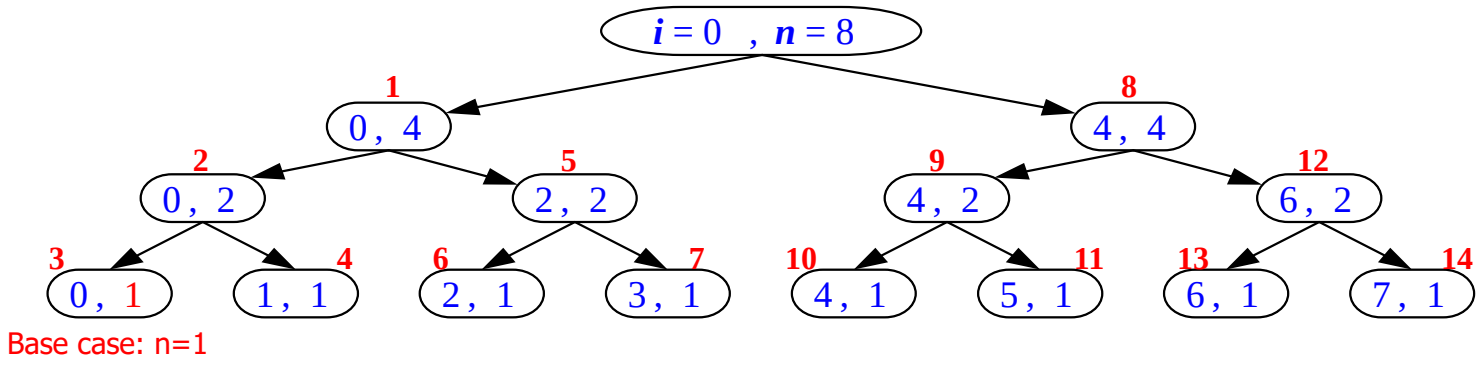
```
Algorithm BinarySum(A, i, n):
Input: array A and int i and n
Output: sum of n int in A starting at i
   if n = 1 then
      return A[i]
   else
      return BinarySum(A, i, ⌈n/2⌉) + BinarySum(A, i + ⌈n/2⌉, n/2)
```

E.g.) sum(1, 2, 3, 4, 5, 6, 7, 8)

→ sum(1, 2, 3, 4) + sum(5, 6, 7, 8)

→ { sum(1, 2) + sum(3, 4) } +
{ sum(5, 6) + sum(7, 8) }

→ …




**[ Visual trace of BinarySum(A, 0, 8) ]**

# • Binary Recursion

```
public class BinaryTest {
    public static int BinarySum(int[] arr, int i, int n) {
        if (n == 1) {
            return arr[i];
        }
        else {
            return BinarySum(arr, i, n/2) + BinarySum(arr, i+n/2, n/2);
        }
    }
    public static void main(String[] args) {

        int[] array = {1,2,3,4,5,6,7,8};
        System.out.println(BinarySum(array, 0, 8));
    }

}
```

- ***Practice***

# • Practice - 1

Linear Recursive Algorithm을 이용하여 문자열에 포함 된 'm'의 수를 계산하시오

String str = "my mom loves me!"

**Algorithm** StringCounter(s, i)

**Input** : String s, Integer i

**Output** : the number of characters in string

```
if  i = s.length then
        if s[i] = 'm' then return 1
        else return 0
else
        if s[i] = 'm' then return 1 + StringCounter(s, i+1)
        else return 0 + StringCounter(s, i+1)
```

# • 과제

**과제1.**
Practice1을 자신만의 알고리즘을 이용하여 만들어 보시오

조건1. Linear Recursive Algorithm을 사용할 것
조건2. 문자열로 "my mom loves me!"를 사용할 것
조건3. Practice1과 다른 방법을 이용할 것

**과제2.**
Binary Recursive Algorithm을 이용하여 문자열에 포함 된 'm'의 수를 계산하시오

조건1. Binary Recursive Algorithm을 사용할 것
조건2. 문자열로 "my mom loves me!"를 사용할 것

**제출기한 : 2017/03/26(일) 24:00까지**

# • Submission

**e-mail : 2017kudatastructure@gmail.com**

**Attach methods :**

1. Create zip file (java project folder)
2. Modify the file names :
   ID_name_datastructure[Classcode].dat
   ex ) 201173378_이명재_datastructure[2403-1].dat
3. e-mail title :
   datastructure_name_chapter
   ex ) datastructure_최수용_chapter2[2404-1]

**Classcode**

| | | |
|---|---|---|
| 2403-1 **[A-1반]** | 2403-2 **[A-2반]** | |
| 2404-1 **[B-1반]** | 2404-2 **[B-2반]** | 2405 **[C반]** |