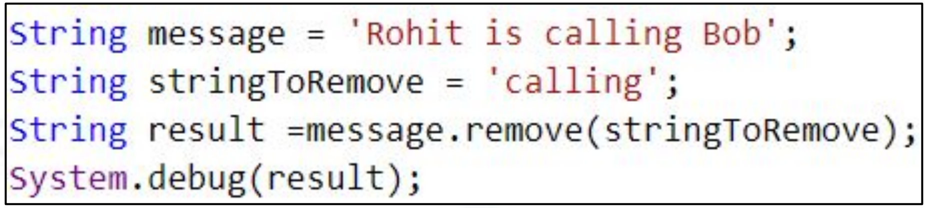
String Class Methods

- **remove:**
  - This method delete the string from given specified string.
    - This is useful when we need to delete certain string but we don't know exact location of that string.
  - This method is case-sensitive. So if the same string is appears but case is different, then this method will not work.
  - If the string that is targeted to be removed exists more than once in the original string, then all occurrences are removed.

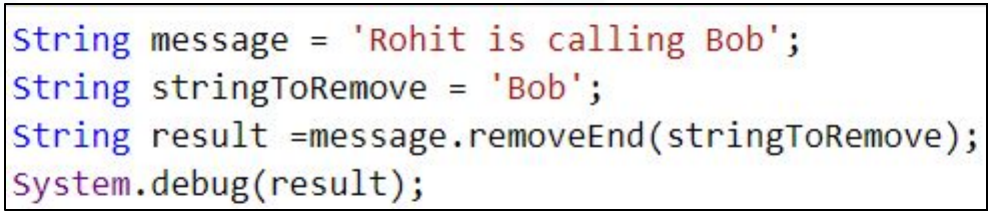


```
String message = 'Rohit is calling Bob';
String stringToRemove = 'calling';
String result =message.remove(stringToRemove);
System.debug(result);
```

Output: Rohit is Bob

String Class Methods

- **removeEnd:**
  - This method delete the specified substring only if it appears at the end of the String.
  - This method is case-sensitive. So if the same string appears but case is different, then this method will not work.

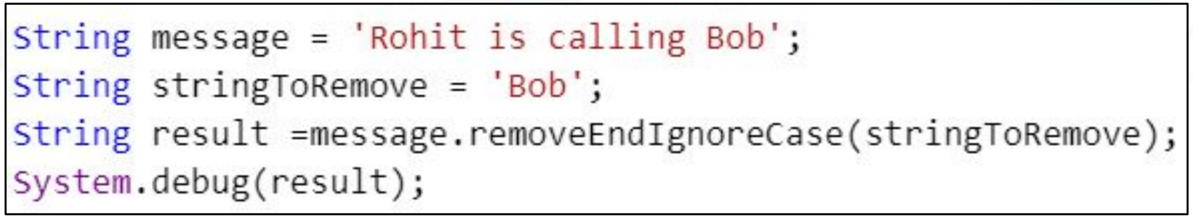


```
String message = 'Rohit is calling Bob';
String stringToRemove = 'Bob';
String result =message.removeEnd(stringToRemove);
System.debug(result);
```

Output: Rohit is calling

String Class Methods

- **removeEndIgnoreCase:**
  - This method delete the string from given specified string but only if it occurs at end.
  - This method is not case-sensitive.



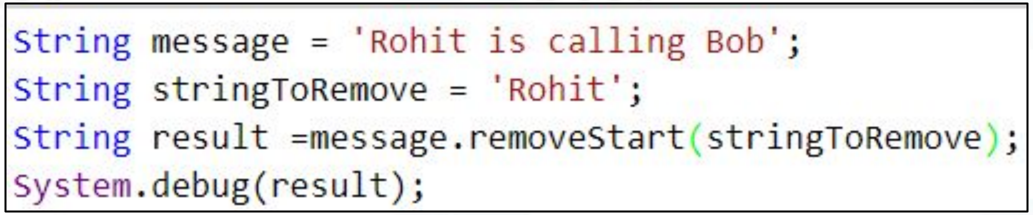
```
String message = 'Rohit is calling Bob';
String stringToRemove = 'Bob';
String result =message.removeEndIgnoreCase(stringToRemove);
System.debug(result);
```

Output: Rohit is calling

String Class Methods

- **removeStart:**
  - This method delete the specified substring only if it occurs at the beginning of the String.

  - This method is case-sensitive. So if the same string appears but case is different, then this method will not work.

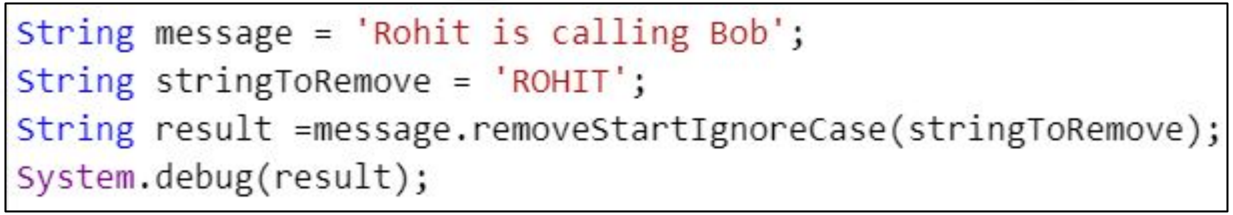


```
String message = 'Rohit is calling Bob';
String stringToRemove = 'Rohit';
String result =message.removeStart(stringToRemove);
System.debug(result);
```

Output: is calling Bob

String Class Methods

- **removeStartIgnoreCase:**
  - This method delete the string from given specified string but only if it occurs at beginning.
  - This method is **NOT** case-sensitive .


```
String message = 'Rohit is calling Bob';
String stringToRemove = 'ROHIT';
String result =message.removeStartIgnoreCase(stringToRemove);
System.debug(result);
```

Output: is calling Bob

Assignment

- Write a program in Apex with two String variables and assign it with
  - ‘PROGRAMMER says a programmer will be a programmer’
  - and ‘programmer’ respectively.
- You have to remove text of **second variable from the first variable** to so that you can get this final output as below. (Do not use values directly)

  After removing programmer: PROGRAMMER says a will be a
  After removing from start: says a programmer will be a programmer
  After removing from end: PROGRAMMER says a programmer will be a

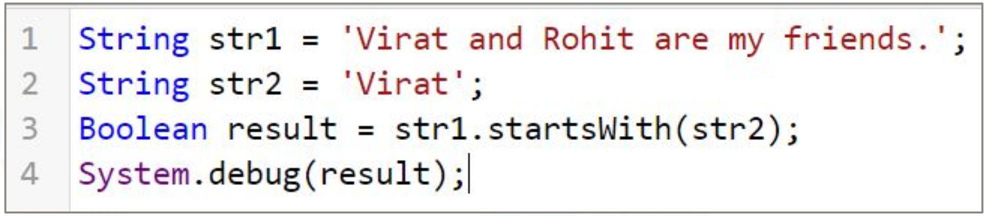
  [Hint: Use String methods. Pay attention to the case of the text while assigning values to the variables]

Homework 5

- Write a program in Apex with two String variables.
- Assign it with ‘Emily is calling Emily to party with Emily’ and ‘Emily’ respectively.
- Remove text of second variable from the first variable.
- Print the output as following -
  Original: Emily is calling Emily to party with Emily
  After Removal: is calling to party with
  Only from Start: is calling Emily to party with Emily
  Only from End: Emily is calling Emily to party with

String Class Methods : (Continue)

- **startsWith:**
  - This method returns true if given string starts with prefix provided in the method.
  - If method does not starts with given prefix, then this method returns false.
  - This method is case-sensitive.
  - There is case insensitive version as well: startsWithIgnoreCase()

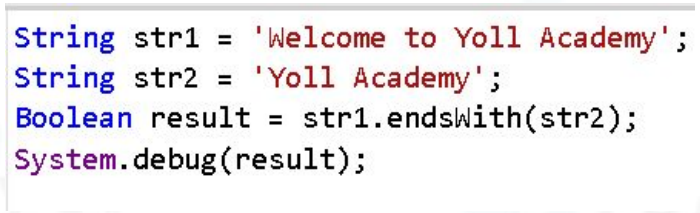


```
String str1 = 'Virat and Rohit are my friends.';
String str2 = 'Virat';
Boolean result = str1.startsWith(str2);
System.debug(result);
```

Output: true

String Class Methods : (Continue)

- **endsWith:**
  - This method returns true if given string starts with suffix provided in the method.
  - If method does not ends with given suffix, then this method returns false.
  - This method is case-sensitive.
  - There is case insensitive version as well: endsWithIgnoreCase()



```
String str1 = 'Welcome to Yoll Academy';
String str2 = 'Yoll Academy';
Boolean result = str1.endsWith(str2);
System.debug(result);
```

Output: true

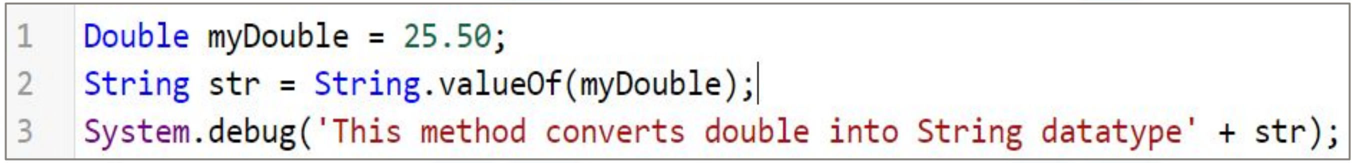
Assignment

- Write a program in Apex with a String variable and assign ‘Apex is a programming language used in Salesforce’
- print whether the text is having a ‘Apex’ as prefix and ‘Salesforce’ as suffix or not.
  Sample Output:
  Text is starting with Apex: true
  Text is starting with Salesforce: false
  Text is ending with Apex: false
  Text is ending with Salesforce: true

  [Direction: Use proper string methods to print true or false instead of printing them directly in single quotes]

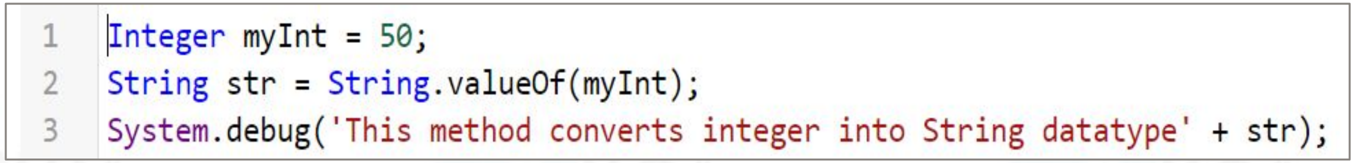
String Class Methods

- **valueOf:**
  - This method converts one datatype’s value into String datatype.



```
Double myDouble = 25.50;
String str = String.valueOf(myDouble);
System.debug('This method converts double into String datatype' + str);
```

Output: This method converts double into String datatype 25.50

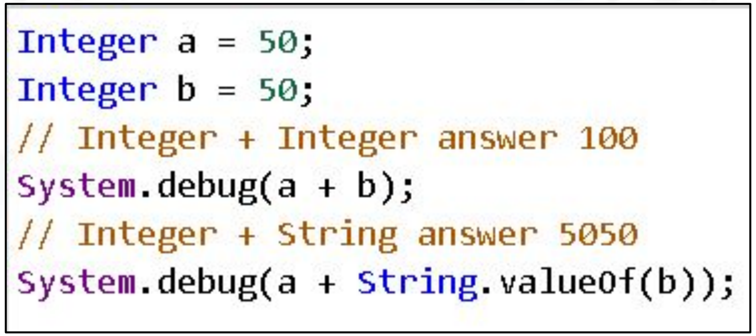


```
Integer myInt = 50;
String str = String.valueOf(myInt);
System.debug('This method converts integer into String datatype' + str);
```

Output: This method converts integer into String datatype 50

String Class Methods

- valueOf



```
Integer a = 50;
Integer b = 50;
// Integer + Integer answer 100
System.debug(a + b);
// Integer + String answer 5050
System.debug(a + String.valueOf(b));
```

Output: 100
5050

Assignment

- Write a program in Apex and create variables to assign following values. (integer and string accordingly)
    - 1
    - 4
    - 25
    - 30
    - Yoll
    - @
    - You
- Using these variables only, print the output as shown below

  Sample Output:

  Yoll 4 You

  12530Yoll@

  ****Only variables is allowed inside system.debug***

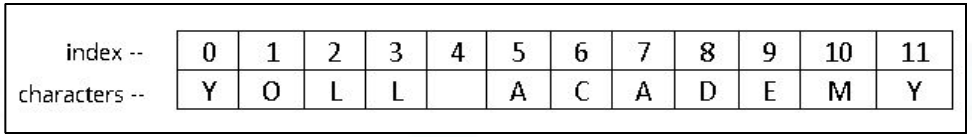
Homework 6

- Write a program in Apex using 3 variables
- Assign the variables with each of the following values.
  - option1 = ‘Good Morning, Have a wonderful day!’
  - option2 = ‘Good Afternoon, How are you?’
  - option3 = ‘Good Night, Sleep Tight.’
- Create another variable, and assign with any of the above options:
  - String userInput = option1/option2/option3
- Using proper String methods, print the output as following -

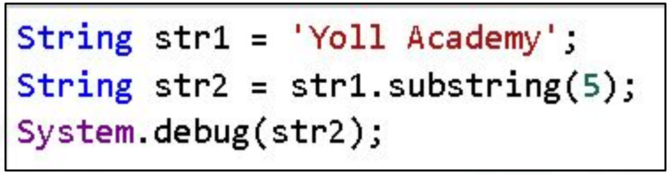
  Is it morning for user?: <TRUE/FALSE>
  Is it afternoon for user?: <TRUE/FALSE>
  Is it night for user?: <TRUE/FALSE>
  Did user ask a question? (?) : <TRUE/FALSE>
  Did user make a statement (.) : <TRUE/FALSE>
  Did user use exclamation? (!) : <TRUE/FALSE>

String Class Methods

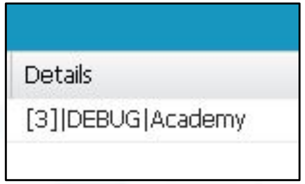
- **substring:**
  - Returns a new String that begins with the character at the specified startIndex and extends to the end of the String.



index --	0	1	2	3	4	5	6	7	8	9	10	11
characters --	Y	O	L	L		A	C	A	D	E	M	Y



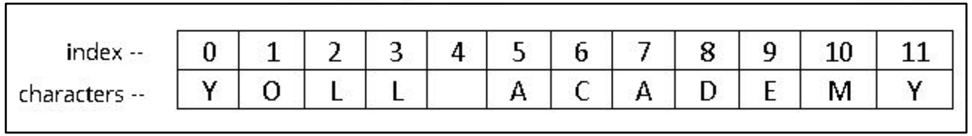
```
String str1 = 'Yoll Academy';
String str2 = str1.substring(5);
System.debug(str2);
```



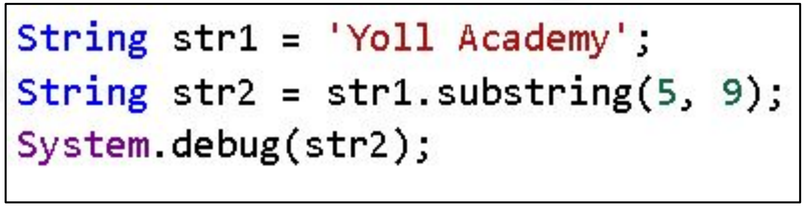
```
Details
[3]|DEBUG|Academy
```

String Class Methods

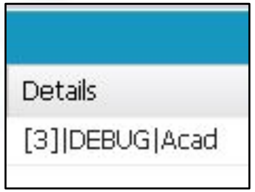
- **substring:** **(version 2)**
  - In the second version of this method we can pass two indexes: **startIndex** and **endIndex**.
  - It returns a new String that begins with the character at the startIndex and extends to the character at endIndex - 1.



index --	0	1	2	3	4	5	6	7	8	9	10	11
characters --	Y	O	L	L		A	C	A	D	E	M	Y



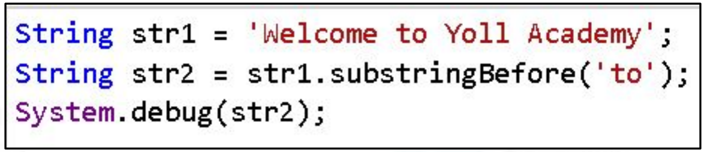
```
String str1 = 'Yoll Academy';
String str2 = str1.substring(5, 9);
System.debug(str2);
```



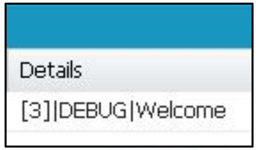
```
Details
[3]|DEBUG|Acad
```

String Class Methods

- **substringBefore:**
  - Returns the substring that occurs before the first occurrence of the specified separator.

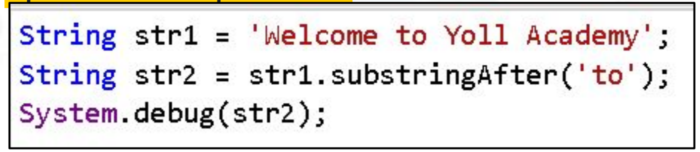


```
String str1 = 'Welcome to Yoll Academy';
String str2 = str1.substringBefore('to');
System.debug(str2);
```

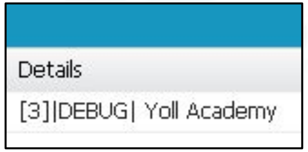


```
Details
[3]|DEBUG|Welcome
```

- **substringAfter:**
  - Returns the substring that occurs after the first occurrence of the specified separator.



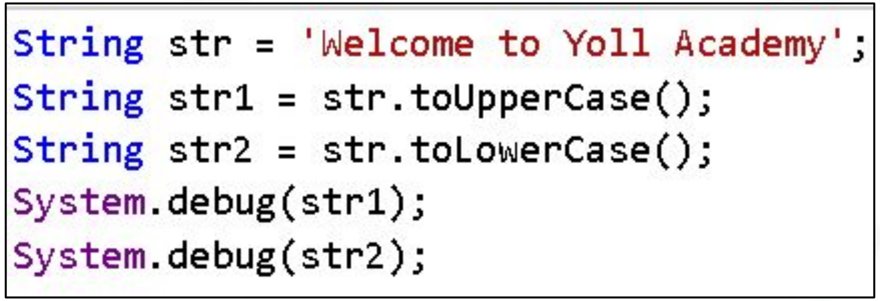
```
String str1 = 'Welcome to Yoll Academy';
String str2 = str1.substringAfter('to');
System.debug(str2);
```



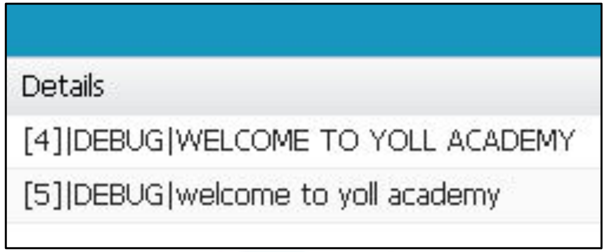
```
Details
[3]|DEBUG| Yoll Academy
```

String Class Methods

- **toUpperCase:**
  - Converts all of the characters in the String to uppercase.
- **toLowerCase:**
  - Converts all of the characters in the String to lowercase.



```
String str = 'Welcome to Yoll Academy';
String str1 = str.toUpperCase();
String str2 = str.toLowerCase();
System.debug(str1);
System.debug(str2);
```



```
Details
[4]|DEBUG|WELCOME TO YOLL ACADEMY
[5]|DEBUG|welcome to yoll academy
```

Capitalize vs toUppercase

- Difference between Capitalize and toUppercase

String word = 'hello world';

System.debug(word.capitalize()); → **H**ello world

System.debug(word.toUpperCase()); → **HELLO WORLD**

Assignment

- Write a program in Apex with a String variable and assign ‘My name is Emily’
- Print the output as shown below using String methods
  Output:
  Original Text: My name is Emily
  Name in Text: EMILY

Assignment

- Write a program in Apex with a String variable and assign 'Washington, D.C. is capital of USA' and print the output as shown below.
  Sample Output:
  Original Text: Washington, D.C. is capital of USA
  Capital: Washington, D.C.
  Country: USA

Assignment

- Write a program in Apex with a String variable and assign ‘Salesforce use Apex as a Programming Language’ and print the output as shown below.
  Sample Output:
  Original: Salesforce use Apex as a Programming Language
  Uppercase: SALESFORCE USE APEX AS A PROGRAMMING LANGUAGE
  Lowercase: salesforce use apex as a programming language