

Name: ZISHNENDU SARKER

Roll: 2K19/CO/450

Java Programming

Lab assignment 07

Java Program to Convert String to Date

Theory: We convert String objects into Date objects. We'll start with the new Date-Time API, java.time, that was introduced in Java 8 before looking at the old java.util. Date data type is also used for representing dates.

Code:

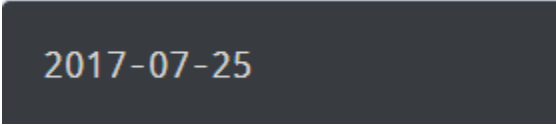
```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class TimeString {

    public static void main(String[] args) {
        // Format y-M-d or yyyy-MM-d
        String string = "2017-07-25";
        LocalDate date = LocalDate.parse(string, DateTimeFormatter.ISO_DATE);

        System.out.println(date);
    }
}
```

Output:



2017-07-25

Learning Output:

In the above program, we've used the predefined format ISO_DATE that takes date string in the format 2017-07-25 or 2017-07-25+05:45'. The LocalDate's parse() function parses the given string using the given format.

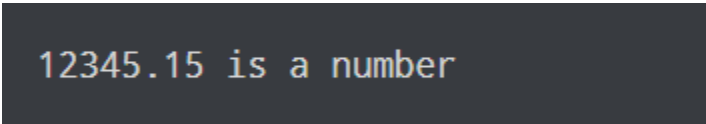
Java Program to Check if a String is Numeric

Theory: Let us see an example using Integer.parseInt(). We can create a user-defined function to check if a string is numeric or not. In the below example, the isInteger() method validates if a string is numeric. We use the Integer.parseInt() method within the try-catch block and try to parse the input string. If it successfully parses, we return true, else return false within the catch block. This is because the parseInt() method throws NumberFormatException if it cannot parse the string. We can then directly call this function from the main method to check if a string is numeric.

Code:

```
public class Numeric {  
  
    public static void main(String[] args) {  
  
        String string = "12345.15";  
        boolean numeric = true;  
  
        try {  
            Double num = Double.parseDouble(string);  
        } catch (NumberFormatException e) {  
            numeric = false;  
        }  
  
        if(numeric)  
            System.out.println(string + " is a number");  
        else  
            System.out.println(string + " is not a number");  
    }  
}
```

Output:



```
12345.15 is a number
```

Learning Output:

We have a String named string that contains the string to be checked. We also have a boolean value numeric which stores if the final result is numeric or not. To check if the string contains numbers only, in the try block, we use Double's parseDouble() method to convert the string to a

Double. If it throws an error (i.e. NumberFormatException error), it means the string isn't a number and numeric is set to false. Else, it's a number.

Java Program to Capitalize the first character of each word in a String

Theory: A naive solution is to split the given text using space as a delimiter. Then we iterate through the word array, capitalize the first character of each word, and append it to a StringBuilder along with space. Finally, return the string representation of the StringBuilder. The simplest solution is to use the WordUtils class from Apache Commons Lang. It already provides a capitalize() method that serves the same purpose.

Code:

```
class Main {  
    public static void main(String[] args) {  
  
        // create a string  
        String name = "srusti";  
  
        // create two substrings from name  
        // first substring contains first letter of name  
        // second substring contains remaining letters  
        String firstLetter = name.substring(0, 1);  
        String remainingLetters = name.substring(1, name.length());  
  
        // change the first letter to uppercase  
        firstLetter = firstLetter.toUpperCase();  
  
        // join the two substrings  
        name = firstLetter + remainingLetters;  
        System.out.println("Name: " + name);  
  
    }  
}
```

Learning Output:

Using Java 8 Streams The easiest way to capitalize the first character of each word of a string is by using Java 8 Stream API: String str = "welcome to java"; String output = Arrays.stream(str.split("\\s+")).map(t -> t.substring(0, 1).toUpperCase() + t.substring(1)).collect(Collectors.joining(" ")); System.out.println(output);