Think and Tell

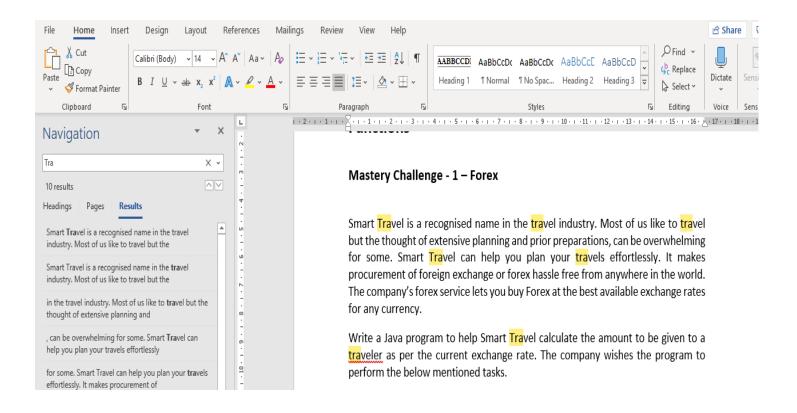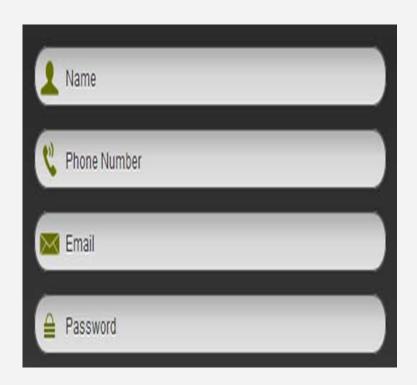**How can you find files saved in different formats in your system?**

What does the "find and replace" feature help you to do in a word document?
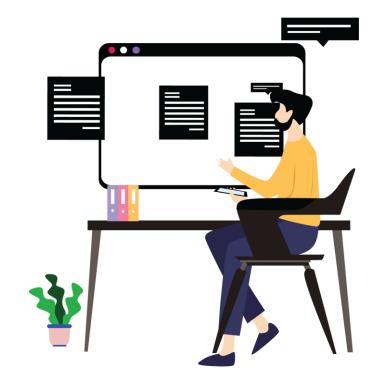
Registration Form

Following form filling guidelines are generally provided for filling registration forms

- Valid phone numbers
  - Must Start with 7,8,9
  - Should not be more than 10 digits

- Valid email address
  - Must have '@' symbol in between

How can you ensure that users enter the right data?

String Manipulations with RegEx

Learning Objectives

- Define regular expression in Java

- Apply pattern class and its methods

- Implement matcher class and its methods

- Differentiate between StringBuffer and StringBuilder

- Compare String, StringBuffer and StringBuilder

Regular Expression

How do we match a pattern to perform different string operations in Java?

How do we split a string based on a specific pattern?

What is a Regular Expression?

- **Regular Expression** (RegEx) is a sequence of characters that form a search pattern, mainly to be used in string matching

- A regular expression either matches a text, part of the text, or it fails to match

- RegEx is kind of a language within a language

- There are several ways to define a regular expression:
  - Literals
  - Character classes
  - Metacharacters
  - Quantifiers

Character Class in Regular Expression

Regular Expression	Description
[abc]	Matches the text if it contains either a, b or c. This should be only once
[^abc]	^ denotes negation, single characters except for a, b, or c are taken
[a-zA-Z]	a to z and A to Z are considered
[a-dx-z]	Union of both ranges a to d and x to z
[a-z&&[xyz]]	a to z and (x, y or z)
[ad-z]	Performs subtraction
[a-z&&[^x-z]]	a to z and not x to z

Metacharacters Used in Regular Expression

- Metacharacters supported by Java RegEx Engine are:

Regular Expression	Description
\d	Represents a digit
\D	Represents a non-digit
\s	Represents a whitespace
\S	Represents a non-whitespace
\w	Represents a word character (letters, numbers or an underscore)
\W	Represents a non-word character
"."	Represents any character

Quantifiers

- It is used to specify the number of occurrences

X	x, once or not at all
X*	x, zero or more times
X+	x, one or more times
X{n}	x, exactly n times
X{n,}	x, at least n times
X{n,m}	x, at least n but not more than m times

Implement RegEx in Java

How can we implement RegEx in Java?

Implementing RegEx in Java

- To implement RegEx in Java, we should make use of the following classes
  - Pattern Class:
    - This class defines the pattern to be searched
  - Matcher Class:
    - This class is used to search a pattern
  - PatternSyntaxException:
    - This class indicates the syntax error in a regular expression pattern

Matcher Class Methods

Method	Description
boolean matches()	Tests whether the regular expression matches the pattern
boolean find()	Finds the next expression that matches the pattern
boolean find(int start)	Finds the next expression that matches the pattern from the given start number
replaceAll()	Replaces all the matches of the regular expression
int start()	Returns the starting index of the matched subsequence
int end()	Returns the ending index of the matched subsequence
replaceFirst()	Replaces only the first match

Pattern Class Methods

Method	Description
static Pattern compile(String regex)	Compiles the given RegEx and returns the instance of the pattern
Matcher matcher(CharSequence input)	Creates a matcher that matches the given input with the pattern
static boolean matches(String regex, CharSequence input)	Works as a combination of compile and matcher methods
String[] split(CharSequence input)	Splits the given input string around matches of a given pattern
String pattern()	Returns the RegEx pattern

Interactive Demo

Write a program to accept the phone of a user
and validate its phone number by using regular
expression pattern matching.

Task1: Phone number must be of 10 digit

Task2: It must begin with either 7,8 or 9

Interactive Demo

Write a program to accept the email ID of a user and valid it by using the regular expression pattern matching.

StringBuffer

- StringBuffer is a mutable string

- Java StringBuffer class is thread-safe, which means that multiple threads cannot access it simultaneously, so it is safe

Syntax:

```
StringBuilder var = new StringBuilder(str);
```

StringBuilder

- StringBuilder is a mutable string

- It is not thread-safe, which means that multiple threads can access it simultaneously, so it is not safe

- StringBuilder is more efficient

Syntax:

```
StringBuilder var = new StringBuilder(str);
```

String, StringBuffer and StringBuilder

Index	String	StringBuffer	StringBuilder
Storage	String pool	Heap	Heap
Modifiable	It is not modifiable because it is immutable	It is modifiable because it is mutable	It is modifiable because it is mutable
Thread Safe	It is thread-safe, multiple threads can't access it simultaneously	It is thread-safe, multiple threads can't access it simultaneously	It is not thread-safe, multiple threads can access it simultaneously
Synchronized	Its methods are synchronized	Its methods are synchronized	Its methods are not synchronized
Performance	performance is high	performance is low	performance is high
Usage	A string can be used if the contents are fixed	To use StringBuffer, we need thread safety, if the contents are not fixed	To use StringBuilder, we don't need thread safety If contents are not fixed

Key Takeaways

- Regular Expression

- Methods used for implementing matcher class and pattern class

- StringBuffer and StringBuilder

- Compare String, StringBuffer and StringBuilder

Thank you!