REGULAR EXPRESSION

MATCH IT, CATCH IT

REGULAR EXPRESSION : INTRODUCTION

Regular expressions are patterns used to match character combinations in strings

In JavaScript, regular expressions are also objects. These patterns are used with the exec() and test() methods of RegExp

In JavaScript, regular expressions are also used with the match(), matchAll(), replace(), replaceAll(), search(), and split() methods of String

Regex can be used in programming languages such as Python, SQL, Javascript, R, Google Analytics, Google Data Studio, and throughout the coding process

REGULAR EXPRESSION : BASIC MATCHERS

The character or word we want to find is written directly. It is similar to a normal search process

Text	“I have no special talents. I am only passionately curious.”
Regex	/_________/g

DOT . : ANY CHARACTER

The period . allows selecting any character, including special characters and spaces

Text	abcABC123.:!?
Regex	/__________/g
Answer	/./g

CHARACTER SETS [abc]

If one of the characters in a word can be various characters, we write it in square brackets [] with all alternative characters.
For example, to write an expression that can find all the words in the text, type the characters a, e, i, o, u adjacently within square brackets [].

Text	bar ber bir bor bur
Regex	/__________/g

NEGATED CHARACTER SETS [^abc]

To find all words in the text below, except for ber and bor, type e and o side by side after the caret ^ character inside square brackets []

Text	bar ber bir bor bur
Regex	/_________/g

LETTER RANGE[a-z]

To find the letters in the specified range, the starting letter and the ending letter are written in square brackets [] with a dash between them -. It is case-sensitive. Type the expression that will select all lowercase letters between e and o, including themselves.

Text	abcdefghijklmnopqrstuvwxyz
Regex	/_________/g

NUMBER RANGE[0-9]

To find the numbers in the specified range, the starting number and the ending number are written in square brackets [] with a dash - between them. Write an expression that will select all numbers between 3 and 6, including themselves.

Text	0123456789
Regex	/_________/g

PRACTICE EXERCISE

1	Type the expression to select individual letters, numbers, spaces, and special characters in the text. The expression you type must match any character.	azAZ09_-=!?.,:;
2	Write the phrase that matches each word in the text. The only characters that change are the initials of the words.	beer deer feer
3	Write down the expression that will match anything other than the words beor and beur in the text. Do this using the negated character set.	bear beor beer beur
4	Write the expression that will select the letters from g to k in the text. g and k letters should also be included in this range.	abcdefghijklmnopqrstuvwxyz
5	Type an expression to select numbers from 2 to 7 in the text. 2 and 7 should also be included in this range.	0123456789

REPETITIONS

Some special characters are used to specify how many times a character will be repeated in the text. These special characters are the plus +, the asterisk *, and the question mark ?

*	indicate that the character may either not match at all or can match many times
+	indicate that a character can occur one or more times
?	indicate that a character is optional

ASTERISK *

We put an asterisk * after a character to indicate that the character may either not match at all or can match many times.
For example, indicate that the letter e should never occur in the text, or it can occur once or more side by side.

Text	br ber beer
Regex	/__________/g

PLUS SIGN +

To indicate that a character can occur one or more times, we put a plus sign + after a character. For example, indicate that the letter e can occur one or more times in the text.

Text	br ber beer
Regex	/__________/g

QUESTION MARK ?

To indicate that a character is optional, we put a ? question mark after a character. For example, indicate that the following letter u is optional.

Text	color, colour
Regex	/__________/g

CURLY BRACES

To express the occurrence of a character in a certain number range, we write curly braces {x,y} with the interval we want to go to the end. For example, indicate that the following letter e can only occur between 1 and 3

Text	ber beer beeer beeeer
Regex	/__________/g

PRACTICE EXERCISE

1	Write the expression using the plus sign + to select words in which the letter e occurs one or more times in the text	azAZ09_-=!?.,:;
2	Write the expression using curly braces {} that will find texts containing 4 numbers side by side. Remember that the range [0-9] will match a single digit.	10/9/2021
3	Write the expression using curly braces {} that will find texts containing at least 1 and at most 4 numbers side by side.	10/9/2022

PARENTHESES ( ): GROUPING

We can group an expression and use these groups to reference or enforce some rules. To group an expression, we enclose () in parentheses

Text	ha-ha,haa-haa
Regex	/___________/g

PIPE CHARACTER |

It allows to specify that an expression can be in different expressions. Thus, all possible statements are written separated by the pipe sign |.
Add another pipe sign | to the end of the expression and type rat so that all words are selected

Text	cat Cat rat
Regex	/__________/g

ESCAPE CHARACTER \

There are special characters that we use when writing regex. { } [ ] / \ + * . $^ | ?
Before we can select these characters themselves, we need to use an escape character \

Text	(*) Asterisk.
Regex	/__________/g

CARET SIGN ^

Selecting by Line Start.
We were using [0-9] to find numbers. To find only numbers at the beginning of a line, prefix this expression with the ^ sign

Text	1. 3 eggs, beaten 2. 1 tsp sunflower oil 3. 1 tsp butter
Regex	/__________/g

DOLLAR SIGN $

Selecting by End of Line.
Let's use the $ sign after the html value to find the html texts only at the end of the line

Text	https://domain.com/what-is-html.html https://otherdomain.com/html-elements https://website.com/html5-features.html
Regex	/__________/g

WORD CHARACTER \w

Selection of word – Capital-case, Lower-case and underscore

Text	abcABC123 _.:!?
Regex	/__________/g

NUMBER CHARACTER \d

\d is used to find only number characters

Text	abcABC123 .:!?
Regex	/__________/g

SPACE CHARACTER \s

\s is used to find only space characters

Text	abcABC123 .:!?
Regex	/__________/g

FLAGS/MODIFIERS

Flags change the output of the expression. That's why flags are also called modifiers. Flags determine whether the typed expression treats text as separate lines, is case sensitive, or finds all matches

g	causes the expression to select all matches
m	use the multiline flag to handle each line separately
i	remove the case-sensitivity of the expression

GLOBAL FLAG

The global flag causes the expression to select all matches. If not used it will only select the first match. Now enable the global flag to be able to select all matches

Text	domain.com, test.com, site.com
Regex	/__________/g

MULTILINE FLAG

Regex sees all text as one line. But we use the multiline flag to handle each line separately. In this way, the expressions we write to identify patterns at the end of lines work separately for each line. Now enable the multiline flag to find all matches

Text	domain.com test.com site.com
Regex	/__________/m

CASE-INSENSITIVE FLAG

In order to remove the case-sensitivity of the expression we have written, we must activate the case-insensitive flag

Text	DOMAIN.COM TEST.COM SITE.COM
Regex	/__________/i

GREEDY MATCHING

Regex does a greedy match by default. This means that the matchmaking will be as long as possible. Check out the example below. It refers to any match that ends in r and can be any character preceded by it. But it does not stop at the first letter r

Text	ber beer beeer beeeer
Regex	/.*r/

LAZY MATCHING

Lazy matchmaking, unlike greedy matching, stops at the first matching. For example, in the example below, add a ? after * to find the first match that ends with the letter r and is preceded by any character. It means that this match will stop at the first letter r

Text	ber beer beeer beeeer
Regex	.*?r

REFERENCES

READING MATERIAL

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions
- https://javascript.info/regular-expressions

VIDEO MATERIAL

- https://www.youtube.com/watch?v=WeIRHETbcLs
- https://www.youtube.com/watch?v=B5iF6XBpcsI