Regular Expressions

- Regular expression starts with forward slash and ends with forward slash / /

Regular expression	Test string	Match found
/ninja/	lsnf lktf ninja	1-match
/ninja/	jktl ninjatwl	1-match
/^ninja$/	jktl ninja twl	0-match

↑ this will only match when there is nothing before it or nothing after it

example.

/^ninja$/	ninja	1-match
/^ninja$/	ninjatwluw jtl	0-match
/^[a-z]$/	p	1-match

↑ this will only match when there is single character(word) between a to z

/^[a-z]$/	hp	0-match
/^[a-zA-Z]$/	H	1-match

↑ this will only match when there is single character(word) betweeen capital A to Z or small a to z

/^[a-zA-Z]$/	HT	0-match
/^[a-zA-Z]$/	g	1-match
/^[a-zA-Z]$/	i	1-match
/^[a-zA-Z]$/	ij	0-match

/^[a-zA-Z]{6,10}$/ hellot 1-match

↑

this will only match when there is word (single word) which has characters from 6 to 10.

/^[a-zA-Z]{6,10}$/ hello 0-match

/^[a-zA-Z]{6,10}$/ hellojtolj 1-match

/^[a-zA-Z]{6,10}$/ hellohelloj 0-match

/^[a-zA-Z0-9]{6,10}$/ ninja12 1-match

↑

this will only match when there is single word which has characters from 6 to 10 and contains lowercase alphabets, uppercase alphabets and digits from 0 to 9

/^[a-zA-Z0-9]{6, 10}$/ ninja12@ 0-match

↑

due to special symbol there is no match

/^.{6,10}$/ ninjas@[! 1-match

↑

this will accept every character, letter, number, special symbol, etc. but it should be or total characters should be from 6 to 10

/^.{6,10}$/ 123456![@17 0-match

✓

Testing Regular Expression patterns

-
  ```
  const username = 'shaunp';
  const pattern = /^[a-z]{6,}$/;
  let result = pattern.test(username);
  if (result) {
      console.log('success!');
  } else {
      console.log('failure!');
  }
  ```

  → success

↓

Here /^[a-z]{6,}$/ means single word which contains lowercase alphabets from a to z and they are at least 6 i.e. {6,} this means at least 6

-
  ```
  const username = 'shaunp12';
  const pattern = /^[a-z]{6,}$/;
  let result = pattern.test(username);
  if (result) {
      console.log('success!');
  } else {
      console.log('failure!');
  }
  ```

  → failure

↓

'failure' because it contains numbers

```
- const  username = 'shaunp';
  const  patteen = /^[a-z]{6,}$/;
  let  result = username.search(pattern);
  console.log(result);                    → 0
```

↓

this will give integer as output, when it will find that particular word then it will give the position of word and if not found then -1 will be output.

```
- const  username = 'shaupp12';
  const  patteen = /^[a-z]{6,}$/;
  let result = username.search(patteen);
  console.log(result);                    → -1
```

↓

-1 because pattern didn't match due to numbers included.

✓

Basic Form validations

```
* <form class="signup-form">
      <input type="text" name="username"
                      placeholder="username" >
      <input type="submit" value="submit">
      <div class="feedback"></div>
  </form>
```

=

```
- const form = document.queryselector('.signup-form');
  const feedback = document.queryselector('.feedback');
  form.addEventListener('submit', e => {
       e.preventDefault();
       // validation
       const username = form.username.value;
       const usernamePattern = /^[a-zA-z]{6,12}$/;

       if(usernamePattern.test(username)) {
           //feedback good info
           feedback.textContent = 'valid username!';
       }else{
           // feedback help info
           feedback.textContent = 'invalid username!';
       }
  });
```

Here when user will type valid username
i.e. [single word which contains uppercase or
lowercase characters and it should between
i.e. total numbers, should be between 6 to 12
then it is valid] → of characters

and then click on submit button then if it is valid then "valid username" will come below submit button because we have use their .textContent and if it is not valid then "invalid username" will come below submit button.

Below submit button means in div which has class="feedback" (.feedback)