1. Можно использовать три вида кавычек: ‘’,””,``.  
   `` отличаются возможностями:

-можно разбить строку внутри , например, не `a,b,c`, a `a,

b,

c`

-удобнее включать переменную, т.е. возможность написать `blabla ${model\_1} blabla}

-внутри строки в обратных кавычках можно обернуть нужное слово в прямые, и ничего не сломается

2. найти подстроку в строке можно:

-методом includes(), чувствителен к регистру

-методом indexOf() находит индекс первого вхождения

- методом LastIndexOf() находит индекс последнего вхождения

- Search() выполняет поиск строки для заданного значения, возвращая позицию совпадения

3. \n – символ перехода на новую строку (по аналогии с Enter в текстовых редакторах)

4. let name = “nastya”

const name\_1 = () => {

let name\_1 = name[0].toUpperCase() + name.slice(1);

console.log(name\_1);

}

5.

let date = new Date(2021, 0, 24, 22, 51, 0, 0);

alert( date );

6.

function getSecondsToTomorrow() {

let now = new Date();

// завтрашняя дата

let tomorrow = new Date(now.getFullYear(), now.getMonth(), now.getDate()+1);

let diff = tomorrow - now; // разница в миллисекундах

return Math.round(diff / 1000); // преобразуем в секунды

}

alert(getSecondsToTomorrow());

7.str.replace(/[^0-9]/g,"")

8. join()-склеить из массива строку

Split() – преобразовать строку в массив (разделяем на отдельные элементы)

9.math.round(parameter)

10. Math.ceil(Math.random()\*100)

11. str.trim() удаляет из строки пробелы

12. Флаг – пeременная, которая влияет на поиск, особенно удобна в регулярных выражениях. Например, с помощью флага i можно производить поиск без учета регистра, с помощью флага g можно выбрать вообще все выражения, подходящие под условия поиска (без него выберется только первое)