Exercise on String and Template Literals

Practice your string and template literals by printing an array in a tabular format, manipulating strings, printing emojis, and playing with if-else clauses.

Exercise 1:

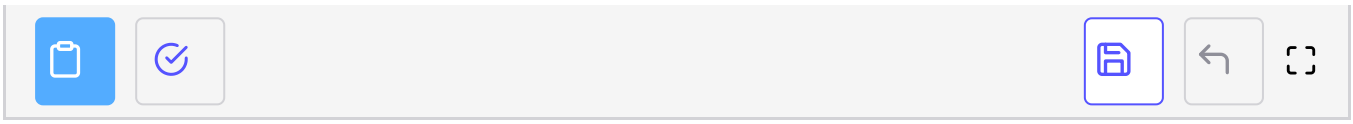
Use template literals to create a table that prints the following array in a tabular format.

```
let departures = [
    {
        id                    : 'KL 1255',
        destination           : 'Amsterdam',
        departureTime         : '21:55',
        gate                  : 'A13',
    },
    {
        id                    : 'OK 001',
        destination           : 'Prague',
        departureTime         : '20:40',
        gate                  : 'A13',
        status                : 'Check-in'
    },
    {
        id                    : '4U 2011',
        destination           : 'Stuttgart',
        departureTime         : '20:35',
        gate                  : 'A11',
        status                : 'Check-in'
    },
    {
        id                    : 'LX 911',
        destination           : 'Zurich',
        departureTime         : '20:15',
        expectedDepartureTime : '21:15',
        status                : 'check-in'
    },
    {
        id                    : 'OS 133',
        destination           : 'Vienna',
        departureTime         : '19:25',
        gate                  : 'A06',
        status                : 'Departed'
    }
];

let headers = {
    id                    : 'Id',
```

```
    destination           : 'Destination',
    departureTime         : 'DepartureTime',
    expectedDepartureTime : 'Expected Departure Time',
    gate                  : 'Gate',
    status                : 'Status'
}

//Write your code here...
```



- We will build our solution brick by brick and start with the table header.
- We will then create a template literal for one table row. For the sake of simplicity, we will use departures[0] as sample data.
- We already know how to insert one row into a template. We will then insert all rows.

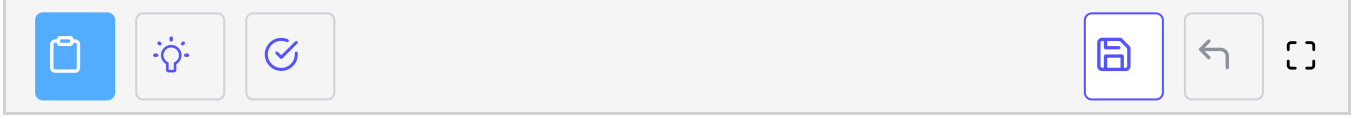
In real life, you might want to research escaping the result, and automatizing some parts of the template building process. By substituting the variables recursively in the template literal, we get the literal as given in the solution.

Exercise 2:

Create a tag function that transforms all of its String substitutions into upper case.

```
let upper = (textArray, ...substitutions) => {
        //Write your code here...
};

let a = 1, b = 'ab', c = 'DeF';
console.log(upper`x ${a} x ${b} x ${c} x`);
```



Follow the tag function syntax from the theory part. We have a textArray of length `substitutions.length - 1`. All we need to do is connect the text and substitution segments, and transform each substitution.

Exercise 3:

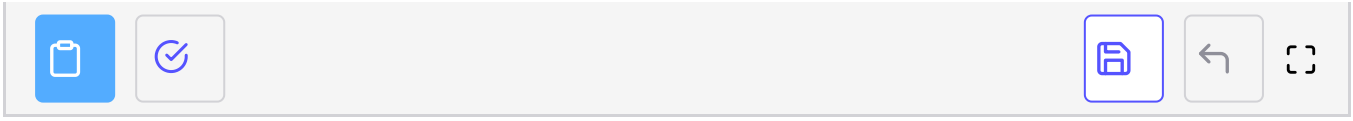
Do you remember Exercise 2 of the section on “for-of loops”? It is now time to

create a template literal that prints out all the emojis in a tabular format. The rows of the table should be the last character in the hexadecimal code of the emoji (`0`-`F`), while the columns should be the fourth character of the emoji (`0`-`4`). Make sure to print out the table to the console.

Reference: http://apps.timwhitlock.info/emoji/tables/unicode

```
let prefix = '1F6';
let digits4 = '01234';
let digits5 = '012345678 90ABCDEF';

//Write your code here...
```



Define the header first. Notice the spread operator to handle the digits4 string as an array of characters.

Building the rows should be straightforward based on the first exercise.

- We used String.fromCodePoint(hexString) to convert a string of a hexadecimal number to a Unicode character. “0x” denotes that the integer parser should parse a hexadecimal number
- Read the code starting in the rightmost indentation level, and proceed leftwards
- The map function of [...digits4] prepares the cells of each row
- The map function of [...digits5] prepares the five rows of the table

Changing the granularity level of the template for debugging purposes is advised.

If you paste all the code in the Chrome developer tools, and then execute

```
document.body.innerHTML = header
```

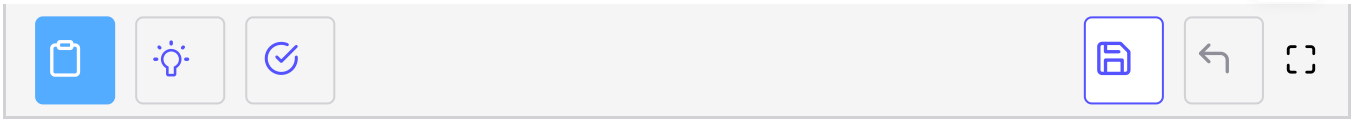
you can see the table of emojis appear.

Exercise 4:

Find a way to create the following `if-else` clause in a template literal.

```
if ( 10 % 2 == 0 ) {
console.log("Even");
} else {
console.log("Odd");
}
```

```
// Write your code here...
```



First of all, note that you don’t have to include conditions, there is always a way to describe the template in more steps.

If-else clauses are used mostly when we want to conditionally hide a template segment, or we want to do type checks, substitute default values, and avoid reference errors. In both cases, the if-else clause should determine if we print out a value or not:

```
if ( condition ) {
// value1
} else {
// value2
}
```

We can simply use the ternary operator to return the proper value.