Recursion II

Overview

```
/*
  - Nested data structures and recursion
  - Nested arrays
  - Nested objects
*/
```

Nested data structures and recursion

```
1   /* JS can have deeply-nested data structures  */
2
3   let crazyArray = [1, [2, [3, [4, [5, [6, [7, [8, [9]]]]]]]]];
4
5   /* are you going to write 9 nested for loops to iterate through the crazy
6      array?? no way! */
7
8   /* recursion can be a great tool to iterate through a nested data
9      structure, especially if you don't know how many levels of nesting
10     will be in the array or object! */
11
12
13
14
```

Example: logsAnArray

```
1   /* when writing recursive functions that will handle a nested data
2      structure, it's good to start by making sure it works with a flat
3      data structure. think of this as the base case!
4
5   function logsAnArray(array) {
6     for (let i = 0; i < array.length; i++) {
7       let element = array[i];
8       console.log(element);
9     }
10  }
11
12  logsAnArray([1, 2, 3]);
13
14
```

1
[2, 3]

Example: logsAnArray

```
1   function logsAnArray(array) {
2     for (let i = 0; i < array.length; i++) {
3       let element = array[i];
4       console.log(element);
5     }
6   }
7
8   logsAnArray([1, [2, 3]]); // what if the input is nested?
9   /* consider element on line 3. that could be a number, or it could be
10    another array! */
11
12  /* if element is an array, we want to log every number inside of it. if
13    only we had a function that takes an array and logs out all of its
14    values! oh wait… */
```

Example: logsAnArray

```
1   function logsAnArray(array) {
2     for (let i = 0; i < array.length; i++) {
3       let element = array[i];
4
5       if (Array.isArray(element)) {
6         logsAnArray(element);
7       }
8       else {
9         console.log(element);
10      }
11    }
12  }
13
14  logsAnArray([1, [2, 3]]);
```

Callstack

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       finalString += element;
6     }
7     return finalString;
8   }
9
10  let result = concatEls(['a', 'b', 'c']);
11  console.log(result);
12
13
14
```

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['a', ['b', 'c']])	''

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['a', ['b', 'c']])	'' += 'a'

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['a', ['b', 'c']])	'a'

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['a', ['b', 'c']])	'a' += concatEls(['b, c'])

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['b', 'c'])	''
concatEls(['a', ['b', 'c']])	'a' += concatEls(['b, c'])

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['b', 'c'])	'' += 'b'
concatEls(['a', ['b', 'c']])	'a' += concatEls(['b, c'])

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['b', 'c'])	'b'
concatEls(['a', ['b', 'c']])	'a' += concatEls(['b, c'])

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['b', 'c'])	'b' += 'c'
concatEls(['a', ['b', 'c']])	'a' += concatEls(['b, c'])

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['b', 'c'])	'bc'
concatEls(['a', ['b', 'c']])	'a' += concatEls(['b, c'])

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['b', 'c'])	=> 'bc'
concatEls(['a', ['b', 'c']])	'a' += concatEls(['b, c'])

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['a', ['b', 'c']])	'a' += 'bc'

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['a', ['b', 'c']])	'abc'

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString
concatEls(['a', ['b', 'c']])	=> 'abc'

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Callstack	finalString

Example: concatEls

```
1   function concatEls(array) {
2     let finalString = '';
3     for (let i = 0; i < array.length; i++) {
4       let element = array[i];
5       if (Array.isArray(element)) {
6         finalString += concatEls(element);
7       } else {
8         finalString += element;
9       }
10    }
11    return finalString;
12  }
13  let result = concatEls(['a', ['b', 'c']]);
14  console.log(result);
```

Nested objects

```
/* working with nested objects is very similar to working with nested
   arrays */

/* use a for-in loop to iterate through the object */

/* if the value is another object, recursively call the function on the
   nested object */
```

Example: sumVals

```
1   /* write a function sumVals that sums the values in an object */
2
3   function sumVals(obj) {
4     let sum = 0;
5     for (let key in obj) {
6       let value = obj[key];
7       sum += value;
8     }
9     return sum;
10  }
11
12  let result = sumVals({a: 10, b: 20});
13  console.log(result);
14
```

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}})
    console.log(result);
```

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	0

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	0 += 1

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({c: {d: {e:2, f: 3}}})	0
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({d: {e:2, f: 3}})	0
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({d: {e:2, f: 3}})	0 += sumVals({e:2, f: 3})
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({e:2, f: 3})	0
sumVals({d: {e:2, f: 3}})	0 += sumVals({e:2, f: 3})
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({e:2, f: 3})	0 += 2
sumVals({d: {e:2, f: 3}})	0 += sumVals({e:2, f: 3})
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({e:2, f: 3})	2
sumVals({d: {e:2, f: 3}})	0 += sumVals({e:2, f: 3})
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({e:2, f: 3})	2 += 3
sumVals({d: {e:2, f: 3}})	0 += sumVals({e:2, f: 3})
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({e:2, f: 3})	5
sumVals({d: {e:2, f: 3}})	0 += sumVals({e:2, f: 3})
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({e:2, f: 3})	=> 5
sumVals({d: {e:2, f: 3}})	0 += sumVals({e:2, f: 3})
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({d: {e:2, f: 3}})	0 += 5
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({d: {e:2, f: 3}})	5
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({d: {e:2, f: 3}})	=> 5
sumVals({c: {d: {e:2, f: 3}}})	0 += sumVals({d: {e:2, f: 3}})
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({c: {d: {e:2, f: 3}}})	0 += 5
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({c: {d: {e:2, f: 3}}})	5
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({c: {d: {e:2, f: 3}}})	=> 5
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += sumVals({c: {d: {e:2, f: 3}}})

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	1 += 5

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	6

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum
sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}}	=> 6

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b:
    console.log(result);
```

Callstack	sum

Example: sumVals

```
1   function sumVals(obj) {
2     let sum = 0;
3     for (let key in obj) {
4       let value = obj[key];
5       if (typeof value === 'object') {
6         sum += sumVals(value)
7       } else {
8         sum += value;
9       }
10    }
11    return sum;
12  }
13
14  let result = sumVals({a: 1, b: {c: {d: {e:2, f: 3}}}})
    console.log(result);
```

Recap

```
/*
  - Nested data structures and recursion
  - Nested arrays
  - Nested objects
*/
```