Övningar (funktioner)

Agenda

Repetition

Bygga program med funktioner

Övningar och repetition

Exercise 02 - repeatString

Write a function that simply repeats the string a given number of times:

```
repeatString('hey', 3) // returns 'heyheyhey'
```

This function will take two arguments, `string` and `num`.

Hints

- Take note of the above function call- how exactly is it being called?
- You're going to want to use a loop for this one.
- Create a variable to hold the string you're going to return, create a loop that repeats the given number of times and add the given string to the result on each loop.

Övningar och repetition

Exercise 03 - Reverse a String

Pretty simple, write a function called `reverseString` that returns its input, reversed!

```
reverseString('hello there') // returns 'ereht olleh'
```

Hints

Strings in JavaScript cannot be reversed directly so you're going to have to split it into something else first.. do the reversal and then join it back together into a string.

Övningar och repetition

Exercise 04 - removeFromArray

Implement a function that takes an array and some other arguments then removes the other arguments from that array:

```
removeFromArray([1, 2, 3, 4], 3); // should remove 3 and return [1,2,4]
```

Hints

The first test on this one is fairly easy, but there are a few things to think about(or google) here for the later tests:

- how to remove a single element from an array
- how to deal with multiple optional arguments in a javascript function

Övningar och repetition

Exercise 11 - Get the Titles!

You are given an array of objects that represent books with an author and a title that looks like this:

```
const books = [
  {
    title: 'Book',
    author: 'Name'
  },
  {
    title: 'Book2',
    author: 'Name2'
  }
]
```

Your job is to write a function that takes the array and returns an array of titles:

```
getTheTitles(books) // ['Book','Book2']
```

Övningar och repetition

Exercise 12 - Find the Oldest

Given an array of objects representing people with a birth and death year, return the oldest person.

Hints

- You should return the whole person object, but the tests mostly just check to make sure the name is correct.
- This can be done with a couple of chained array methods, or by using `reduce`.
- One of the tests checks for people with no death-date.. use JavaScript's Date function to get their age as of today.

Övningar och repetition

Exercise XX - snakeCase

Convert phrases and words into snake case

> Snake case (or snake_case) is the practice of writing compound words or phrases in which the elements are separated with one underscore character (_) and no spaces, with each element's initial letter usually lowercased as in "foo_bar"

```
snakeCase('Hello, World!') // hello_world
```

Övningar och repetition

Given three arguments — an object `obj` of the stolen items, the pet's `name` and a `value` — return an object with that name and value in it (as key-value pairs).

Examples

```
addName({}, "Brutus", 300) → { Brutus: 300 }

addName({ piano: 500 }, "Brutus", 400) → { piano: 500, Brutus: 400 }
```

Övningar och repetition

Create a function that determines whether a shopping order is eligible for free shipping. An order is eligible for free shipping if the total cost of items purchased exceeds $50.00.

Examples

```
freeShipping({ "Shampoo": 5.99, "Rubber Ducks": 15.99 }) → false

freeShipping({ "Flatscreen TV": 399.99 }) → true
```

Övningar och repetition

Create a function that takes an object and returns the keys and values as separate arrays. Return the keys sorted alphabetically, and their corresponding values in the same order.

Examples

```
keysAndValues({ a: 1, b: 2, c: 3 })
→ [["a", "b", "c"], [1, 2, 3]]

keysAndValues({ a: "Apple", b: "Microsoft", c: "Google" })
→ [["a", "b", "c"], ["Apple", "Microsoft", "Google"]]

keysAndValues({ key1: true, key2: false, key3: undefined })
→ [["key1", "key2", "key3"], [true, false, undefined]]
```

Övningar och repetition

Try finding your ancestors and offspring with code.

Create a function that takes a number `x` and a character `y` (`"m"` for male, `"f"` for female), and returns the **name of an ancestor (m/f)** or **descendant (m/f)**.

- If the number is **negative**, return the **related ancestor**.
- If **positive**, return the **related descendant**.
- You are generation `0`. In the case of `0` (male or female), return `"me!"`.

Examples

```
generation(2, "f") → "granddaughter"

generation(-3, "m") → "great grandfather"

generation(1, "f") → "daughter"
```

Generation	Male	Female
-3	great grandfather	great grandmother
-2	grandfather	grandmother
-1	father	mother
0	me!	me!
1	son	daughter
2	grandson	granddaughter
3	great grandson	great granddaughter

Övningar och repetition

Create the function that takes an array with objects and returns the sum of people's budgets.

Examples

```
getBudgets([
  { name: "John", age: 21, budget: 23000 },
  { name: "Steve",  age: 32, budget: 40000 },
  { name: "Martin",  age: 16, budget: 2700 }
]) → 65700

getBudgets([
  { name: "John",  age: 21, budget: 29000 },
  { name: "Steve",  age: 32, budget: 32000 },
  { name: "Martin",  age: 16, budget: 1600 }
]) → 62600
```

Övningar och repetition

You call your spouse to inform his/her most precious item is gone! Given an object of stolen items, return the most expensive item on the list.

Examples

```
mostExpensiveItem({
  piano: 2000,
}) → "piano"

mostExpensiveItem({
  tv: 30,
  skate: 20,
}) → "tv"

mostExpensiveItem({
  tv: 30,
  skate: 20,
  stereo: 50,
}) → "stereo"
```