4. JavaScript

4.1 Tooling

If you had any problems with this, we can look at them during one of the evening sessions!

4.2 Syntax

4.2.1 Data Types

JavaScript

```
console.log(
            (typeof undefined)
  + ", " + (typeof true)
  + ", " + (typeof "foo")
  + ", " + (typeof 42)
  + ", " + (typeof 12345678901234567890n)
  + ", " + (typeof { a: true })
  + ", " + (typeof null)
  + ", " + (typeof Symbol("foo"))
)
```

> Did you notice that the typoe of `null` is stated to be `object`? By many, that's considered a "bug". They tried to change it, but had to roll back (see here [a] if you're *really* interested).

[a] http://wiki.ecmascript.org/doku.php?id=harmony:typeof_null

4.2.2 Data Structures

4.2.2.1 Constants and Variables

JavaScript

```
const a = "it's";
let b = "hi! ";
let c = 37;
const d = 42;
b = a; // ok
// a = b; // fails, saying "TypeError: Assignment to constant variable"
c = d; // ok
d = c; // fails, saying "TypeError: Assignment to constant variable"
b = d; // ok, b now holds a number
c = a; // ok, c now holds a string
```

4.2.2.2 Objects

JSON

```
{
  "forenames": ["Christoph", "Emanuel"],
  "knownas": "Chris",
  "surname": "Zwicker",
  "dob": "1981-03-15T00:00:00.000-00:01",
  "family": [
    {
      "knownas": "Michelle"
    },
    {
      "knownas": "Philippe"
    },
    {
      "knownas": "Alex"
    }
  ]
}
```

Here's a possible good find about JSON dates on StackOverflow [1]

4.2.2.3 Arrays & Maps

JavaScript

```
const urls = ["www.stackoverflow.com", "www.yahoo.com", "www.twitter.com"];
const passwords = ["wer92klc;.asdf", "934ldfg92;/sk34"];
const pw4url = { 0: 1, 1: 0, 2: 1};
```

[1] https://stackoverflow.com/questions/10286204/what-is-the-right-json-date-format

4.2.3 Control Structures

4.2.3.1 Conditions

What was missing in the exercise description? If `x` or `y` are **exactly** 10, and the other variable is greater than 5, there was no execution path to take. We've solved this here by defining that case to be a "fit".

JavaScript

```
let x = 5;
let y = 9;
if (x > 10 || y > 10) {
  console.log("out of bounds");
} else {
  if (x * y < 50) {
    console.log("uncomfortable");
  } else {
    console.log("fits!");
  }
}
```

4.2.3.2 Case Matching

JavaScript

```
let x = undefined;
// let x = true;
// let x = "da, da, da";
// let x = 99;
// let x = 23489123982489123n;
// let x = { a: 9, b: "out of ten"};
// let x = null;
// let x = Symbol("foo");

if (typeof x === "symbol") {
  console.log("symbol: " + String(x) + "; how symbolic ;-)");
} else {
  // toString has to be called explicitly: symbol doesn't allow implicit conversion
  let result = (typeof x) + ": " + x + "; ";
  switch(typeof x) {
    case "undefined":
      console.log("undefined; what's that supposed to mean?");
      break;
    case "boolean":
      console.log(result + "always be true to yourself");
      break;
    case "string":
      console.log(result + "just stringing along?");
      break;
    case "number":
      console.log(result + "is there any but 42?");
      break;
    case "bigint":
      console.log(result + "now you're exaggerating!");
      break;
    case "object":
      if (x === null) {
        console.log("null; or did you think it was 'object'?");
      } else {
        console.log("object: " + JSON.stringify(x) + "; now that's impressive..");
      }
      break;
  }
}
```

4.2.3.3 Loops

JavaScript

```
const input = [6,1,29,-4,4,13,0];
let step = 1;
let sorted;
let temp;
console.log("Initial: " + JSON.stringify(input));
do {
  sorted = true;
  for(let i = 0; i < input.length - 1; i++) {
    if (input[i] > input[i + 1]) {
      sorted = false;
      temp = input[i];
      input[i] = input[i + 1];
      input[i + 1] = temp;
    }
  }
  if(!sorted) {
    console.log("Step " + step++ + ": " + JSON.stringify(input));
  }
} while (!sorted);
```

4.2.4 Program Structures

4.2.4.1 Methods

JavaScript

```
const pointInsideBox = (point, box) => {
  const point1 = box[0];
  const point2 = box[1];

  return point.x > Math.min(point1.x, point2.x)
    &&   point.x < Math.max(point1.x, point2.x)
    &&   point.y > Math.min(point1.y, point2.y)
    &&   point.y < Math.max(point1.y, point2.y)
    &&   point.z > Math.min(point1.z, point2.z)
    &&   point.z < Math.max(point1.z, point2.z)
  ;
}
```

- Search for e.g. `[javascript] "arrow function" return object` and look at e.g. this [2] answer.
- Searching for e.g. `[javascript] property same name shorthand` leads to e.g. this [3], which contains the answer to our question in the question asked.

[2] https://stackoverflow.com/a/28770578/7159043
[3] https://stackoverflow.com/questions/50179669/shorthand-for-arrow-functions-for-object-property-names

4.3 Language Features

4.3.1 Template Strings & Spread Operator

JavaScript

```javascript
const me = JSON.parse(`
  {
    "forenames": ["Christoph", "Emanuel"],
    "knownas": "Chris",
    "surname": "Zwicker",
    "dob": "1981-03-15T00:00:00.000-00:01",
    "family": [
      {
        "knownas": "Michelle"
      },
      {
        "knownas": "Philippe"
      },
      {
        "knownas": "Alex"
      }
    ]
  }`
)
const meNow = {
  currentActivity: "Teaching JS",
  currentStateOfMind: "concentrated"
};
console.log(JSON.stringify({ ...me, ...meNow }));
```

4.3.2 Variable as Property Key

JavaScript

```javascript
const getProperty = (key, person) => {
  if (
    ![
      "forenames",
      "knownas",
      "surname",
      "dob",
      "family",
      "currentActivity",
      "currentStateOfMind"
    ].includes(key)
  ) {
    return `Unknown property: ${key}`;
  }
  if (person[key] === undefined || person[key] === null) {
    return `Missing property: ${key}`;
  }
  return person[key];
}
console.log(getProperty("wolfpack", {}));
console.log(getProperty("currentStateOfMind", {}));
console.log(getProperty("knownas", { knownas: "Chris"}));
```

4.4 Algorithms

4.4.1 Searching

JavaScript

```
1 const retainEven = (input) => input.filter(e => e % 2 === 0);
2 console.log(retainEven([1,2,5,17,30]));
```

4.4.2 Sorting

JavaScript

```
1 const sortAlpha = (input) => input.sort();
2 const sortNumeric = (input) => input.sort((a,b) => a - b);
3 console.log(sortAlpha([1,5,30,17,2]));
4 console.log(sortNumeric([1,5,30,17,2]));
```

4.4.3 Mapping

JavaScript

```
1 const toCoordinate = (input) => input.map(e => ({ x: e[0], y: e[1], z: e[2] }));
2 console.log(toCoordinate([[1,3,17], [-2, 1, 6], [100, -27, -8]]));
```

4.4.4 Aggregating

JavaScript

```
1 function objectify(names) {
2   return names.reduce((previous, current) => ({ ...previous, [current]:
      current.length}), {});
3 }
4
5 console.log(objectify(["an", "apple", "must", "be", "red"]));
6 // => { "an": 2, "apple": 5, "must": 4, "be": 2, "red": 3 }
```