VERERBUNG
CONSTRUCTOR AND PROTOTYPES

Constructors können Werte von anderen Constuctors übernehmen, sie werden dazu mit .call() aufgerufen.

```
function PersonConstructor(vorname, nachname, alter, job) {
this.vorname = vorname;
this.nachname = nachname;
this.alter = alter;
this.job = job;
}

let holger = new PersonConstructor(`Holger`, `Ebrez`, 40, `arbeitssuchend`);
```

// Das Objekt holger wird hier mit Hilfe des Constructors PersonConstructor erschaffen, es nutzt dazu die angebenen Parameter.

```
console.log(holger);
```
// Gibt auf der Console aus:

```
PersonConstructor {vorname: "Holger", nachname: "Ebrez", alter:
40, job: "arbeitssuchend"}
```

```
1.alter: 50
2.job: "arbeitssuchend"
3.nachname: "Ebrez"
4.vorname: "Holger"
```

```
function StudentConstructor(vorname, nachname, alter, fach) {
PersonConstructor.call(this, vorname, nachname, alter, `Student`);
this.fach = fach;
}

let patrick = new StudentConstructor(`Patrick`, `Schwarz`, 49, `Javascript`);
```

// Das Objekt patrick wird hier mit Hilfe des Constructors StudentConstructor erschaffen, es nutzt dazu die angebenenen Parameter und den Constructor PersonConstructor.
Dem key “job“ wird dabei immer der Wert “Student“ übergeben.

```
console.log(patrick);
```
// Gibt auf der Console aus:

```
StudentConstructor {vorname: "Patrick", nachname: "Schwarz",
alter: 49, job: "Student", fach: "Javascript"}
```

```
1.alter: 49
2.fach: "Javascript"
3.job: "Student"
4.nachname: "Schwarz"
5.vorname: "Patrick"
```

Verknüpfung von Prototype-Ketten

Constructors können Methoden als Prototypes haben. Diese werden mit folgendem Syntax verknüpft:

ContructorName.prototype.methodenname = function (parameter) {code}

```
function Mieter(vorname, nachname, alter, solvent) {
	this.vorname = vorname;
	this.nachname = nachname;
	this.alter = alter;
	this.saldo = 0;
}

Mieter.prototype.zahlungsgeingang = function (mietzahlung) {
	this.saldo += mietzahlung;
}

Mieter.prototype.kontobelastung = function (faellligeMiete) {
	this.saldo -= faellligeMiete;
}
```

// Hier werden dem Constructor Mieter die Methoden zahlungseingang und kontobelastung als prototypes zugewiesen

```
function WohnungsMieter(vorname, nachname, alter, wohnungsNummer) {
Mieter.call(this, vorname, nachname, alter);
this.wohnungsNummer = wohnungsNummer;
}

function ParkplatzMieter(vorname, nachname,alter, parkplatzNummer) {
Mieter.call(this, vorname,nachname, alter);
this.parkplatzNummer = parkplatzNummer;
this.schrankenSchluesselErhalten = true;
}
```

// Hier werden die Constructors Wohnungsmieter und Parkplatzmieter erstellt, sie greifen per .call() auf den Constructor Mieter zurück, jedoch nicht auf seine Prototypes!
Würden wir jetzt den nachfolgenden Code ausführen, gäbe es auf der Console eine Fehlermeldung:

```
let mieterNr001 = new WohnungsMieter(`Marianne`, `Musterfrau`, 33, `P 1 - 33`);

mieterNr001.kontobelastung(40);

// Gibt auf der Console aus:

Uncaught TypeError: mieterNr001.kontobelastung is not a function
```

Damit Objekte, welche mit den Contructors Wohnungmieter und Parkplatzmieter erstellt wurden, auf die Prototypes des Constructors Mieter zugreifen kann, müssen die Prototype-Ketten (prototype-chains) miteinander verknüpft werden.

Dies geschieht mit folgendem Syntax:

NeuConstructorName.prototype = Object.create(ConstructorName);

In diesem Beispiel also:

```
WohnungsMieter.prototype = Object.create(Mieter.prototype);
ParkplatzMieter.prototype = Object.create(Mieter.prototype);
```

Jetzt können Objekte, welche mit den Contructors WohnungsMieter oder Parkplatzmieter erschaffen wurden, auf die Prototypes (in diesem Fall die Methoden kontobelastung() und zahlungseingang())vom Constructor Mieter zurückgreifen

```
let mieterNr002 = new ParkplatzMieter(`Marianne`, `Musterfrau`, 28, `P 1 - 33`);

mieterNr002.kontobelastung(40);

console.log(mieterNr002.saldo);
// Gibt auf der Console aus:
-40
```

```
console.log(mieterNr002);
```
// Gibt auf der Console aus:

ParkplatzMieter {vorname: "Marianne", nachname: "Musterfrau", alter: 28, saldo: -40, parkplatzNummer: "P 1 – 33", …}

```
1.alter: 28
2.nachname: "Musterfrau"
3.parkplatzNummer: "P 1 - 33"
4.saldo: -40
5.schrankenSchluesselErhalten: true
6.vorname: "Marianne"
```