Exercise 1:

Suppose an array of `firstName`, `email`, `basketValue` triples are given. Create ONE JavaScript expression that puts a default value of `'-'` and `0` to the `firstName` or `basketValue` fields respectively, whenever the `firstName` or the `basketValue` keys are missing.

Example:

```
let baskets = [
{ firstName: 'Andrew', basketValue: 55 },
{ firstName: 'Andrew' },
{ basketValue: 55},
{}
]

newBaskets : [
{ firstName: 'Andrew', basketValue: 55 },
{ firstName: 'Andrew', basketValue: 0 },
{ firstName: '-', basketValue: 55},
{ firstName: '-', basketValue: 0}
]
```

Exercise 2:

Create a object (with name `basketProto`) with the following methods:

- `addToBasket( value )` adds `value` to the basket value,
- `clearBasket()` sets the basket value to 0
- `getBasketValue()` returns the basket value
- `pay()` logs the message `{getBasketValue()} has been paid`, where `{getBasketValue()}` is the return value of the method with the same name. We can pay for the same basket as many times as we'd like.

Exercise 3:

Create an object `myBasket`, and set its prototype to the object created in Exercise 2. Create an array field in `myBasket`, containing all the items that you purchase in the following format:

```
{ itemName: 'string', itemPrice: 9.99 }
```

Redefine the `addToBasket` method such that it accepts an `itemName` and an `itemPrice`. Call the `addToBasket` method in the prototype for the price administration, and store the `itemName`-`itemPrice` data locally in your array. Make sure you modify the `clearBasket` method accordingly.

Exercise 4:

Extend your solution in Exercise 3 by adding a `removeFromBasket( index )` method. The parameter `index` should be the index of the element in the array that you would like to remove.