JS Advanced Final Exam

Problem 2. Restaurant

```
class Restaurant {
    //TODO: implement this class
}
```

Write a **class Restaurant** which has the following **functionality**:

Constructor

Should have 4 properties:

- **budgetMoney - number**
- **menu - object**
- **stockProducts - object**
- **history - array**

At initialization of the **Restaurant class**, the **constructor** accepts only the **budget!** The rest of the properties must be **empty**!

Methods

loadProducts()

Accept 1 argument **products** (**array from strings**).

- **Every element** into this array is information about product in format:

  "{productName} {productQuantity} {productTotalPrice}"

- They are separated by a **single space**

  Example: **["Banana 10 5", "Strawberries 50 30", "Honey 5 50"…]**

This method **appends products** into our products in stock (**stockProducts**) under the following circumstances:

- **If the budget allows us to buy the current product ({productTotalPrice} <= budget)** , we add it to **stockProducts** keeping **the name** and **quantity** of **the meal** and we **deduct the price of the product** from **our budget.** If the current product already exists into **stockProducts** just add the new quantity to the old one
- And finally, **whether or not** we have **added** a product to stock or **not**, we **record** our **action** in the **history**:
  - If we **were able to add** the current product:

    "Successfully loaded {productQuantity} {productName}"

- If we **not**:

"There was not enough money to load {productQuantity} {productName}"

This method must **return all actions joined by a new line!**

addToMenu()

- Accept 3 arguments **meal** (string)**, needed products** (array from strings) and **price** (number).
- Every element into **needed products** is in format: **"{productName} {productQuantity}"** They are separated by a **single space**!
- If the meal is not in our menu, **appends a meal** into object **menu**. **Must have properties products and price!**
- Check how many meals have in menu and **returns one of** the following messages:
  - One meal:

"Great idea! Now with the {meal} we have 1 meal in the menu, other ideas?"

  - Zero, Two or more meal:

"Great idea! Now with the {meal} we have {the number of all meals in the menu} meals in the menu, other ideas?"

- Otherwise, if we already have this meal return the **message**:

"The {meal} is already in the our menu, try something different."

showTheMenu()

- This method just **return all meals** from our **menu separated by a new line** in format:

```
{meal} - $ {meal price}
{meal} - $ {meal price}
{meal} - $ {meal price}
…
```

- If our menu **is empty**, just return the **message**:

"Our menu is not ready yet, please come later..."

makeTheOrder()

Accept 1 argument **meal** (string).

- This method **searches the menu** for a **certain meal**.
- If **we do not have** the **given meal**, **return** the following **message**:

"There is not {meal} yet in our menu, do you want to order something else?"

- **Otherwise,** if we **have this meal** in **the menu**, we need to check if we have the **needed products** to make it!

  If we **do not have all needed products** for this meal, **return** the following **message**:

"For the time being, we cannot complete your order ({meal}), we are very sorry..."

- If we **have this meal in the menu** and also, we **have all needed products** to make it, **return** the following message:

"Your order ({meal}) will be completed in the next 30 minutes and will cost you {the current price of the meal}."

- You also **need to reduce quantity of all used products** from those in stock and **add the price** of the meal to the **total budget**.

Examples

Input 1
`let kitchen = new Restaurant(1000);` `console.log(kitchen.loadProducts(['Banana 10 5', 'Banana 20 10', 'Strawberries 50 30', 'Yogurt 10 10', 'Yogurt 500 1500', 'Honey 5 50']));`

Output 1

`Successfully loaded 10 Banana`
`Successfully loaded 20 Banana`
`Successfully loaded 50 Strawberries`
`Successfully loaded 10 Yogurt`
`There was not enough money to load 500 Yogurt`
`Successfully loaded 5 Honey`

Input 2

`let kitchen = new Restaurant(1000);`
`console.log(kitchen.addToMenu('frozenYogurt', ['Yogurt 1', 'Honey 1', 'Banana 1', 'Strawberries 10'], 9.99));`
`console.log(kitchen.addToMenu('Pizza', ['Flour 0.5', 'Oil 0.2', 'Yeast 0.5', 'Salt 0.1', 'Sugar 0.1', 'Tomato sauce 0.5', 'Pepperoni 1', 'Cheese 1.5'], 15.55));`

Output 2
`Great idea! Now with the frozenYogurt we have 1 meal in the menu, other ideas?`

```
Great idea! Now with the Pizza we have 2 meals in the menu, other
ideas?
```

Input 3

```
let kitchen = new Restaurant(1000);
console.log(kitchen.showTheMenu());
```

Output 3

```
frozenYogurt - $ 9.99
Pizza - $ 15.55
```

Input 4

```
let kitchen = new Restaurant(1000);
kitchen.loadProducts(['Yogurt 30 3', 'Honey 50 4', 'Strawberries 20
10', 'Banana 5 1']);
kitchen.addToMenu('frozenYogurt', ['Yogurt 1', 'Honey 1', 'Banana 1',
'Strawberries 10'], 9.99);
console.log(kitchen.makeTheOrder('frozenYogurt'));
```

Output 4

```
Your order (frozenYogurt) will be completed in the next 30 minutes and
will cost you 9.99.
```

Submission

Submit only the **Restaurant class**.