JS Advanced: Regular Exam

Problem 3. Computer

```
class Computer {
    // TODO: implement this class...
}
```

Your Task

Write a **Computer class** which supports the described functionality below.

Functionality

constructor()

Receives **3** parameters at initialization of the class (**`ramMemory, cpuGHz, hddMemory`**), where each of them is a **number.**

Should have these **5** properties:

- **`ramMemory`** - **number** (should be the same as the received **ramMemory**)
- **`cpuGHz`** - **number** (should be the same as the received **cpuGHz**)
- **`hddMemory`** - **number** (should be the same as the received **hddMemory**)
- **`taskManager`** – **empty array**
- **`installedPrograms`** - **empty array**

installAProgram({name}, {requiredSpace})

This **function** should **install a new program** on the computer and **save** it in the **`installedPrograms`** **property**.

- If the **total `hddMemory`** is **exceeded** while trying to install a new program, **new error** should be **thrown** with the following message:
  "**`There is not enough space on the hard drive`**"
- If there is available space to install the given program, a **new object** with the given **name** and **`requiredSpace`** should be created and stored to the **`installedPrograms`** **array property.**

Keep in mind that when you successfully install a program you must **decrease** the total **hdd memory** on the computer with the **capacity** for the **currently installed program**!

The following function should **return** the **newly created object**.

uninstallAProgram({name})

This **function** should **uninstall** an **already installed** program on the computer (**remove** the **first program** with the **given name** from the **`installedPrograms`** **property**).

- If there **are no installed** programs with the **given name,** a **new error** should be **thrown** with the following message:
  "**`Control panel is not responding`**"
- If **`installedPrograms` property** contains an object with the **given name**, that object should be **removed** from the array.

Also logically reversed move is to **increase** the total **hdd memory** with the **capacity** of the **currently uninstalled program**!

This function should **return** the **`installedPrograms` array** where the given program name is excluded.

openAProgram({name})

This **function** should **open** an already installed program on the computer.

Receives a **string** (name of that program)

- If the given **name** is **not** present in the **`installedPrograms` property**, a **new error** should be **thrown** with the following message:
  "**`The ${name} is not recognized`**"
- If the given name is an installed program and it is already open, a new error should be thrown with the following message:
  `The ${name} is already open`"

To open an installed program, you must **calculate** how much **RAM memory** and **CPU usage** the program will need.

To find out how much:

- **ram memory** the current program will need, use the following formula:
  `(programRequiredSpace / totalRamMemory) * 1.5`
- **cpu usage** the current program will need, use the following formula:
  `( ( programRequiredSpace / CPU GHz ) / 500) * 1.5`

  Keep in mind the both formulas calculate a numbers in **percent** (%) for the current ram and cpu usage.

If the **total ram usage reaches or exceeds 100%** (the ram usage for all opened programs), the function should throw a **new error** with the following message:
`"{programName} caused out of memory exception`"

If the **total cpu usage reaches or exceeds 100%** (the cpu usage for all opened programs), the function should throw a **new error** with the following message:
"**`{programName} caused out of cpu exception`**"

If both (**ram usage and cpu usage**) **reaches** or **exceeds 100**% return the ram memory exception case.

When **ram** and **cpu usages** is **calculated**, create a **new object** with:

- **name** (name of the program)
- **ramUsage** (current ram usage that the program uses in %)

- **cpuUsage** (current cpu usage that the program uses in %).

The properties must be exactly as they are mentioned! Also, you don't have to round the numbers!

When the object is created, push it in the **`taskManager` array property.**

The function must **return** the **newly created object.**

taskManagerView()

This **function** prints all opened programs (the objects in the **`taskManager` array property**). Keep in mind that the percentages for (**cpu** and **ram usages**) must be shown **without** any **decimal part** (You can use the **`.toFixed(0)`**).

- If there is no opened program, the function **returns** a **string** with the following message: "**`All running smooth so far`**"
- If there is at least one opened program, visualize it in the following format: "**`Name - {programName} | Usage - CPU: {cpuUsage}%, RAM: {ramUsage}%`**"

If there is more than one opened program, each of them must be in **new line**.

This function **returns** a **string** in the format mentioned above.

Submission

Submit only your **`Computer` class.**

Examples

This is an example how the code is **intended to be used**:

Sample code usage

`let computer = new Computer(4096, 7.5, 250000);`
 	
`computer.installAProgram('Word', 7300);`
`computer.installAProgram('Excel', 10240);`
`computer.installAProgram('PowerPoint', 12288);`
`computer.uninstallAProgram('Word');`
`computer.installAProgram('Solitare', 1500);`
 	
`computer.openAProgram('Excel');`
`computer.openAProgram('Solitare');`
 	
`console.log(computer.installedPrograms);`
`console.log(('-').repeat(50)) // Separator`
`console.log(computer.taskManager);`

Corresponding output

```
[ { name: 'Excel', requiredSpace: 10240 },
  { name: 'PowerPoint', requiredSpace: 12288 },
  { name: 'Solitare', requiredSpace: 1500 } ]
-------------------------------------------------
[ { name: 'Excel', ramUsage: 3.75, cpuUsage: 4.096 },
  { name: 'Solitare',
    ramUsage: 0.54931640625,
    cpuUsage: 0.6000000000000001 } ]
```

Sample code usage

```
let computer = new Computer(4096, 7.5, 250000);

computer.installAProgram('Word', 7300);
computer.installAProgram('Excel', 10240);
computer.installAProgram('PowerPoint', 12288);
computer.installAProgram('Solitare', 1500);

computer.openAProgram('Word');
computer.openAProgram('Excel');
computer.openAProgram('PowerPoint');
computer.openAProgram('Solitare');

console.log(computer.taskManagerView());
```

Corresponding output

```
Name - Word | Usage - CPU: 3%, RAM: 3%
Name - Excel | Usage - CPU: 4%, RAM: 4%
Name - PowerPoint | Usage - CPU: 5%, RAM: 5%
Name - Solitare | Usage - CPU: 1%, RAM: 1%
```

GOOD LUCK! 😊