Smart Hike

As you are training for an ambitious goal – to hike all mountains in the country, you need a program to track your progress and records.

Smart Hike Class

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

Implement a **class SmartHike**, which supports the functionality described below.

Functionality

Constructor

The constructor has **4** properties:

- **username – a string**
- **goals – an empty object**
- **listOfHikes – an empty array**
- **resources – 100**

At the initialization of the **SmartHike** class, the **constructor** accepts only the **username!**

The **goals** property is an **object**, representing a key-value pair of a peak's name and its altitude.

Methods

addGoal (peak, altitude)

This method adds a new goal to the **goals** object. The method accepts 2 arguments:

- **peak – a string**
- **altitude – a number**

If the **goal exists in the goals object**, **return** the following message:

`` `${peak} has already been added to your goals` ``

Otherwise, **add the new goal** to the goals object and **return** the following message:

`` `You have successfully added a new goal - ${peak}` ``

hike (peak, time, difficultyLevel)

Accept 3 arguments:

- `peak – a string`
- `time – a number`
- `difficultyLevel – "hard" or "easy"`

If the peak doesn't exist in the goals object, **throw new Error**:

```
`${peak} is not in your current goals`
```

If the peak exists in the goals object but the **resources** are already **0**, **throw new Error**:

```
"You don't have enough resources to start the hike"
```

Afterwards, find the **difference** between the current **resources** and the **time**, multiplied by 10. If the difference is a negative number, **return** the following message:

```
"You don't have enough resources to complete the hike"
```

Otherwise extract the used resources from all resources and add the hike to the **list of hikes** in the following format: { peak, time, difficultyLevel }

Return the following message:

```
`You hiked ${peak} peak for ${time} hours and you have ${resources}% resources left`
```

rest (time)

Accept 1 argument:

- `time – a number`

Add the time for rest multiplied by 10 to the resources.

If the **resources are more than or equal to 100**, set them to 100 and **return** the following message:

```
`Your resources are fully recharged. Time for hiking!`
```

Otherwise, **return** the following message:

```
`You have rested for ${time} hours and gained ${time*10}% resources`
```

showRecord (criteria)

Accept 1 argument:

- `criteria – a string`

This method **returns information** based on the criteria. The three possible types of criteria are: **"hard"**, **"easy", "all".**

If the **list of hikes is empty**, return the following message:

```
`${username} has not done any hiking yet`
```

Find all hikes from the **list of hikes** depending on the given criterion **"hard" or "easy"** and find the best hike - the hike with the shortest time**.** If there are more than one hike with the same amount of time, list the **first** one added in the **list of hikes**.

Return the following message:

```
`${username}'s best ${criteria} hike is ${peak} peak, for ${time} hours`
```

If there is no hike with the given difficulty level, return:

```
`${username} has not done any ${criteria} hiking yet`
```

If the criterion is **"all", return** all hikes from the **list of hikes** array in following format:

- On first line show the following message:
  ```
  "All hiking records:"
  ```
- On the following lines, display information about each hike:

```
`${username} hiked ${peak} for ${time} hours`
```

Examples

Input 1

```
const user = new SmartHike('Vili');
console.log(user.addGoal('Musala', 2925));
console.log(user.addGoal('Rui', 1706));
console.log(user.addGoal('Musala', 2925));
```

Output 1

```
You have successfully added a new goal - Musala

You have successfully added a new goal - Rui

Musala has already been added to your goals
```

Input 2

`const user = new SmartHike('Vili');`
`console.log(user.addGoal('Musala', 2925));`
`console.log(user.addGoal('Rui', 1706));`
`console.log(user.hike('Musala', 8, 'hard'));`
`console.log(user.hike('Rui', 3, 'easy'));`
`console.log(user.hike('Everest', 12, 'hard'));`

Output 2
You have successfully added a new goal - Musala You have successfully added a new goal - Rui You hiked Musala peak for 8 hours and you have 20% resources left You don't have enough resources to complete the hike Uncaught Error: Everest is not in your current goals

Input 3

`const user = new SmartHike('Vili');`
`console.log(user.addGoal('Musala', 2925));`
`console.log(user.hike('Musala', 8, 'hard'));`
`console.log(user.rest(4));`
`console.log(user.rest(5));`

Output 3
You have successfully added a new goal - Musala

```
You hiked Musala peak for 8 hours and you have 20% resources left

You have rested for 4 hours and gained 40% resources

Your resources are fully recharged. Time for hiking!
```

Input 4

```
const user = new SmartHike('Vili');
console.log(user.showRecord('all'));
```

Output 4

```
Vili has not done any hiking yet
```

Input 5

```
const user = new SmartHike('Vili');
user.addGoal('Musala', 2925);
user.hike('Musala', 8, 'hard');
console.log(user.showRecord('easy'));
user.addGoal('Vihren', 2914);
user.hike('Vihren', 4, 'hard');
console.log(user.showRecord('hard'));
user.addGoal('Rui', 1706);
user.hike('Rui', 3, 'easy');
console.log(user.showRecord('all'));
```

Output 5

```
Vili has not done any easy hiking yet

Vili's best hard hike is Musala peak, for 8 hours

All hiking records:
```