System Split

This document defines the workshop for "Java OOP" course @ Software University. Please submit your solution (source code) of below described problem in Judge.

You have been given the task to gather statistics about The System. The System is a network of components, connected together to build something which functions logically, but you don't need to know that. You need to build a program which processes statistics about The System.

Overview

The System consists, mainly, of two types of components – **Hardware** and **Software** components.

Hardware components have a name, a type, a maximum capacity and a maximum memory.

There are **2 types** of Hardware components:

- Power Hardware decreases 75% of its given capacity, and increases its memory by 75%.
- Heavy Hardware -doubles its given capacity and decreases 25% of its given memory.

Software components have a name, a type, capacity consumption and memory consumption.

- **Express Software doubles** its given **memory consumption**.
- Light Software increases its given capacity consumption by 50% and decreases its given memory consumption by 50%.

```
Example: If a Power Hardware has 150 given capacity, his capacity will be - 75% from 150 =
150 - ((150 * 3) / 4) =
150 - (450 / 4) =
150 - 112 = 38
```

Note that you are working with **INTEGERS**.

Software components are stored on Hardware components. Each Software component takes up a specific amount of capacity and a specific amount of memory from the Hardware, in order to function properly. When registered, a Software component is stored on a **specified Hardware Component**.

There are several main commands you should configure in order for your program to function as needed.

Commands

- RegisterPowerHardware(name, capacity, memory)
- RegisterHeavyHardware(name, capacity, memory)
 - Registers a Hardware component of the specified type on The System with the given name, capacity, and memory.
- RegisterExpressSoftware(hardwareComponentName, name, capacity, memory)
- RegisterLightSoftware(hardwareComponentName, name, capacity, memory)
 - Registers a Software component of the specified type on the given Hardware component, with the given name. The Software Component takes up from the hardware's capacity and memory - the given capacity and memory.



© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

















- o If the given Hardware component does NOT exist in The System, the command should do nothing.
- If the given Hardware component does NOT have enough capacity or memory to contain the Software component, the command should do nothing.
- ReleaseSoftwareComponent(hardwareComponentName, softwareComponentName)
 - Destroys the Software Component with the given name, from the Hardware Component with the given name.
 - In case there is NO such Hardware Component, in The System, the command should do nothing.
 - In case there is NO such Software Component, on the given Hardware Component, the command should do nothing.

Analyze()

Shows statistics about the **components currently** in **The System** in the following format:

"System Analysis

Hardware Components: {countOfHardwareComponents}

Software Components: {countOfSoftwareComponents}

Total Operational Memory: {totalOperationalMemoryInUse} / {maximumMemory}

Total Capacity Taken: {totalCapacityTaken} / {maximumCapacity}"

 The total operational memory in use and total capacity taken is calculated from all the Software components currently in The System. You must also print the maximum memory and capacity available from all the Hardware Components currently in The System.

System Split

- o This command **finalizes** the work of the program, and prints information about the whole System.
- The System is split, and all of the Hardware components are to be printed **one by one**.
- The format of printing is the following:

"Hardware Component – {componentName}

Express Software Components - {countOfExpressSoftwareComponents}

Light Software Components - {countOfLightSoftwareComponents}

Memory Usage: {memoryUsed} / {maximumMemory}

Capacity Usage: {capacityUsed} / {maximumCapacity}

Type: {Power/Heavy}

Software Components: {softwareComponent1, softwareComponent2...}"

- Power Hardware Components must be printed before the Heavy Hardware Components.
- When printing the Software Components, print only their names.
- In case the Hardware component does not have any Software Components, print "None".
- The general **order of output** for all of the components is **by order of entrance**.

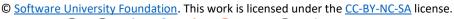
Input

- The input will come in the **form of commands**, in the format specified above.
- The input will consist **only** of the commands specified above.
- The input ends when you receive the command "System Split".

Output

- The only output you must print is the one specified for the Analyze command, and the final output.
- All of the output must be exactly in the format specified above.



















Constraints

- The names of the components will be strings, and will consist of English alphabet letters and digits.
- The names of the Hardware Components will always be unique.
- The names of the Software Components will be unique for every Hardware Component.
- The memory and capacity of each component will be integer numbers in range [0, 2³¹ 1].
- The type of a Hardware Component can be "Power" or "Heavy".
- The type of a Software Component can be "Express" or "Light".
- There will be **NO** invalid input commands.
- Allowed time/memory: 250ms / 32MB.

Examples

Input	Output
RegisterPowerHardware(HDD, 200, 200) RegisterHeavyHardware(SSD, 400, 400) Analyze() RegisterLightSoftware(HDD, Test, 0, 10) RegisterExpressSoftware(HDD, Test2, 100, 100) RegisterExpressSoftware(HDD, Test3, 50, 100) RegisterLightSoftware(SSD, Windows, 20, 50) RegisterExpressSoftware(SSD, Linux, 50, 100) RegisterLightSoftware(SSD, Unix, 20, 50) Analyze() ReleaseSoftwareComponent(SSD, Linux) System Split	System Analysis Hardware Components: 2 Software Components: 0 Total Operational Memory: 0 / 650 Total Capacity Taken: 0 / 850 System Analysis Hardware Components: 2 Software Components: 5 Total Operational Memory: 455 / 650 Total Capacity Taken: 160 / 850 Hardware Component - HDD Express Software Components - 1 Light Software Components - 1 Memory Usage: 205 / 350 Capacity Usage: 50 / 50 Type: Power Software Components: Test, Test3 Hardware Component - SSD Express Software Components - 0 Light Software Components - 2 Memory Usage: 50 / 300 Capacity Usage: 60 / 800 Type: Heavy Software Components: Windows, Unix

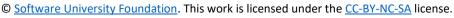
BONUS TASK: Dump Analysis

There is also a bonus task for you to implement in your program.

The System is hyper-dynamic – it is constantly changing its infrastructure. Addition and removal of components are frequent actions. For data safety reasons, The System contains a Dump. The Dump contains all elements that are temporarily deleted, so they can be restored if needed. If, however, the temporarily deleted components are **deleted from The Dump itself**, restoring them would be **impossible**.

- Dump(hardwareComponentName)
 - Removes from The System the Hardware component with the given name, and throws it into The **Dump**, along with all of its Software components.

















- o Dumped units do NOT take any memory or capacity on The System.
- o In case there is no component with the given name in The System, the command should do nothing.

Restore(hardwareComponentName)

- o Restores the given Hardware component, from **The Dump**, to **The System**.
- In case there is NO such component in The Dump, the command should do nothing.

Destroy(hardwareComponentName)

- Removes the given Hardware component from The Dump. After this action the component should no longer exist.
- In case there is **NO** such component in **The Dump**, the command should do nothing.

DumpAnalyze()

Shows statistics about the whole Dump in the following format:

"Dump Analysis

Power Hardware Components: {countOfPowerHardwareComponents} **Heavy Hardware Components: {countOfHeavyHardwareComponents}** Express Software Components: {countOfExpressSoftwareComponents} Light Software Components: {countOfLightSoftwareComponents}

Total Dumped Memory: {totalDumpedMemory} Total Dumped Capacity: {totalDumpedCapacity}"

The dumped memory, capacity, and is calculated from all the components, currently in The Dump.

Input	Output
RegisterPowerHardware(HDD, 300, 250)	System Analysis
RegisterHeavyHardware(SSD, 600, 1200)	Hardware Components: 0
RegisterExpressSoftware(HDD, Test1, 1, 1)	Software Components: 0
RegisterExpressSoftware(HDD, Test2, 1, 1)	Total Operational Memory: 0 / 0
RegisterExpressSoftware(HDD, Test3, 1, 1)	Total Capacity Taken: 0 / 0
RegisterLightSoftware(SSD, Test1, 5, 10)	Dump Analysis
RegisterLightSoftware(SSD, Test2, 5, 10)	Power Hardware Components: 1
Dump(HDD)	Heavy Hardware Components: 1
Dump(SSD)	Express Software Components: 3
Analyze()	Light Software Components: 2
<pre>DumpAnalyze()</pre>	Total Dumped Memory: 16
System Split	Total Dumped Capacity: 17

Input	Output
RegisterPowerHardware(CPU, 150, 235)	System Analysis
RegisterHeavyHardware(RAM, 450, 750)	Hardware Components: 1
RegisterExpressSoftware(CPU, ALU2, 10, 0)	Software Components: 0
Dump(CPU)	Total Operational Memory: 0 / 563
Analyze()	Total Capacity Taken: 0 / 900
Restore(CPU)	System Analysis
Analyze()	Hardware Components: 2
Dump(CPU)	Software Components: 1
Destroy(CPU)	Total Operational Memory: 0 / 974
RegisterPowerHardware(SSD, 3000, 5000)	Total Capacity Taken: 10 / 938
RegisterExpressSoftware(SSD, Windows, 400, 1750)	System Analysis
RegisterExpressSoftware(SSD, Skype, 50, 200)	Hardware Components: 2
RegisterExpressSoftware(SSD, Linux, 250, 300)	Software Components: 3
Analyze()	Total Operational Memory: 4500 / 9313
System Split	Total Capacity Taken: 700 / 1650

















Hardware Component - SSD

Express Software Components - 3 Light Software Components - 0 Memory Usage: 4500 / 8750 Capacity Usage: 700 / 750

Type: Power

Software Components: Windows, Skype,

Linux

Hardware Component - RAM Express Software Components - 0 Light Software Components - 0

Memory Usage: 0 / 563 Capacity Usage: 0 / 900

Type: Heavy

Software Components: None















