OOP Basics Exam – System Split

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** – decreases **25%** of its given **memory** and **doubles** its given **capacity**.

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**.
  - 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**
    - 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, 2^{31} - 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)`	`System Analysis`
`RegisterHeavyHardware(SSD, 400, 400)`	`Hardware Components: 2`
`Analyze()`	`Software Components: 0`
`RegisterLightSoftware(HDD, Test, 0, 10)`	`Total Operational Memory: 0 / 650`
`RegisterExpressSoftware(HDD, Test2, 100, 100)`	`Total Capacity Taken: 0 / 850`
`RegisterExpressSoftware(HDD, Test3, 50, 100)`	`System Analysis`
`RegisterLightSoftware(SSD, Windows, 20, 50)`	`Hardware Components: 2`
`RegisterExpressSoftware(SSD, Linux, 50, 100)`	`Software Components: 5`
`RegisterLightSoftware(SSD, Unix, 20, 50)`	`Total Operational Memory: 455 / 650`
`Analyze()`	`Total Capacity Taken: 160 / 850`
`ReleaseSoftwareComponent(SSD, Linux)`	`Hardware Component - HDD`
`System Split`	`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.

    - Dumped units **do NOT take** any **memory** or **capacity** on The System.
    - In case there is no component with the **given name** in The System, the command should do nothing.
- **Restore(hardwareComponentName)**
    - 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)
RegisterHeavyHardware(SSD, 600, 1200)
RegisterExpressSoftware(HDD, Test1, 1, 1)
RegisterExpressSoftware(HDD, Test2, 1, 1)
RegisterExpressSoftware(HDD, Test3, 1, 1)
RegisterLightSoftware(SSD, Test1, 5, 10)
RegisterLightSoftware(SSD, Test2, 5, 10)
Dump(HDD)
Dump(SSD)
Analyze()
DumpAnalyze()
System Split
	System Analysis
	Hardware Components: 0
	Software Components: 0
	Total Operational Memory: 0 / 0
	Total Capacity Taken: 0 / 0
	Dump Analysis
	Power Hardware Components: 1
	Heavy Hardware Components: 1
	Express Software Components: 3
	Light Software Components: 2
	Total Dumped Memory: 16
	Total Dumped Capacity: 17

Input	Output

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