# Exam Preparation

Part I: judge.[softuni](https://judge.softuni.bg/Contests/Practice/Index/996#0).bg/[Contests](https://judge.softuni.bg/Contests/Practice/Index/996#0)/[Practice](https://judge.softuni.bg/Contests/Practice/Index/996#0)/Index/996

Part II: [judge.softuni.bg/Contests/Practice/Index/996](https://judge.softuni.bg/Contests/Practice/Index/997#0)

# Programming Fundamentals Retake Exam –

# 25 April 2018

# Problem 1 – Rage Expenses

As a MOBA challenger player, Pesho has the bad habit to trash his PC when he loses a game and rage quits. His gaming setup consists of **headset, mouse, keyboard and display**. You will receive Pesho`s **lost games count**.

Every **second** lost game, Pesho trashes his **headset.**

Every **third** lost game, Pesho trashes his **mouse**.

When Pesho trashes **both** **his mouse and headset** in the **same** lost game, he also trashes his **keyboard**.

**Every** **second time, when he trashes his keyboard**, he also trashes his **display**.

You will receive the price of each item in his gaming setup. Calculate his rage expenses for renewing his gaming equipment.

## Input / Constraints

* On the first input line - **lost games count** – integer in the range **[0, 1000]**.
* On the second line – **headset price** - floating point number in range **[0, 1000]**.
* On the third line – **mouse price** - floating point number in range **[0, 1000]**.
* On the fourth line – **keyboard price** - floating point number in range **[0, 1000]**.
* On the fifth line – **display price** - floating point number in range **[0, 1000]**.

## Output

* As output you must print Pesho`s total expenses: **"Rage expenses: {expenses} lv."**
* Allowed working **time** / **memory**: **100ms** / **16MB**.

## Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 7  2  3  4  5 | Rage expenses: 16.00 lv. | Trashed headset -> 3 times  Trashed mouse -> 2 times  Trashed keyboard -> 1 time  Total: 6 + 6 + 4 = 16.00 lv; |
| 23  12.50  21.50  40  200 | Rage expenses: 608.00 lv. |  |

# Problem 2 – Memory View

As a Gamer, Pesho is thrilled with source code and is excited to have deeper understanding of the games code, so he started digging in the memory view. But because he can`t read it, you should write a programm which transforms the memory display in readable form.

Untill you receive **"Visual Studio crash"** you will be receiving lines from the memory view in 2-byte integer unsigned display. **Each line** consists of exactly 22 integers, separated by whitespace. You should find every string in the whole input and print them on the console.

Every string starts with -> "32656 19759 32763"

After the pointer there is one zero and the size of the string -> "0 5"

After the size of string there is one more zero and on the next n columns are the integers for

each character -> **" 0 80 101 115 104 111"**

The above example is the string "Pesho".

You must find all strings and display them on the console, using the **ASCII code for each character.**

## Input

* The input will consist of **several** **lines of 22 integers, separated by spaces**.

## Output

* You should print on a new line every string you`ve found in the input in their order of appearance.

## Constraints

* The lines of the input may contain any 32-bit integer in the range [0 – 32763].
* The input will allways be valid.
* Allowed working **time** / **memory**: **100ms** / **16MB**.

## Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 32656 19759 32763 0 5 0 80 101 115 104 111 0 0 0 0 0 0 0 0 0 0 0  0 32656 19759 32763 0 7 0 83 111 102 116 117 110 105 0 0 0 0 0 0 0 0  Visual Studio crash | Pesho  Softuni |
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 32656 19759 32763 0  5 0 80 101 115 104 111 0 0 0 0 0 0 0 0 0 32656 19759 32763 0 4 0  75 105 114 111 0 0 0 0 0 0 0 0 0 0 32656 19759 32763 0 8 0 86 101  114 111 110 105 107 97 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  Visual Studio crash | Pesho  Kiro  Veronika |

# Problem 3 – Tseam Account

As a gamer, Pesho has Tseam Account. He loves to buy new games. You are given Pesho`s account with all of his games-> strings, separated by space. Until you receive **"Play!"** you will be receiving commands which Pesho does with his account.

You may receive the following commands:

* Install {game}
* Uninstall {game}
* Update {game}
* Expansion {game}-{expansion}

If you receive **Install command**, you should **add** the game at last position in the account, but only if it isn`t installed already.

If you receive **Uninstall command**, **delete** the game if it exists.

If you receive **Update command**, you should **update** the game if it exists and place it on **last position**.

If you receive **Expansion command**, you should check if the game exists and **insert** after it the expansion in the following format: "**{game}:{expansion}";**

## Input

* On the **first input line** you will receive Pesho`s **account** – sequence of game names, separated by space.
* Until you receive **"Play!"** you will be receiving **commands**.

## Output

* As output you must print Pesho`s Tseam **account**.

## Constraints

* The **command will always be valid.**
* The **game** and **expansion** will be strings and will contain any character, except **'-'**.
* Allowed working **time** / **memory**: **100ms** / **16MB**.

## Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| CS WoW Diablo  Install LoL  Uninstall WoW  Update Diablo  Expansion CS-Go  Play! | CS CS:Go LoL Diablo | We receive the account => CS, WoW, Diablo  We Install LoL => CS, WoW, Diablo, LoL  Uninstall WoW => CS, Diablo, LoL  Update Diablo => CS, LoL, Diablo  We add expansion => CS, CS:Go, LoL, Diablo  We print the account. |
| CS WoW Diablo  Uninstall XCOM  Update PeshoGame  Update WoW  Expansion Civ-V  Play! | CS Diablo WoW |  |

# Problem 4. MOBA Challenger

Pesho is a pro MOBA player, he is struggling to become master of the Challenger tier. So he watches carefully the statistics in the tier.

You will receive **several input lines** in one of the following formats:

"{player} -> {position} -> {skill}"

"{player} vs {player}"

The player and position are strings, the given **skill** will be an integer number. You need to keep track of **every player**.

When you receive a **player and his position and skill**, add him to the player pool, if he isn`t present, **else add** his position and skill **or update** his skill, only if the current position skill is lower than the new value.

If you receive **"{player} vs {player}"** and **both players exist** in the tier, **they duel** with the following rules:

Compare their positions, **if they got at least one in common**, the player with better **total skill points** wins and the other is **demoted** from the tier -> remove him. If they have same total skill points, **the duel is tie** and they both continue in the Season.

If they don`t have positions in common, **the duel isn`t happening** and both continue in the Season.

You should end your program when you receive the command "Season end". At that point you should print the players, **ordered by total skill in desecending order, then ordered by player name in ascending order**. **Foreach** player print their position and skill, **ordered desecending by skill, then ordered by position name in ascending order.**

## Input / Constraints

* The input comes in the form of commands in one of the formats specified above.
* Player and position **will always be one word string, containing no whitespaces**.
* Skill will be an **integer** in the **range [0, 1000]**.
* There will be **no invalid** input lines.
* The programm ends when you receive the command "Season end".

## Output

* The output format for each player is:

"{player}: {totalSkill} skill"

"- {position} <::> {skill}"

## Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| Pesho -> Adc -> 400  Gosho -> Jungle -> 300  Stamat -> Mid -> 200  Stamat -> Support -> 250  Season end | Stamat: 450 skill  - Support <::> 250  - Mid <::> 200  Pesho: 400 skill  - Adc <::> 400  Gosho: 300 skill  - Jungle <::> 300 | We order the players by total skill points descending, then by name. We print every position along its skill ordered descending by skill, then by position name. |
| **Input** | **Output** |  |
| Pesho -> Adc -> 400  Bush -> Tank -> 150  Faker -> Mid -> 200  Faker -> Support -> 250  Faker -> Tank -> 250  Pesho vs Faker  Faker vs Bush  Faker vs Hide  Season end | Faker: 700 skill  - Support <::> 250  - Tank <::> 250  - Mid <::> 200  Pesho: 400 skill  - Adc <::> 400 | Faker and Pesho don`t have common position, so the duel isn`t valid.  Faker wins vs Bush /common position: "Tank". Bush is demoted.  Hide doesn`t exist so the duel isn`t valid.  We print every player left in the tier. |