# Практически изпит (5 ноември 2017 г.)

Практически упражнения към курса [**"Programming Fundamentals" за ученици**](https://github.com/BG-IT-Edu/School-Programming/tree/main/Courses/Applied-Programmer/Programming-Fundamentals).

Тествайте задачата в judge: [https://judge.softuni.bg/Contests/2674](https://judge.softuni.bg/Contests/2674/Практически-изпит-05-11-2017)

# Problem 1. Anonymous Downsite

The Anonymous informal group of activists have hacked a few commercial websites and the CIA has hired you to write a software which calculates the losses. Based on the given data, use the appropiate data types.

You will receive **2 input lines** – each containing an **integer**.

* The **first** is **N** – the **number** of **websites** which are down.
* The **second** is the security key.

On the **next N lines** you will receive **data** about **websites** in the following format:  
{siteName} {siteVisits} {siteCommercialPricePerVisit}

You must **calculate** the site loss by the following formula: siteVisits \* siteCommercialPricePerVisit

When you **finish reading all data**, you must print the **affected sites’ names** – each on a **new line**.  
Then you must print the total money loss – **sum** of all site loss, on a **new line**.  
Finally you must print the security token, which is the security key, **POWERED** by the **COUNT** of **affected sites**.

### Input

* On the **first input line** you will get **N** – the **count** of **affected websites**.
* On the **second input line** you will the **security key**.
* On the **next N input lines** you will get **data** about the **websites**.

### Output

* As output you must print **all affected websites’ names** – **each** on a **new line**.
* **After** the **website names** you must print the **total loss** of **data**, printed to the **20th digit** after the **decimal point**. The format is “Total Loss: {totalLoss}”.
* Finally you must **print** the **security token**. The format is “Security Token: {securityToken}”.

### Constrains

* The integer **N** will be in **range** **[0, 100]**.
* The **security token** will be in **range** **[0, 10]**.
* The **website name** may contain any **ASCII character** except **whitespace**.
* The **site visits** will be an **integer** in **range [0, 231].**
* The **price per visit** will be a **floating point number** in **range [0, 100]** and will have **up** to **20 digits** after the decimal point.
* Allowed working **time/memory**: **100ms / 16MB**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  8  www.google.com 122300 94.23233  www.abv.bg 2333 11  www.kefche.com 12322 23.3222 | www.google.com  www.abv.bg  www.kefche.com  Total Loss: 11837653.10740000000000000000  Security Token: 512 |
| 1  1  www.facebook.com 100000 10.45 | www.facebook.com  Total Loss: 1045000.00000000000000000000  Security Token: 1 |

*Remember, remember!*

*The fifth of November...*

# Problem 2. Anonymous Threat

The Anonymous have created a cyber hypervirus which steals data from the CIA. You, as the lead security developer in CIA, have been tasked to analyze the software of the virus and observe its actions on the data. The virus is known for his innovative and unbeleivably clever technique of merging and dividing data into partitions.

You will receive a **single input line** containing **STRINGS** separated by **spaces**.   
The strings may contain **any ASCII** character except **whitespace**.

You will then begin receiving commands in one of the following formats:

* merge {startIndex} {endIndex}
* divide {index} {partitions}

Every time you receive the merge command, you must merge all elements from the startIndex, till the endIndex. In other words, you should concatenate them.   
**Example**: {abc, def, ghi} -> merge 0 1 -> {abcdef, ghi}

If **any** of the **given indexes** is **out of the array**, you must take **ONLY** the **range** that is **INSIDE** the **array** and **merge** it.

Every time you receive the divide command, you must **DIVIDE** the **element** at the **given index**, into **several small substrings** with **equal length**. The **count** of the **substrings** should be **equal** to the **given partitions**.

**Example**: {abcdef, ghi, jkl} -> divide 0 3 -> {ab, cd, ef, ghi, jkl}

If the string **CANNOT** be **exactly** **divided** into the **given partitions**, **make all partitions** **except** the **LAST** with **EQUAL LENGTHS**, and make the **LAST one** – **the** **LONGEST**.

**Example**: {abcd, efgh, ijkl} -> divide 0 3 -> {a, b, cd, efgh, ijkl}

The **input ends** when you receive the command “3:1”. At that point you must print the **resulting elements**, **joined** by a **space**.

### Input

* The **first input line** will contain the **array** of **data**.
* On the **next several input** lines you will **receive commands** in the **format specified above**.
* The **input ends** when you receive the command “3:1”.

### Output

* As output you must print a single line containing the elements of the array, **joined** by a **space**.

### Constrains

* The **strings** in the **array** may contain any **ASCII character** except **whitespace**.
* The startIndex and the endIndex will be in **range [-1000, 1000]**.
* The endIndex will **ALWAYS** be **GREATER** than the startIndex.
* The index in the divide command will **ALWAYS** be **INSIDE** the array.
* The partitions will be in **range [0, 100]**.
* Allowed working **time/memory**: **100ms / 16MB**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Ivo Johny Tony Bony Mony  merge 0 3  merge 3 4  merge 0 3  3:1 | IvoJohnyTonyBonyMony |
| abcd efgh ijkl mnop qrst uvwx yz  merge 4 10  divide 4 5  3:1 | abcd efgh ijkl mnop qr st uv wx yz |

*...The Gunpowder treason and plot;...*

# Problem 3. Anonymous Vox

The Anonymous’s main communication channel is based on encoded messages. The CIA has targetted that channel, assuming that it holds sensitive information. You have been hired to decode and break their internal com. system.

You will receive an input line containing a **single string** – the **encoded text**. Then, on the **next line** you will receive several values in the following format: “{value1}{value2}{value3}...”.

You must find the **encoded placeholders** in the **text** and **REPLACE** each one of them with the **value** that corresponds to its **index**.   
**Example**: **placeholder1 – value1**, **placeholder2 – value2** etc. There may be **more values** than **placeholders** or **more placeholders** than **values**.

The **placeholders** consist of 3 blocks {start}{placeholder}{end}. The **start** should consist only of **English alphabet letters**. The **placeholder** may contain **ANY ASCII** character. The **end** should be **EXACTLY EQUAL** to the **start**. The idea is that you have to find the **placeholders**, and **REPLACE** their placeholder block with the **value** at that **index**.

Example Placeholders: “a.....a”, “b!d!b”, “asdxxxxxasd”, “peshogoshopesho”...

You **must** **ALWAYS** match the placeholder with the **LONGEST** start and the **RIGHTMOST** end. For example if you have “asddvdasd” you should **NOT** match “dvd” as a placeholder, you should match “asddvdasd”.

At the end you must **print** the **result** **text**, after you’ve **replaced** the **values**.

### Input

* On the **first input line** you will receive the **encoded text**.
* On the **second input line** you will receive the **placeholders**.

### Output

* As output you must print a **single line** containing the **resulting text**, after the replacing of values.

### Constrains

* The **given** **text** may contain **ANY ASCII** character.
* The **given values** may contain **ANY ASCII** character except ‘{’ and ‘}’.
* The **given values** will **AWLAYS** follow the format specified above.
* Allowed working **time/memory**: **100ms / 16MB**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Hello\_mister,\_Hello  { Jack } | Hello Jack Hello |
| ASD\_\_\_asdfffasd  {this}{exam}{problem}{is}{boring} | ASD\_\_\_asdthisasd |
| Whatsup\_ddd\_sup  {Dude} | WhatsupDudesup |
| HeypalHey\_\_\_\_\_\_how\_ya\_how\_doin\_how  {first}{second} | HeyfirstHey\_\_\_\_\_\_howsecondhow |

*...I know of no reason...*

# Problem 4. Anonymous Cache

The Anonymous are storing data on their dataservers about their activities. The CIA has higher the greatest hacker in the world – You. Your job is to extract their data and send it to the CIA. It won’t be an easy task, Get Ready!

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

* {dataSet}
* {dataKey} -> {dataSize} | {dataSet}

The dataSet and dataKey are both strings. The dataSize is an **integer**. The dataSets hold dataKeys and their dataSizes.

If you receive only a dataSet you should **add** it. If you receive a dataKey and a dataSize, you should add them to the **given** dataSet.

And here’s where the fun begins. If you receive a dataKey and a dataSize, but the given dataSet **does NOT exist**, you should **STORE** those **keys** and **values** in a cache. When the corresponding dataSet is **added**, you should **check** if the cache holds any **keys** and **values** referenced to it, and you should **add** them to the dataSet.

You should end your program when you receive the command “thetinggoesskrra”. At that point you should extract the dataSet from the data with the **HIGHEST** dataSize (**SUM** of all its dataSizes), and you should print it.

**NOTE**: Elements in the cache, **should be CONSIDERED NON-EXISTANT**. You should **NOT** count them in the **final output**.

In case there are **NO** dataSets in the data, you **should** **NOT** **do anything**.

### Input

* The input comes in the form of commands in one of the formats specified above.
* The input ends when you receive the command “thetinggoesskrra”.

### Output

* As output you must print the dataSet with the **HIGHEST** **SUM** of all dataSizes.
* The output format is:

Data Set: {dataSet}, Total Size: {sumOfAllDataSizes}

$.{dataKey1}

$.{dataKey2}

...

* In case there are **NO** dataSets in the data, print **nothing**.

### Constrains

* The dataSet and dataKey are **both strings** which may contain **ANY ASCII** character except ‘ ’, ‘-’, ‘>’, ‘|’.
* The dataSize is a **valid integer** in **range [0, 1.000.000.000]**.
* There will be **NO invalid input lines**.
* There will be **NO** dataSets with **EQUAL SUMMED** dataSize.
* There will be **NO DUPLICATE** keys.
* Allowed working **time/memory**: **100ms / 16MB**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Users  BankAccounts  ADDB444 -> 23111 | BankAccounts  Students -> 2000 | Users  Workers -> 24233 | Users  thetinggoesskrra | Data Set: Users, Total Size: 26233  $.Students  $.Workers |
| Cars  Car1 -> 233333 | Cars  Car23 -> 266666 | Cars  Warehouse2 -> 10000 | Buildings  Warehouse3 -> 480000 | Buildings  Warehouse5 -> 100000 | Buildings  Buildings  thetinggoesskrra | Data Set: Buildings, Total Size: 590000  $.Warehouse2  $.Warehouse3  $.Warehouse5 |

*...Why the Gunpowder treason*

*Should ever be forgot!...*

## Министерство на образованието и науката (МОН)

* Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист".



* Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под **свободен** **лиценз CC-BY-NC-SA** (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).

