# Допълнителни задачи – ламбда изрази и речници

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

Тествайте задачите от тази тема в judge: [https://judge.softuni.bg/Contests/2672](https://judge.softuni.bg/Contests/2672/Допълнителни-задачи-ламбда-изрази-и-речници)

## Сортиране на часове

Напишете програма, която получава **списък от часове** (разделени с интервал, 24-часове формат) и ги **сортира** в **нарастващ ред**. Изведете сортираните часове разделени с интервали.

Примери: **06:55, 02:30, 23:11** 🡺 **02:30, 06:55, 21:11**

### Пример

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 00:00 06:04 02:59 10:33 11:22 06:01 | 00:00, 02:59, 06:01, 06:04, 10:33, 11:22 |
| 04:25 04:21 04:19 | 04:19, 04:21, 04:25 |
| 00:00 23:59 12:00 16:00 | 00:00, 12:00, 16:00, 23:59 |

## Нечетен филтър

Напишете програма, която получава масив от цели числа (раздели с интервал), премахва всички нечетни числа, след което превръща останалите числа в нечетни числа, според:

* Ако числото е по-голямо от средното от колекцията оставащи числа, добавяме 1
* Ако числото е по-малко от средното от колекцията оставащи числа, то намаляваме с 1.

След като превърнете всички елементи към нечетни числа, изведете масива

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 1 2 3 4 5 6 7 8 9 10 | 1 3 5 9 11 |
| 99 88 77 66 55 4 33 22 11 | 89 67 3 21 |
| 23 32 199 723 8127 95 | 31 |

## Имунна система

Всеки организъм може да бъде нападнат от различни видове вируси. Информация за тях се съхраняват в имунната му система. Ако тя вече е среща вируса, ще го победи по-бързо, отколкото ако го среща за първи път.

Имунната система може да изчисли силата на вируса преди да го победи. Това е сумата от кодовете на всички букви в името на вируса, разделена на 3.

Имунната система може да изчисли времето, което трябва да се победи даден вирус в секунди. То е равно на силата на вируса умножено по дължината на името на вируса.

Когато изчислите времето за побеждаване на вируса, превърнете го в минути и секунди, като не използвате водещи нули.

* Ако имунната система победи вируса, изведете:   
  “{virusName} defeated in {virusDefeatMinutes}m {virusDefeatSeconds}s.”
* Ако силата на вируса е по-голяма от силата на имунната система, изведете “Immune System Defeated.” И приключете изпълнението на програмата.

След като вируса е победен, имунната система си възвръща 20% от нейната сила. Ако 20-те процента надвишават първоначалната сила на имунната система, задайте състоянието ѝ до нейната първоначална сила.

Пример: вирус “flu1”:

* сила: **102 (f) + 108 (l) + 117 (u) + 49 (1)** = **376 / 3** = **125.33 = 125**.
* Време за победа: 125 \* 4 (дължина на името) = 500 секунди 🡺 8m 20s.

Пример 2: Срещна на “flu1” за втори път:

* Време за победа: **(125 \* 4) / 3** = **166.66 🡺 166 секунди**

Ако се срещне един и същ вирус последователно, НЕ намалявайте времето за неговата победа допълнително. Когато получите “end”, изведете статуса на имунната система във формата “Final Health: {finalHealth}”.

### Вход

* На първи ред: началната сила на имунната система
* На всеки нов ред, до срещане на “end”: имена на вируси

### Изход

Изход при победен вирус изглежда така:

* Първи ред: “Virus {virusName}: {virusStrength} => {virusDefeatSeconds}”
* Втори ред: “{virusName} defeated in {defeatMins}m {defeatSecs}s.”
* Трети ред: “Remaining health: {remainingHealth}”. Оставащото здраве се извежда преди да се регенерира.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5000  flu1  test  flu1  virusssssss  end | Virus flu1: 125 => 500 seconds  flu1 defeated in 8m 20s.  Remaining health: 4500  Virus test: 149 => 596 seconds  test defeated in 9m 56s.  Remaining health: 4404  Virus flu1: 125 => 166 seconds  flu1 defeated in 2m 46s.  Remaining health: 4834  Virus virusssssss: 419 => 4609 seconds  virusssssss defeated in 76m 49s.  Remaining health: 391  Final Health: 469 |
| 1750  Ebola  ebola  Ebola  end | Virus Ebola: 161 => 805 seconds  Ebola defeated in 13m 25s.  Remaining health: 945  Virus ebola: 171 => 855 seconds  ebola defeated in 14m 15s.  Remaining health: 279  Virus Ebola: 161 => 268 seconds  Ebola defeated in 4m 28s.  Remaining health: 66  Final Health: 79 |
| 5700  wannacry  iskaplache  wannacry | Virus wannacry: 289 => 2312 seconds  wannacry defeated in 38m 32s.  Remaining health: 3388  Virus iskaplache: 348 => 3480 seconds  iskaplache defeated in 58m 0s.  Remaining health: 585  Virus wannacry: 289 => 770 seconds  Immune System Defeated. |

## Поправка на Email-и

Напишете програма, в която получавате последователност от низове, всеки на нов ред, докато срещнете команда “stop”. Първия низ е името на човека. На втори ред, ще получите имейл. Вашата задача е да съберете техните имена и имейли, след което трябва да премахнете имейлите, чиито домейни завършват с "**us**" или "**uk**" (без значение от големината на буквите). Извеждайте в следния формат:

**{name} – > {email}**

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Ivan  ivanivan@abv.bg  Petar Ivanov  petartudjarov@abv.bg  Mike Tyson  myke@gmail.us  stop | Ivan -> ivanivan@abv.bg  Petar Ivanov -> petartudjarov@abv.bg |

## Добра ръка

Напишете програма, която въвежда поредица от хора, като за всеки човек се въвежда какви карти е изтеглил от тестето. Входните данни ще бъдат на отделни редове в следния формат:

* {име}: {PT, PT, PT,… PT}

Където **P (2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A**) е номера/буквата на картата, а **T (S, H, D, C)** е вида. Входа приключва тогава, когато се изтегли **"JOKER"**. Името може да съдържа всякакви ASCII символи, освен **':'**. Входът винаги ще бъде валиден, според описания формат и няма нужда да го проверявате.

Един човек **не може да има повече от една карта от един и същ номер и вид**, ако изтеглят такава карта, те я **отказват**. Хората играят с **няколко тестета**. Всяка карта има стойност, която е изчислена от **номера умножен по типа**. Номерата от 2 до 10 си имат същите стойности, а буквите от **J до A** са **11 до 14**. Типовете съответстват на числа по следния начин: **S -> 4, H-> 3, D -> 2, C -> 1**.

Накрая изведете общата стойност, която всеки играч има в ръката си във формата:

* {име}: {стойност}

### Пример

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Pesho: 2C, 4H, 9H, AS, QS  Slav: 3H, 10S, JC, KD, 5S, 10S  Peshoslav: QH, QC, QS, QD  Slav: 6H, 7S, KC, KD, 5S, 10C  Peshoslav: QH, QC, JS, JD, JC  Pesho: JD, JD, JD, JD, JD, JD  JOKER | Pesho: 167  Slav: 175  Peshoslav: 197 |

## \* Потребителски логове

Мариян е известен сисадмин. Все още не се е родил човека, който може да пробие неговите сървъри. Все пак, има нов вид заплаха, при която потребители наводняват сървъра със съобщение и е трудно да бъдат засечени, понеже си менят IP адресите. Понеже Мариян е сисадмин, а не програмист, той ще има нужда от опитен програмист, за да проследи логовете. Сега топката е у вас!

Ще получите вход в следния формат:

* **IP=(IP.Address) message=(A&sample&message) user=(username)**

Вашата задача е да извлечете IP-то и потребителското име от входните данни за **всеки потребител**, трябва да покажете **всяко IP**, от което съответния потребител е изпратил съобщение и да преброите съобщенията изпратени със съответното IP. В изхода, потребителските имена трябва да бъдат **сортирани азбучно**, докато техните IP адреси трябва да бъдат сортирани по реда на тяхното **първо появяване**.Изходът трябва да бъде в следния формат:

|  |
| --- |
| **Потребителско име:**  **IP => брой, IP => брой.** |

Например, даден е следния вход:

* “**IP=192.23.30.40 message='Hello&derps.' user=destroyer**”,

Ще извлечете потребителското име **destroyer** и IP **192.23.30.40** и ще го покажете в следния формат:

|  |
| --- |
| **destroyer:**  **192.23.30.40 => 1.** |

Потребител **destroyer** е изпратил съобщение от IP **192.23.30.40** веднъж.

Разгледайте примерите по-долу. Те ще изяснят условието.

### Вход

Входът ще бъде подаден като **вариращ брой редове**. Ще трябва да обработвате всяка команда, докато получите команда **end.** Входът ще бъде във формата показан по-горе, няма нужда да го проверявате.

### Изход

За всеки намерен потребител, ще трябва да изведете всеки лог във формата:

**потребител:**

**IP => брой, IP => брой…**

IP адресите трябва да бъдат разделени със запетая и всеки IP адрес трябва да приключва с точка.

### Ограничения

* Броят на командите ще бъде в интервала **[1..50]**
* IP адресите ще бъдат във формата на **IPv4** или **IPv6.**
* Съобщението ще е във формата: This&is&a&message
* Потребителското име ще бъде низ с дължина **[3..50]**
* Времеви лимит: 0.3 секунди. Лимит по памет: 16 MB.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| IP=192.23.30.40 message='Hello&derps.' user=destroyer  IP=192.23.30.41 message='Hello&yall.' user=destroyer  IP=192.23.30.40 message='Hello&hi.' user=destroyer  IP=192.23.30.42 message='Hello&Dudes.' user=destroyer  end | destroyer:  192.23.30.40 => 2, 192.23.30.41 => 1, 192.23.30.42 => 1. |
| IP=FE80:0000:0000:0000:0202:B3FF:FE1E:8329 message='Hey&son' user=mother  IP=192.23.33.40 message='Hi&mom!' user=child0  IP=192.23.30.40 message='Hi&from&me&too' user=child1  IP=192.23.30.42 message='spam' user=destroyer  IP=192.23.30.42 message='spam' user=destroyer  IP=192.23.50.40 message='' user=yetAnotherUsername  IP=192.23.50.40 message='comment' user=yetAnotherUsername  IP=192.23.155.40 message='Hello.' user=unknown  end | child0:  192.23.33.40 => 1.  child1:  192.23.30.40 => 1.  destroyer:  192.23.30.42 => 2.  mother:  FE80:0000:0000:0000:0202:B3FF:FE1E:8329 => 1.  unknown:  192.23.155.40 => 1.  yetAnotherUsername:  192.23.50.40 => 2. |

## \* Преброяване на населението

Толкова много хора! Толкова е трудно да ги преброиш. Но това е работата ви на статистик. Ще получите сурови данни за даден град, а вие трябва да ги обработите.

На всеки ред, ще получите информация във формата: **"**град|държава|население**"**. Няма да има излишни интервали никъде във входните данни**.** Обработете данните по **държава и град** и ги изведете в конзолата.

За всяка държава, изведете нейното **общо население** и на отделни редове данните за всеки град. Държавите се подреждат по тяхното **общо население в намаляващ ред**, а в рамките на всяка държава градовете се подреждат **по същия начин**.

Ако две държави/град имат едно и също население, пазете ги по реда им на въвеждане. Разгледайте примерите и следвайте точно форматирането!

### Вход

* Входните данни се въвеждат от конзолата
* Състоят се от променлив брой редове и приключва, когато се извика команда "**report**".
* Входните данни винаги ще бъдат валидни и ще следват описания формат. Няма нужда да бъдат проверявани.

### Изход

* Изходът трябва да бъде отпечатан на конзолата.
* Изведете обработената информация за всяка държава и град, според форматирането от примерите.

### Ограничения

* Името на града, страната и броя на населението ще бъдат отделени с права черта **('|')**.
* **Броят на редовете** ще е в интервала [2 … 50].
* Двойка град-държава няма да бъде повторена.
* Броят на населението във всеки град ще бъде цяло число в интервала [0 … 2 000 000 000].
* Лимит по време: 0.1 секунди. Лимит по памет: 16 MB.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Sofia|Bulgaria|1000000  report | Bulgaria (total population: 1000000)  =>Sofia: 1000000 |

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Sofia|Bulgaria|1  Veliko Tarnovo|Bulgaria|2  London|UK|4  Rome|Italy|3  report | UK (total population: 4)  =>London: 4  Bulgaria (total population: 3)  =>Veliko Tarnovo: 2  =>Sofia: 1  Italy (total population: 3)  =>Rome: 3 |

## \* Обработка на логове

Ще получите логове в следния формат **<IP> <потребител> <продължителност>**. Например:

* 192.168.0.11 peter 33
* 10.10.17.33 alex 12
* 10.10.17.35 peter 30
* 10.10.17.34 peter 120
* 10.10.17.34 peter 120
* 212.50.118.81 alex 46
* 212.50.118.81 alex 4

Напишете програма, която обработва логовете и извежда за **всеки потребител** общата продължителност на неговите сесии и списък от уникални IP адреси във формата: "**<потребител>: <продължителност> [<IP1>, <IP2>, …]**". Подредете **потребителите в азбучен ред**. Подредете **IPтата в азбучен ред**. В нашия пример, изходът би следния:

* alex: 62 [10.10.17.33, 212.50.118.81]
* peter: 303 [10.10.17.34, 10.10.17.35, 192.168.0.11]

### Вход

Входните данни се въвеждат в конзолата. На първи ред ще получите числото **n**, което ще ви покаже колко реда от лога ще получите. Всеки от следващите n реда ще пази информация от лога във формат **<IP> <потребител> <продължителност>**. Входните данни винаги ще са валидни и ще отговарят на посочения формат. Няма нужда да ги проверявате.

### Изход

Изведете по **един ред за всеки потребител** (подреждайте потребителите в азбучен ред). За всеки потребител извеждайте сумата от продължителността и всички негови IPта, подредени азбучно във формат **<потребител>: <продължителност> [<IP1>, <IP2>, …]**. Премахнете всякакви дублирани стойности в IP адресите и ги подредете азбучно (както се подреждат низове).

### Ограничения

* Броят **n** на редовете е в интервала [1…1000].
* **<IP>** е стандартен IP адрес във формата **a.b.c.d** където **a**, **b**, **c** и **d** са цели числа в интервала [0…255].
* **<потребителско име>** се състои от **латински букви**, с дължина [1…20].
* **<продължителност>** е цяло число в интервала [1…1000].
* Лимит по време: 0.3 sec. Лимит по памет: 16 MB.

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 7  192.168.0.11 peter 33  10.10.17.33 alex 12  10.10.17.35 peter 30  10.10.17.34 peter 120  10.10.17.34 peter 120  212.50.118.81 alex 46  212.50.118.81 alex 4 | alex: 62 [10.10.17.33, 212.50.118.81]  peter: 303 [10.10.17.34, 10.10.17.35, 192.168.0.11] |
| 2  84.238.140.178 nakov 25  84.238.140.178 nakov 35 | nakov: 60 [84.238.140.178] |

## 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).

