# Exercise: Java Fundamentals

This exercise is part of the [“Databases Frameworks” course @ SoftUni](https://softuni.bg/trainings/1635/databases-frameworks-hibernate-and-spring-data-june-2017).

## Students by Group

Print all students from group number 2. Use a Stream query. Order the students by **FirstName**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Mills 1  Andrew Gibson 2  Craig Ellis 1  Steven Cole 2  Andrew Carter 2  END | Andrew Gibson  Andrew Carter  Steven Cole |

## Students by First and Last Name

Using the same input as above print all students whose first name is before their last name lexicographically. Use a Stream query. Print them in order of appearance.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Mills  Andrew Gibson  Craig Ellis  Steven Cole  Andrew Carter  END | Andrew Gibson  Craig Ellis  Andrew Carter |

## Students by Age

Write a Stream query that finds the first name and last name of all students with age between 18 and 24. The query should return the **first name**, **last name** and **age**. Print them in order of appearance.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Mills 24  Andrew Gibson 21  Craig Ellis 19  Steven Cole 35  Andrew Carter 15  END | Sara Mills 24  Andrew Gibson 21  Craig Ellis 19 |

## Sort Students

Using the lambda expressions with Stream query syntax sort the students first by **last name** in **ascending** order and then by **first name** in **descending** order.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Gibson  Andrew Gibson  Craig Ellis  Steven Cole  Andrew Ellis  END | Steven Cole  Craig Ellis  Andrew Ellis  Sara Gibson  Andrew Gibson |

## Filter Students by Email Domain

Print all students that have email **@gmail.com**. Use Stream API. Print the in order of appearance.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Mills smills@gmail.com  Andrew Gibson agibson@abv.bg  Craig Ellis cellis@cs.edu.gov  Steven Cole themachine@abv.bg  Andrew Carter [ac147@gmail.com](mailto:ac147@gmail.com)  END | Sara Mills  Andrew Carter |

## Filter Students by Phone

Print all students with phones in Sofia (starting with **02** / **+3592**). Use a Stream.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Mills 02435521  Andrew Gibson 0895223344  Craig Ellis +3592667710  Steven Cole 3242133312  Andrew Carter +001234532  END | Sara Mills  Craig Ellis |

## Excellent Students

Print all students that have **at least one mark Excellent (6)**. Use a Stream.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Mills 6 6 6 5  Andrew Gibson 3 4 5 6  Craig Ellis 4 2 3 4  Steven Cole 5 6 5 5  Andrew Carter 5 3 4 2  END | Sara Mills  Andrew Gibson  Steven Cole |

## Weak Students

Write a similar program to the previous one to extract the **students with at least 2 marks under or equal to "3"**. Use a Stream.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Sara Mills 6 6 6 5  Andrew Gibson 3 4 5 6  Craig Ellis 4 2 3 4  Steven Cole 5 6 5 5  Andrew Carter 5 3 4 2  END | Craig Ellis  Andrew Carter |

## Students Enrolled in 2014 or 2015

Extract and print the **Marks** of the students that **enrolled in 2014 or 2015** (the students from 2014 have 14 as their 5-th and 6-th digit in the **FacultyNumber**, those from 2015 have 15).

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 554214 6 6 6 5  653215 3 4 5 6  156212 4 2 3 4  324413 5 6 5 5  134014 5 3 4 2  END | 6 6 6 5  3 4 5 6  5 3 4 2 |

## Group by Group

Create a class **Person**. It should consists of **properties** : **name** and **group** (String, Integer). Write a program that extracts all persons (students), **grouped by** **GroupName** and then prints them on the console. Print all group names along with the students in each group. Use the **group by** Stream operations. You will be given an input on the console.

**Output format** : **{group} - {name1}, {name2}, {name3}, ...**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Ivaylo Petrov 10  Stanimir Svilianov 3  Indje Kromidov 3  Irina Balabanova 4  END | 3 - Stanimir Svilianov, Indje Kromidov  4 - Irina Balabanova  10 - Ivaylo Petrov |

## Online Radio Database

Create an online radio station database. It should keep information about all added songs. On the first line you are going to get the number of songs you are going to try adding. On the next lines you will get the songs to be added in the format <artist name>;<song name>;<minutes:seconds>. To be valid, every song should have an artist name, a song name and length.

Design a custom exception hierarchy for invalid songs:

* InvalidSongException
  + InvalidArtistNameException
  + InvalidSongNameException
  + InvalidSongLengthException
    - InvalidSongMinutesException
    - InvalidSongSecondsException

### Validation

* Artist name should be between 3 and 20 symbols.
* Song name should be between 3 and 30 symbols.
* Song length should be between 0 second and 14 minutes and 59 seconds.
* Song minutes should be between 0 and 14.
* Song seconds should be between 0 and 59.

### Exception Messages

|  |  |
| --- | --- |
| **Exception** | **Message** |
| InvalidSongException | "Invalid song." |
| InvalidArtistNameException | "Artist name should be between 3 and 20 symbols." |
| InvalidSongNameException | "Song name should be between 3 and 30 symbols." |
| InvalidSongLengthException | "Invalid song length." |
| InvalidSongMinutesException | "Song minutes should be between 0 and 14." |
| InvalidSongSecondsException | "Song seconds should be between 0 and 59." |

**Note**: Check validity in the order artist **name** -> **song name** -> **song length**

### Output

If the song is added, print "**Song added.**".

If you **can’t add a song**, print an **appropriate exception message**.

On the last two lines print the **number of songs added** and the **total length of the playlist** in format:

**"Playlist length: 0h 7m 47s"**

### Examples

|  |  |
| --- | --- |
| **Exception** | **Message** |
| 3  ABBA;Mamma Mia;3:35  Nasko Mentata;Shopskata salata;4:123  Nasko Mentata;Shopskata salata;4:12 | Song added.  Song seconds should be between 0 and 59.  Song added.  Songs added: 2  Playlist length: 0h 7m 47s |
| 5  Nasko Mentata;Shopskata salata;14:59  Nasko Mentata;Shopskata salata;14:59  Nasko Mentata;Shopskata salata;14:59  Nasko Mentata;Shopskata salata;14:59  Nasko Mentata;Shopskata salata;0:5 | Song added.  Song added.  Song added.  Song added.  Song added.  Songs added: 5  Playlist length: 1h 0m 1s |