* **Use of Comparable is mandated.** Write a program that takes program arguments and stores them into a collection. Duplicates are preserved. The program arguments supplied are expected to be in the format – *firstName,lastName*. Ignore any arguments that are not in the expected format.

It then has to print them out sorted according to the *firstName*. If multiple elements have same *firstName* then they should be sorted according to their *lastName*.

For instance, the execution of - "java <ClassName> Mathews,Lyon George,Greesham Stephen George,Fin Bravo,Carl George,Greesham" should result in

Bravo,Carl

George,Fin

George,Greesham

George,Greesham

Mathews,Lyon

* **Use of Comparator is mandated.** The description and the expected output is same as that of exercise no.1.
* **Use of Comparable is mandated.** Same as exercise no.1 except that the duplicates should be avoided. Expected output is -

Bravo,Carl

George,Fin

George,Greesham

Mathews,Lyon

* **Use of Comparator is mandated.** Same as exercise no.3
* Write a program that takes program arguments and stores them into a collection. The program arguments supplied are expected to be in the format – *firstName,lastName*. Ignore any arguments that are not in the expected format.

It then has to print them out sorted according to the *firstName* along with the number of times each element has occurred. If multiple elements have same *firstName* then they should be sorted according to their *lastName*.

For instance, the execution of - "java <ClassName> Mathews,Lyon George,Greesham Stephen George,Fin Bravo,Carl George,Greesham" should result in

Bravo,Carl - 1

George,Fin -1

George,Greesham - 2

Mathews,Lyon - 1