* Write a program that takes program arguments and stores them into a collection. It then has to print them out along with their index (1-based).

For instance the execution of - "java <ClassName> Mathews George Stephen George Bravo" should result in

Mathews – 1

George - 2

Stephen - 3

George - 4

Bravo - 5

* Write a program that takes program arguments and stores them into a collection. It should be able to avoid duplicates and print them out. The order of printing isn't necessary.

For instance the execution of - "java <ClassName> Mathew George Stephen George Bravo" may result in

Bravo

Stephen

Mathews

George

* Write a program that takes program arguments and stores them into a collection. It should be able to avoid duplicates and print them out in lexicographic order.

For instance the execution of - "java <ClassName> Mathews George Stephen George Bravo" should result in

Bravo

George

Mathews

Stephen

* Write a program that takes program arguments and stores them into a collection. It should be able to avoid duplicates and print them out in the order taken.

For instance the execution of - "java <ClassName> Mathews George Stephen George Bravo" should result in

Mathews

George

Stephen

Bravo

* Write a program that takes program arguments and stores them into a collection. It should print them out along with the number of times each element has occurred. The order of printing isn't necessary.

For instance the execution of - "java <ClassName> Mathews George Stephen George Bravo" may result in

Bravo - 1

Stephen - 1

Mathews - 1

George – 2

* Write a program that takes program arguments and stores them into a collection. It should print them out along with the number of times each element has occurred. The order of printing should be in the order taken.

For instance the execution of - "java <ClassName> Mathews George Stephen George Bravo" should result in

Mathews - 1

George - 2

Stephen - 1

Bravo - 1

* Write a program that takes program arguments and stores them into a collection. It should print them out along with the number of times each element has occurred. The order of printing should be lexicographic.

For instance the execution of - "java <ClassName> Mathews George Stephen George Bravo" should result in

Bravo - 1

George - 2

Mathews - 1

Stephen – 1

* Code an Employee class such that the below program, when executed, gives the expected output.

add(e2);

System.*out*.println(employeeSet.contains(**new** Employee("Adam Brian",**null**)));

System.*out*.println(employeeSet.contains(**new** Employee("Charles Matschua",ceo)));

System.*out*.println(employeeSet.contains(**new** **package** com.assignments.collections.part\_one;

**import** java.util.HashSet;

**import** java.util.Set;

**import** com.assignments.collections.part\_one.dto.Employee;

**public** **class** Tester

{

**public** **static** **void** main(String[] args){

**new** Tester().start();

}

**private** **void** start(){

Employee ceo = **new** Employee("Adam Brian", **null**);

Employee e1 = **new** Employee("Charles Matschua",ceo);

Employee e2 = **new** Employee("David Cynthia",ceo);

Set<Employee> employeeSet = **new** HashSet<Employee>();

employeeSet.add(ceo);

employeeSet.add(e1);

employeeSet.Employee("Matschua Charles",ceo)));

System.*out*.println(employeeSet.contains(**new** Employee("Charles Matschua",**null**)));

System.*out*.println(employeeSet.contains(**new** Employee("Charles Matschua",e2)));

System.*out*.println(employeeSet.contains(e2));

}

}

**Expected Output**

true

true

false

false

false

true