**Day 7 – Assignment**

**Pratik K Kamble**

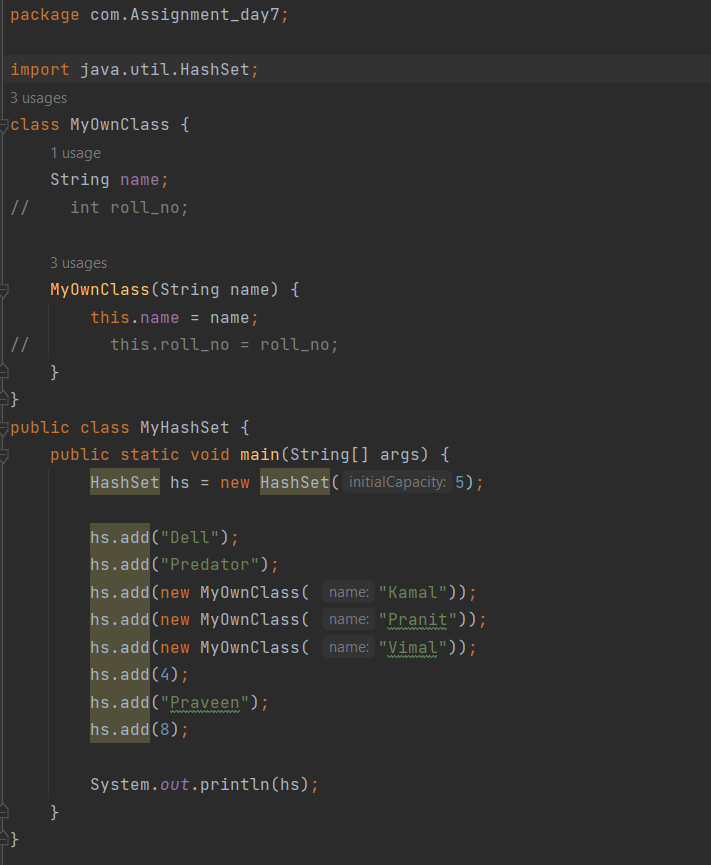
**Employee ID : 46263548**

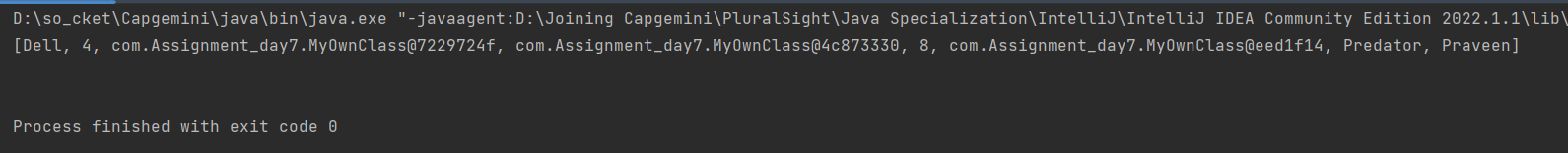
**18 – 09 – 2022**

**Q1**: For your own exercise, do the following.

* Create your own eclipse project as MyHashSet
* Create your own HashSet object with initial capacity of 5
* Add the following objects to the newly created HashSet object o 2 String Objects o 3 MyOwnClass Object (you will have to first create the MyOwnClass.java class) o 3 Integer Objects
* Display the Hash set object
* Try to add same object once again and see the behaviour.
* Observe the behaviour of HashSet working.

**Solution –**

****

****

**Q2:** You have a list of names, your job is to find out which word or words are duplicate, and show the list of distinct names. Follow the below code and complete the code.

**publicclass**Set\_HashSetFindDuplicate {

**publicstaticvoid** main(String[] args) {

// set up test data String names[]={  **new** String("pankaj"),  **new** String("rajesh"),

**new** String("suresh"),

**new** String("pankaj")

};

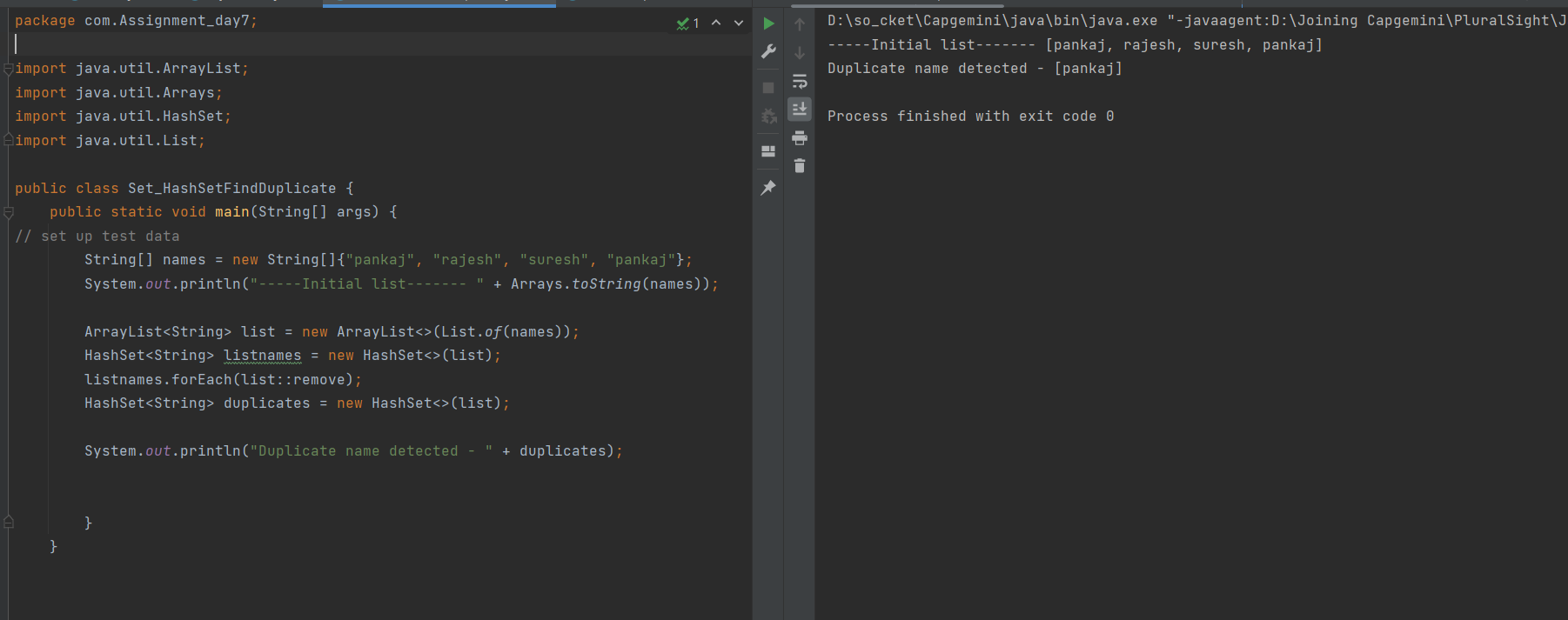
\Output should be as below. }

}

**-----Initial list-------** pankaj, rajesh, suresh, pankaj **Duplicate name detected :pankaj**

*3 distinct words detected :  list : [suresh, pankaj, rajesh]*

**Solution –**

****

**Q3:** Modify the above app, and name it as Set\_HashSetFindDuplicate2 .In this app you need to show the list of unique names and show which names are duplicate. Find the below code and add your logic.

**publicclass** Set\_HashSetFindDuplicate2 {

**publicstaticvoid** main(String[] args) {  // set up test data

  String names[]={  **new** String("pankaj"),  **new** String("rajesh"),         **newnew**  String(String("suresh""pankaj")),,           **newnew**  String(String("suresh""aman") ),

  };

}

  }

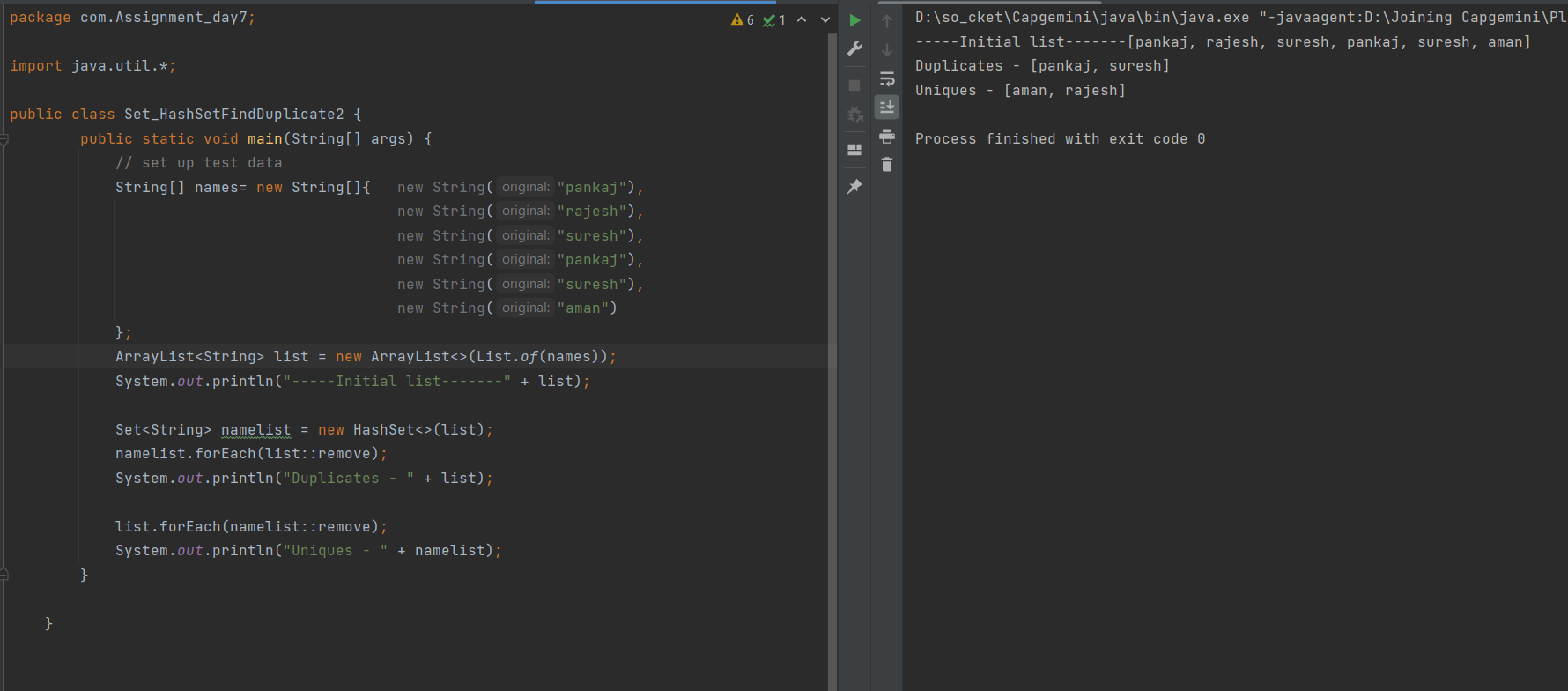
Your output should be as below.

-----Initial list-------

pankaj, rajesh, suresh, pankaj, suresh, aman

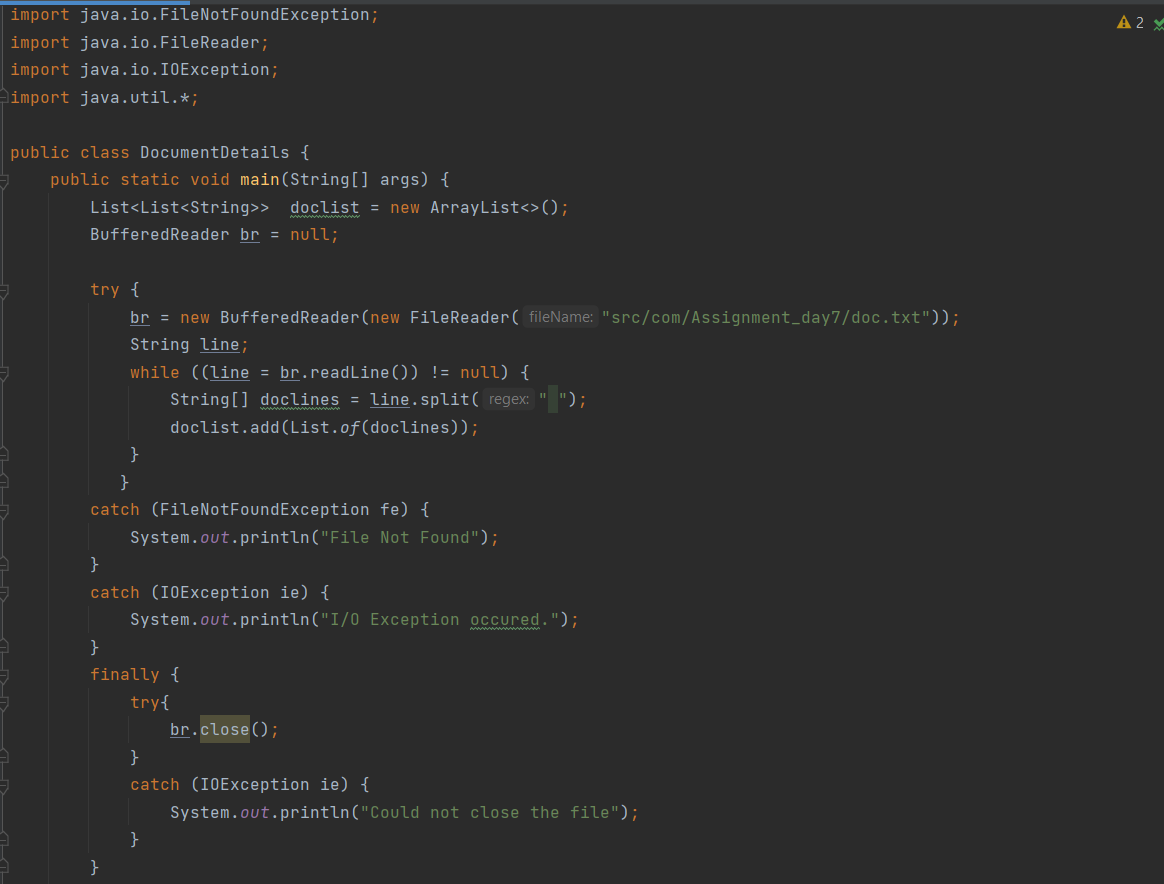
unique names : [aman, rajesh] duplicate names : [suresh, pankaj]

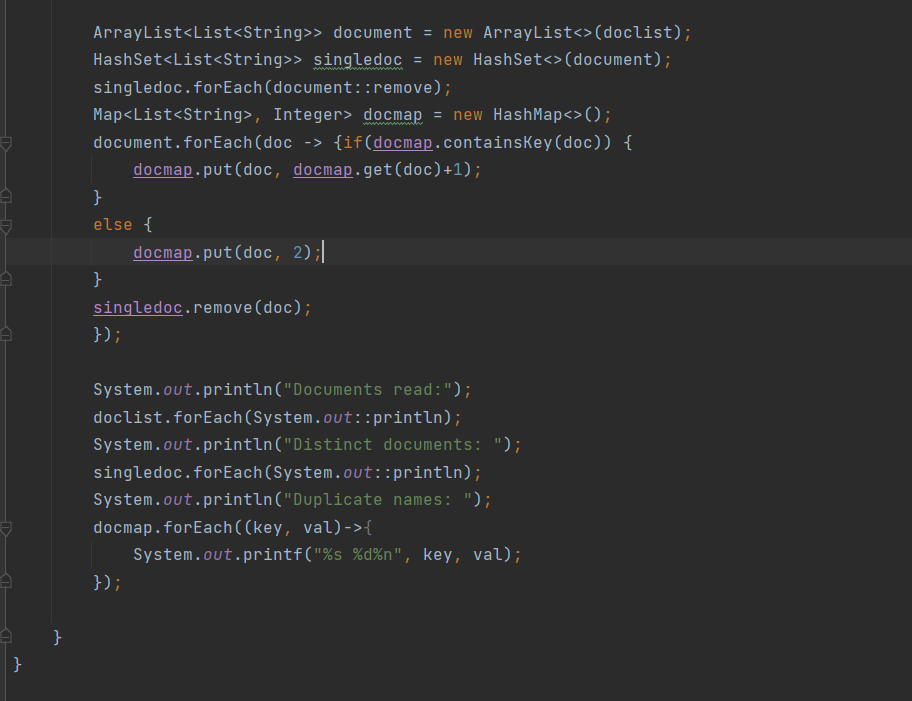
**Solution –**

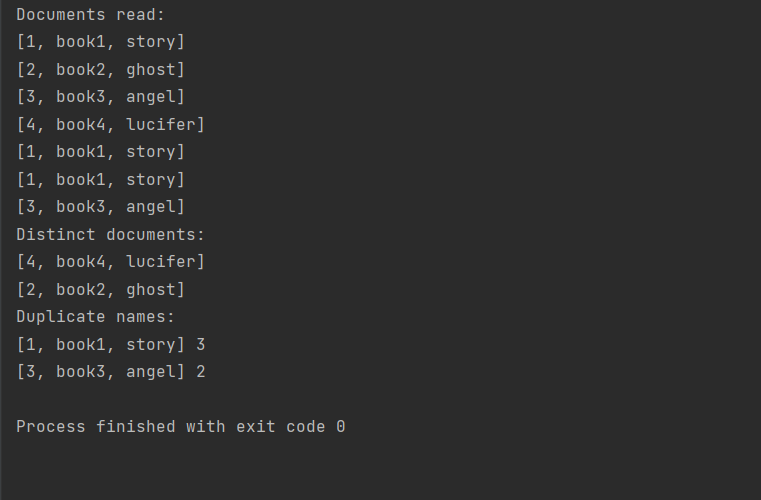
****

**Q4:** Create an application to add document detail as (id, title, description). This detail should be added permanently in file system. Your application should read all the documents from the docs.txt file and show separately how many documents are duplicated for how many number of times, and show the list of distinct documents.

**Solutions –**

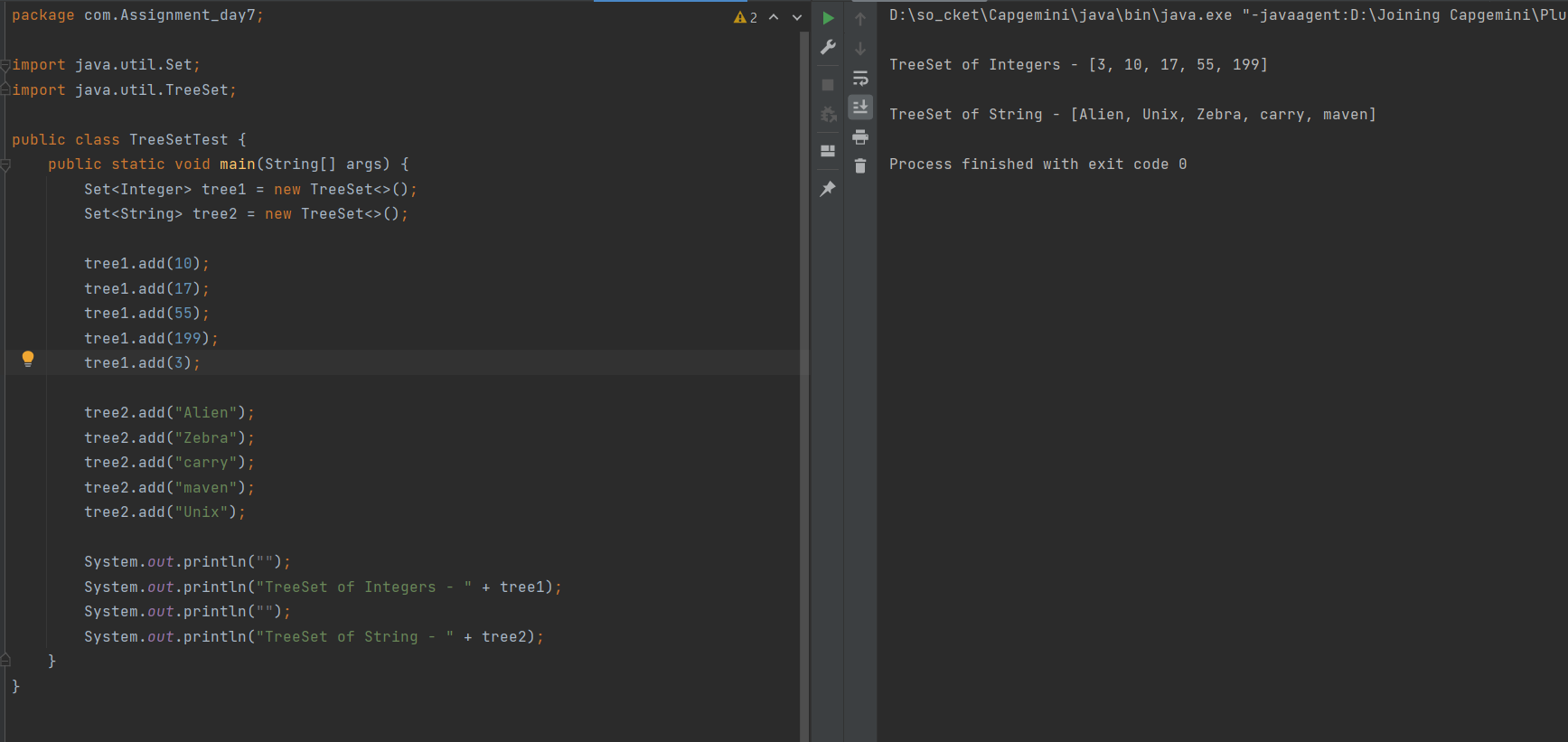






**Q5:** Create two TreeSet objects and put them in two set references, one object will hold the string type value and other object will hold the integer type values. Observe the behaviour of adding values in TreeSet. Come up with solution how it is happening?

**Solution –**

****

**Q6:** You need to find out the frequency of the word repeated in an array of names. Attempt it using Map. You can use the below skeleton for your reference.

**package** map;

**import**java.util.HashMap; **import**java.util.Map; **publicclass**Map\_HashMap {

**privatestaticfinal** Integer ***ONE***=**new**Integer(1);

**publicstaticvoid** main(String[] args) {

  // Set up testing data

        String name[] = { **new**

 String("pankaj"), **new** String("rajesh"), **new** String("pankaj"), **new** String("deepak"), **new** String("pankaj")         };

}

}

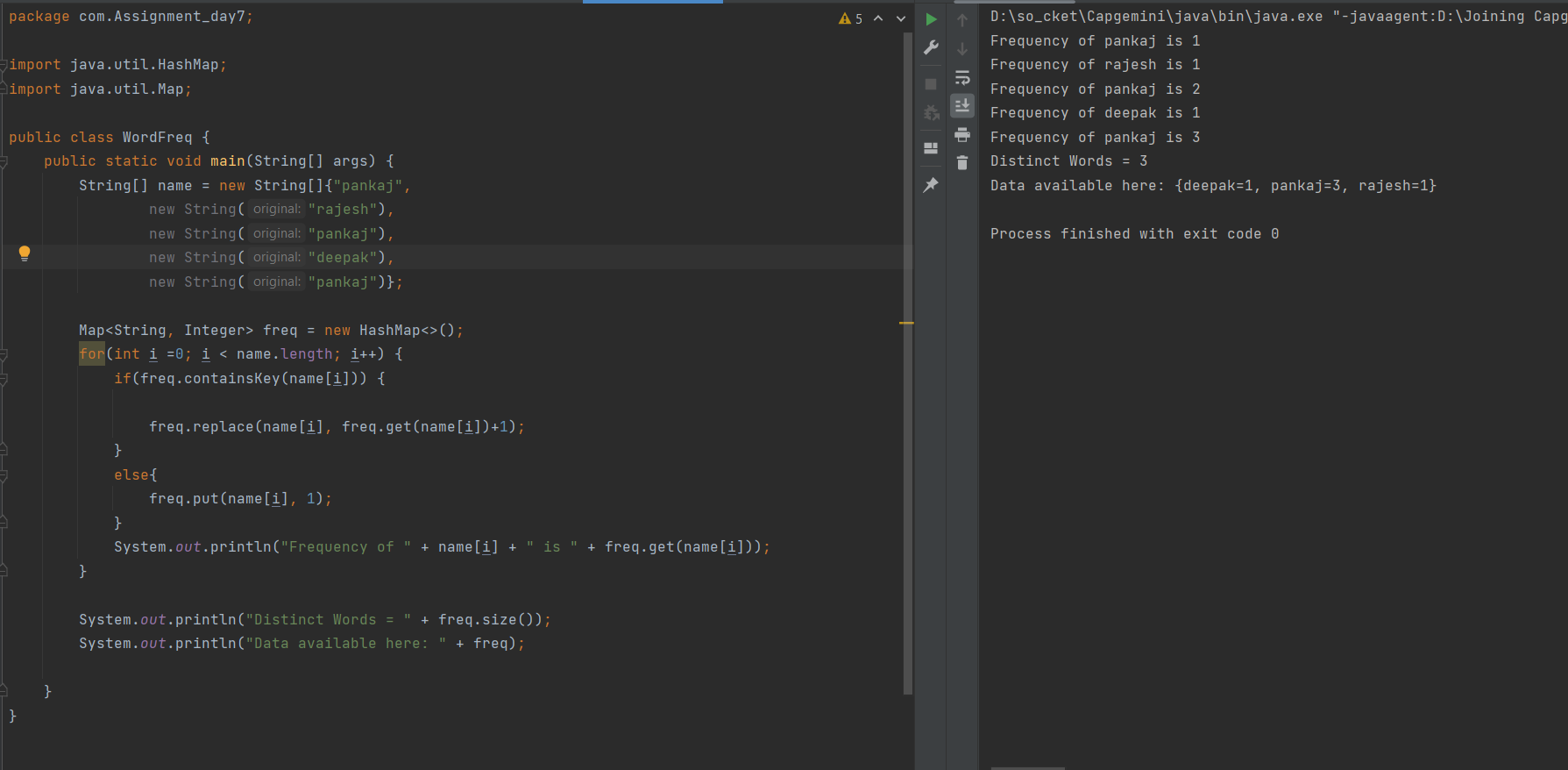
Output should be as below

frequency of :pankaj is : 1 frequency of :rajesh is : 1 frequency of :pankaj is : 2 frequency of :deepak is : 1 frequency of :pankaj is : 3

3 distinct word detected

Display of HashMap object : {deepak=1, pankaj=3, rajesh=1}

**Solution –**

****

**Q7:** Creating your own collection.

In this exercise you need to create your own ArrayList. You are not suppose to use the in build methods of the ArrayList. For the specified operation you need to create your own logic. But it should work same as ArrayList.

For this exercise some setup activity is as below.

Create Document with below details

* name
* purpose
* validity
* showDocumentDetail

* You may have different type of documents.like Personal, Official, Confidential etc.
* Create a list of documents and display the report of each type of documents. (User your List Interface and Your ArrayList Class)

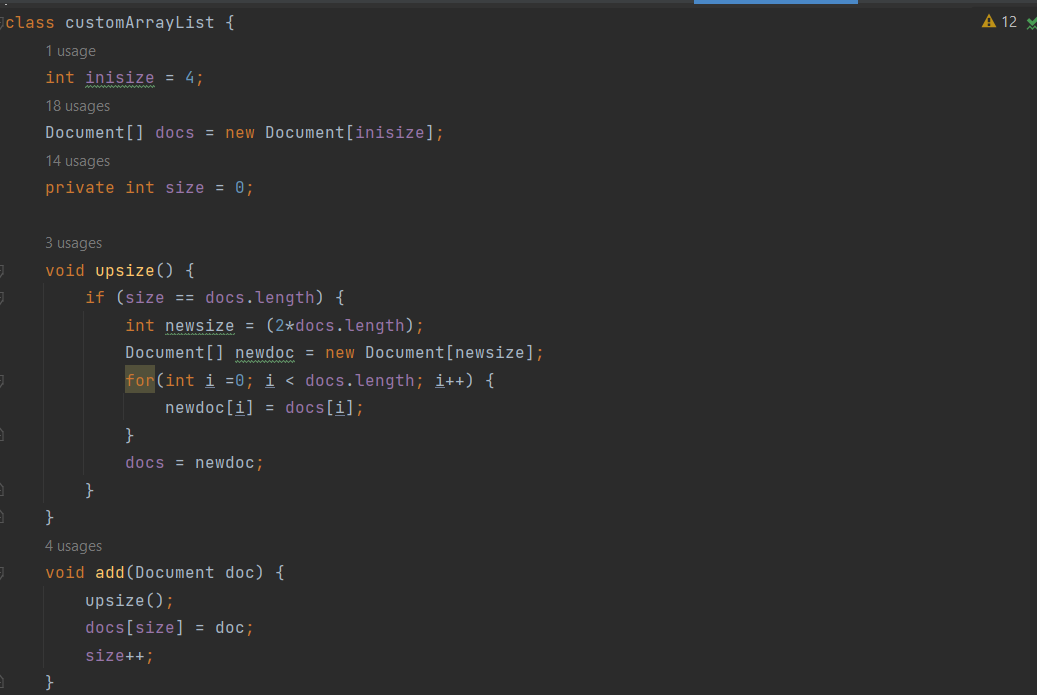
Limitations

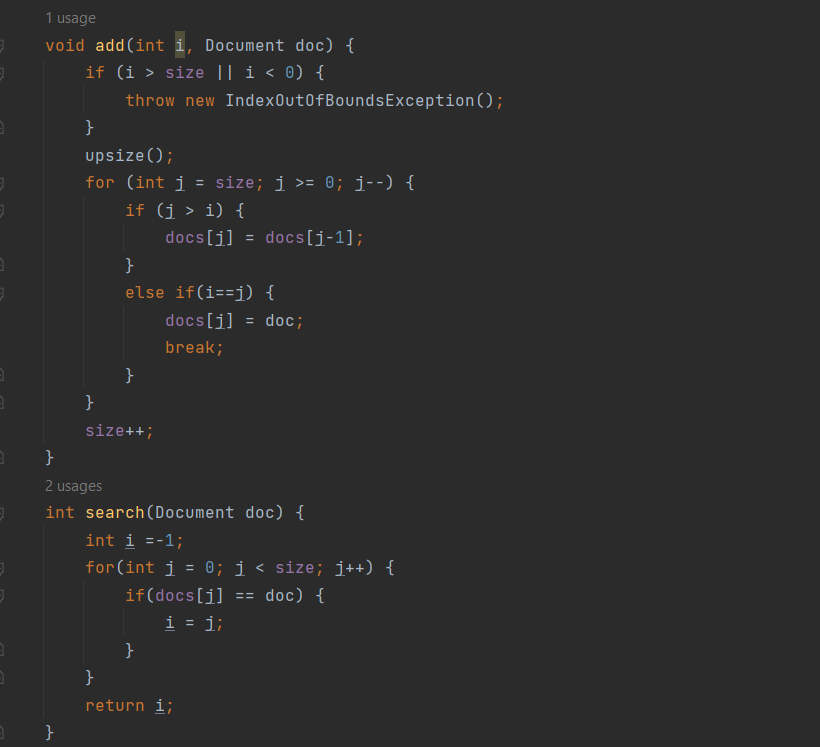
* list should be used from your own collection api, not from java.util.
* list should accept only Documents; other types should not be allowed.

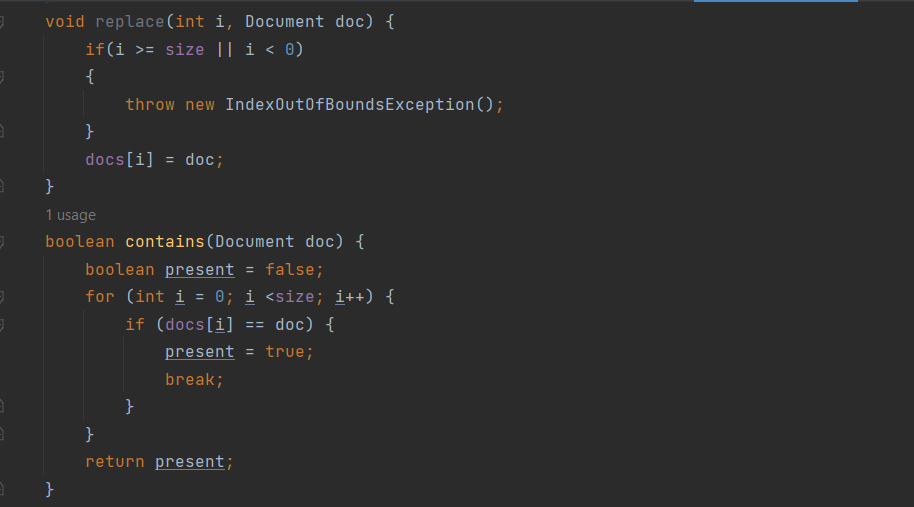
Perform below operations

* Adding
* Searching
* Removing
* Replacing
* Find total documents

**Solution –**

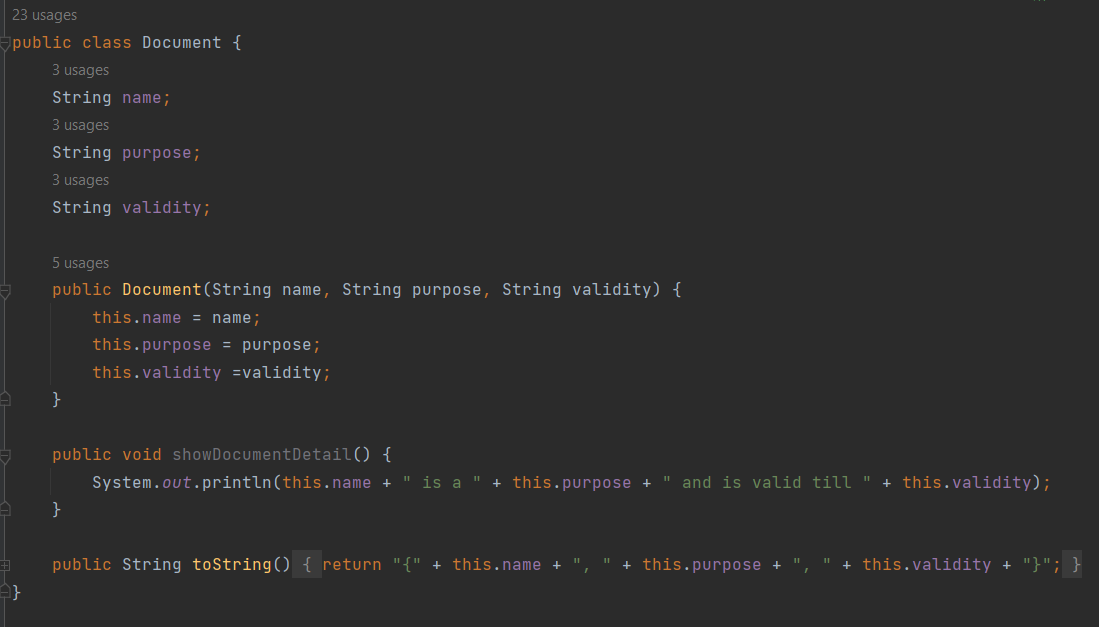
****

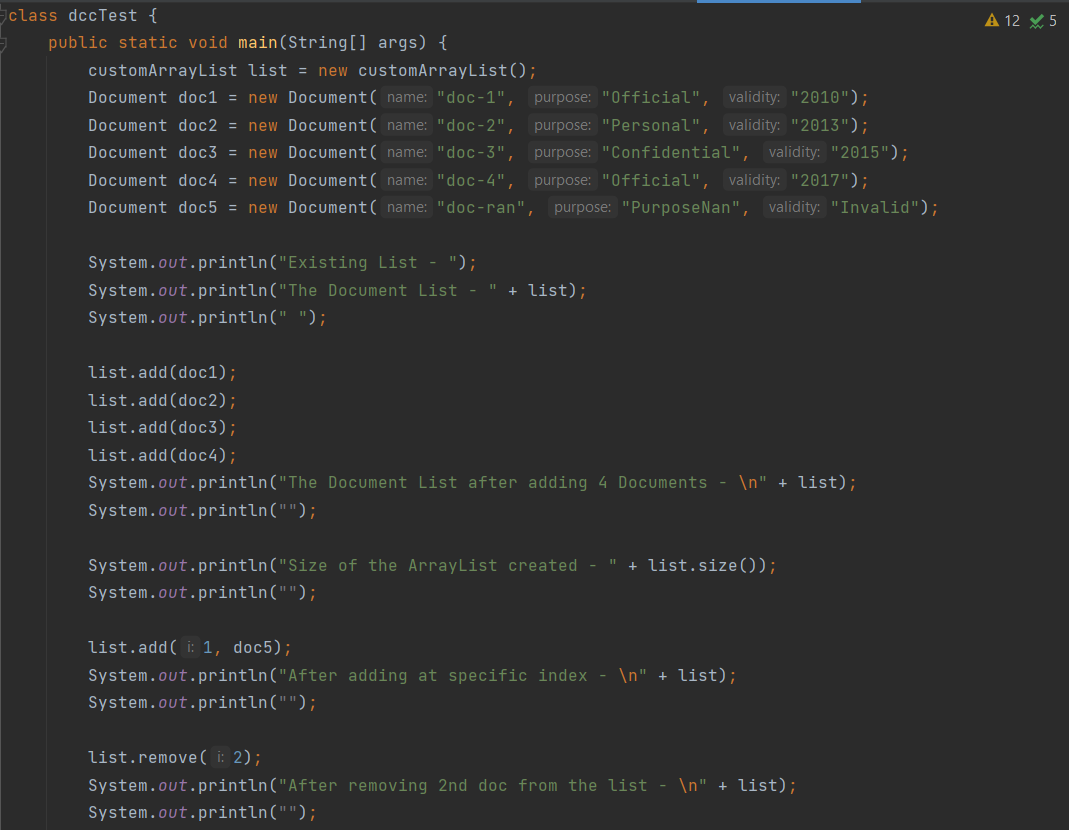
****

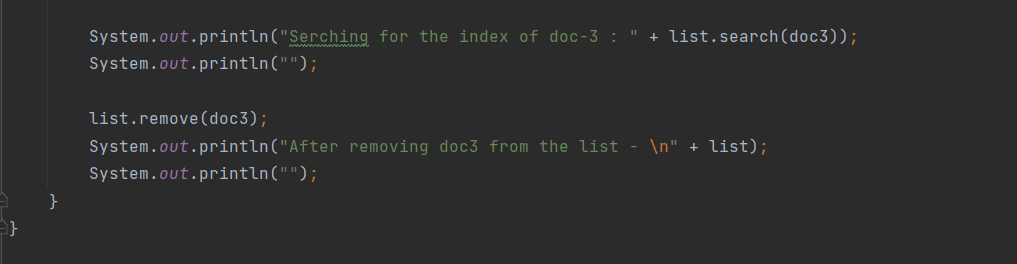
****

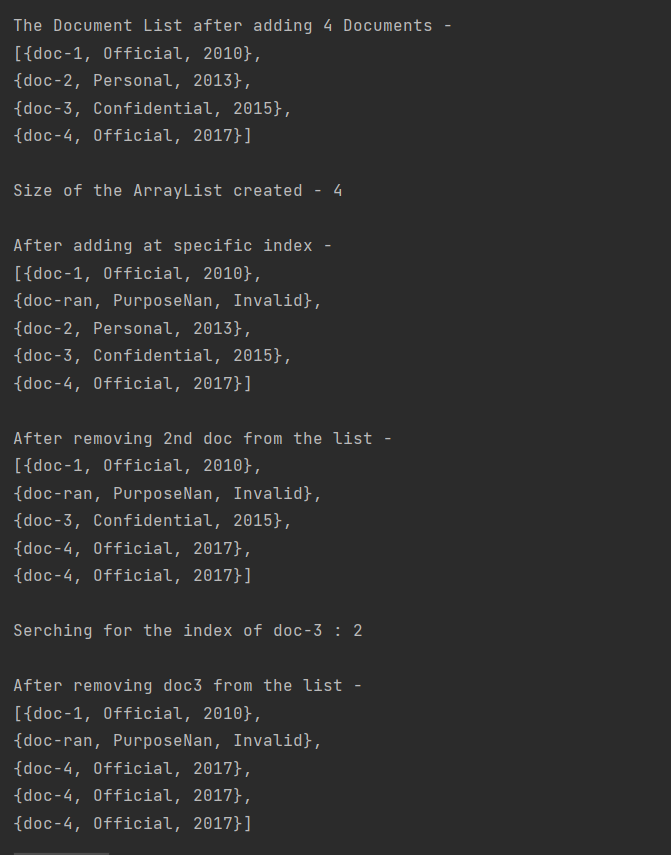
****

****

****

****

****

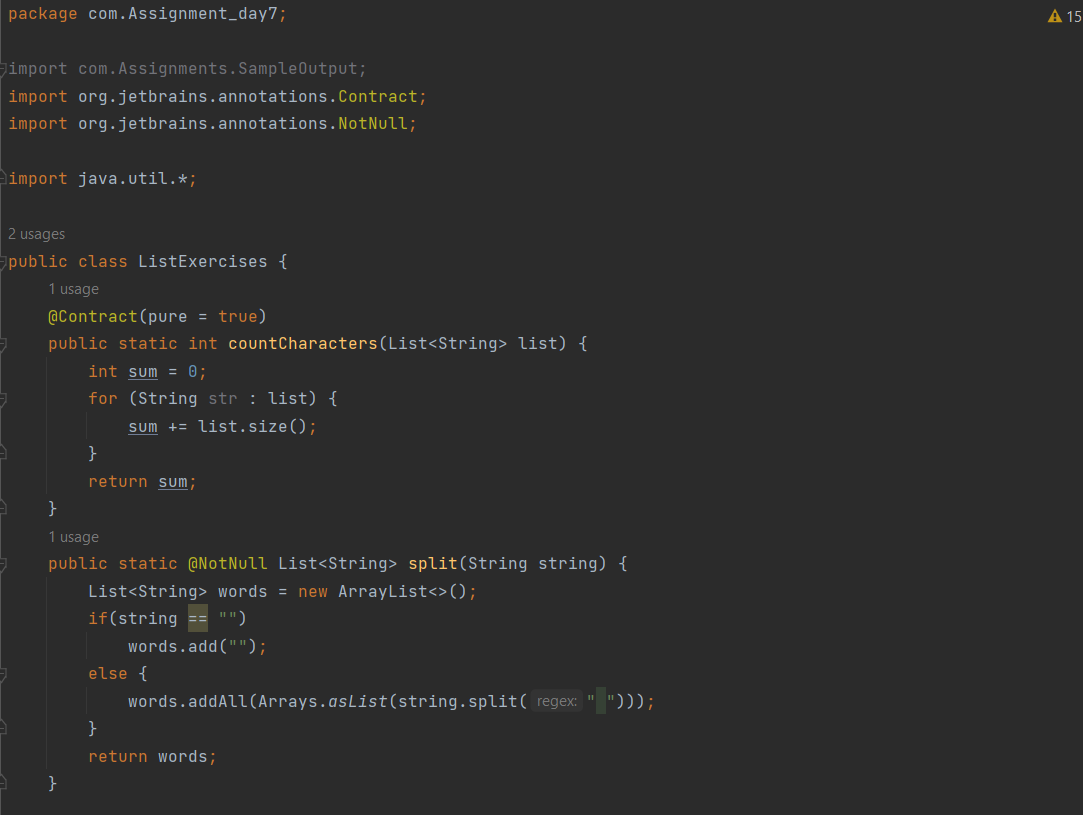
****

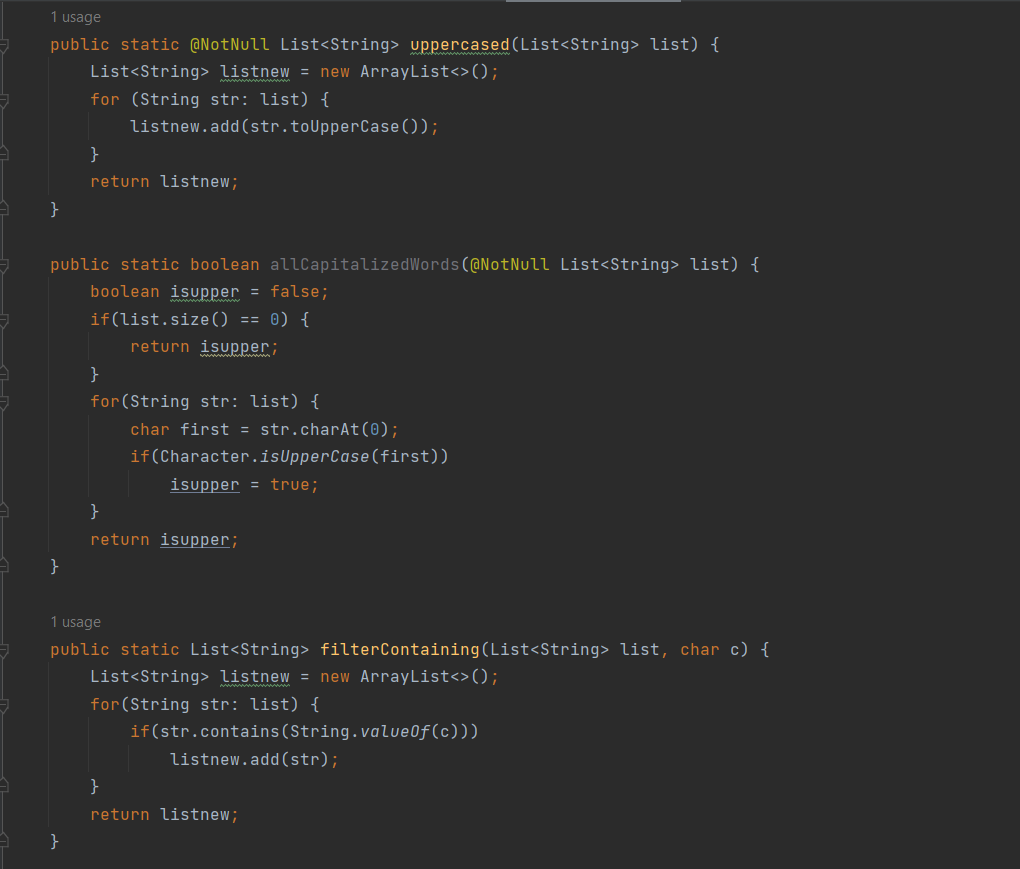
**Q8:** Create ListExercises class with below methods. Method description is provided. You need to create a main class to test all the functionalities of the ListExercises.

|  |  |
| --- | --- |
| Class Name | ListExercises |
| public static int countCharacters(List<String> list) {  return 0;  } | Counts the number of characters in total across all strings in the supplied list. In other words, the sum of the lengths of the all the strings. list is a non null list of string. return the number of characters. |
| public static List<String> split(String string) {  return null;  } | Splits a string into words and returns a list of the words. If the string is empty, split returns a list containing an empty string. string a non-null string of zero or more words return a list of words |
| public static List<String> uppercased(List<String> list) { return null; } | Returns a copy of the list of strings where each string has been uppercased (as by String.toUpperCase).The original string is unchanged. List is a non null list of string return a list of uppercased strings |
| public static boolean  allCapitalizedWords(List<String> list) {  return false;  } | Returns true if and only if each string in the supplied list of strings starts with an uppercase letter. If the list is empty, returns false. list is a non null list of string return true if each string starts with an uppercase letter |
| public static List<String>  filterContaining(List<String> list, char c) {  return null;  } | Returns a list of strings selected from a supplied list, which contain the character c. The returned list is in the same order as the original list, but it omits all strings that do not contain c. The original list is unmodified. list is a non null list of string c the character to filter on return a list of strings containing the character c, selected from list. |
| public static void insertInOrder(String string,  List<String> list) {  } | Inserts a string into a sorted list of strings, maintaining the sorted property of the list. string is the string to insert list is a non null sorted list of strings |

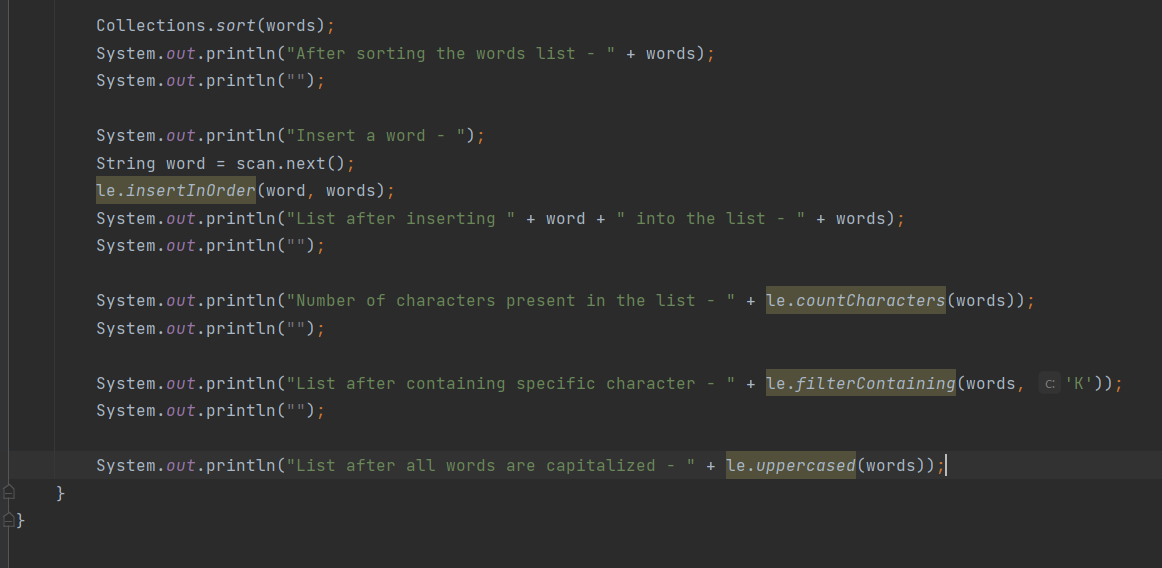
|  |  |
| --- | --- |
| Class Name | ListExercisesTest |
| public static void main(String [] args){ //TODO  : Your testing code goes here  } | Use main method to test your ListExercises class’s methods |

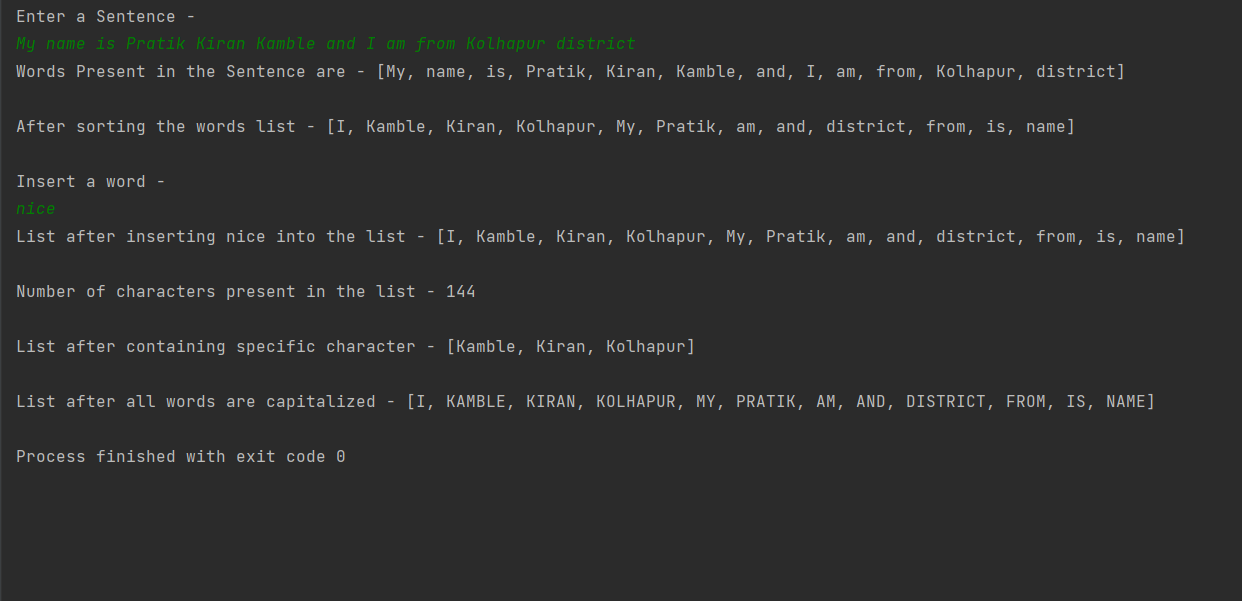
**Solution –**

****

****

****

****

****