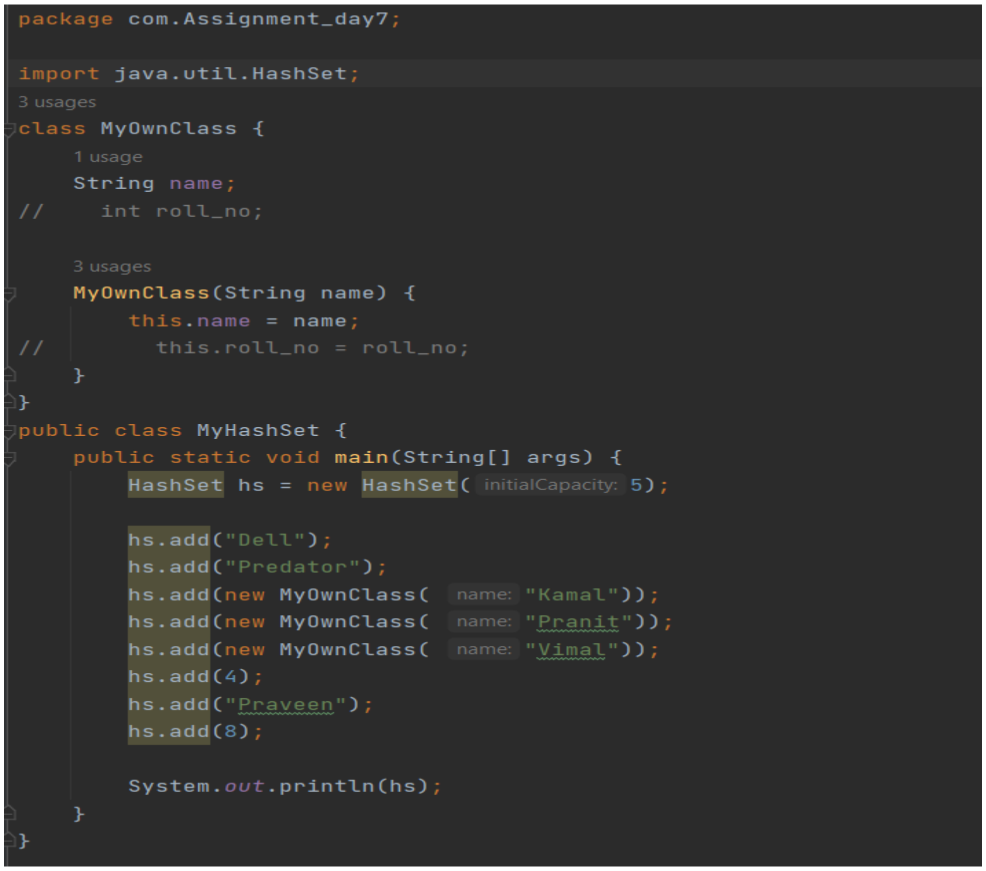
Day 7 – Assignment

Pratik K Kamble
Employee ID : 46263548
18 – 09 – 2022

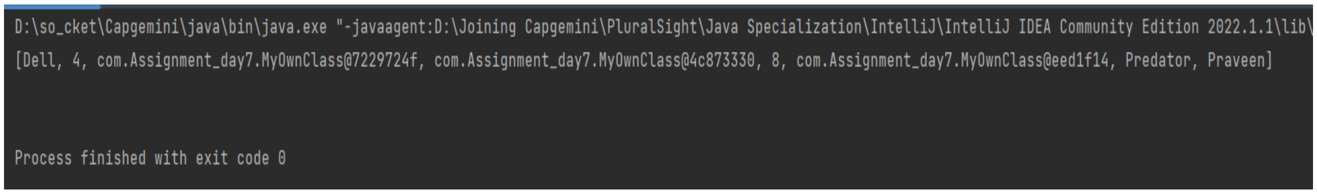
Q1: For your own exercise, do the following.

- 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 ○ 2 String Objects ○ 3 MyOwnClass Object (you will have to first create the MyOwnClass.java class) ○ 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 –



```
package com.Assignment_day7;

import java.util.HashSet;
3 usages
class MyOwnClass {
    1 usage
    String name;
//    int roll_no;

    3 usages
    MyOwnClass(String name) {
        this.name = name;
//        this.roll_no = roll_no;
    }
}
public class MyHashSet {
    public static void main(String[] args) {
        HashSet hs = new HashSet( initialCapacity: 5);

        hs.add("Dell");
        hs.add("Predator");
        hs.add(new MyOwnClass( name: "Kamal"));
        hs.add(new MyOwnClass( name: "Pranit"));
        hs.add(new MyOwnClass( name: "Vimal"));
        hs.add(4);
        hs.add("Praveen");
        hs.add(8);

        System.out.println(hs);
    }
}
```



```
D:\so_cket\Capgemini\java\bin\java.exe "-javaagent:D:\Joining Capgemini\PluralSight\Java Specialization\IntelliJ\IntelliJ IDEA Community Edition 2022.1.1\lib\
[Dell, 4, com.Assignment_day7.MyOwnClass@7229724f, com.Assignment_day7.MyOwnClass@4c873330, 8, com.Assignment_day7.MyOwnClass@eed1f14, Predator, Praveen]


Process finished with exit code 0
```

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.

```
publicclassSet_HashSetFindDuplicate {
 publicstaticvoid main(String[] args) {
              // set up test data String names[]={  new String("pankaj"),  new String("rajesh"),
                                                                    new String("suresh"),
 new String("pankaj")
              };
```

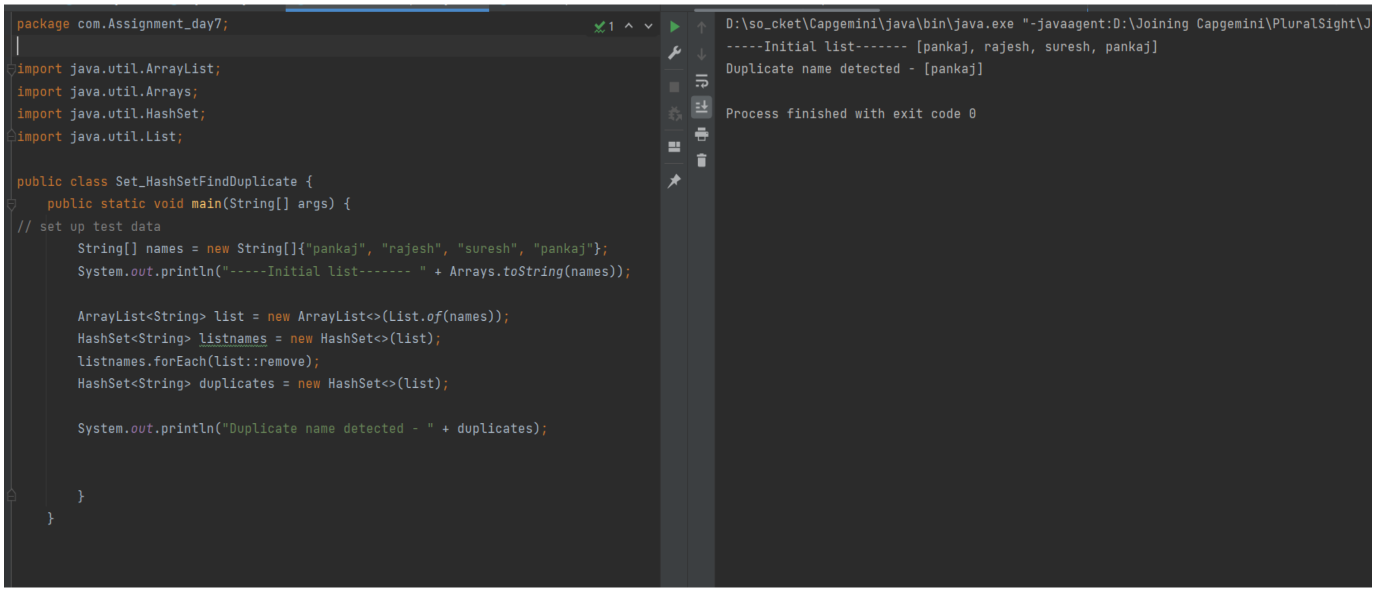
\Output should be as below. }

}

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

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

Solution –



```
package com.Assignment_day7;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashSet;
import java.util.List;

public class Set_HashSetFindDuplicate {
    public static void main(String[] args) {
// set up test data
        String[] names = new String[]{"pankaj", "rajesh", "suresh", "pankaj"};
        System.out.println("-----Initial list------- " + Arrays.toString(names));

        ArrayList<String> list = new ArrayList<>(List.of(names));
        HashSet<String> listnames = new HashSet<>(list);
        listnames.forEach(list::remove);
        HashSet<String> duplicates = new HashSet<>(list);

        System.out.println("Duplicate name detected - " + duplicates);


        }
    }
```

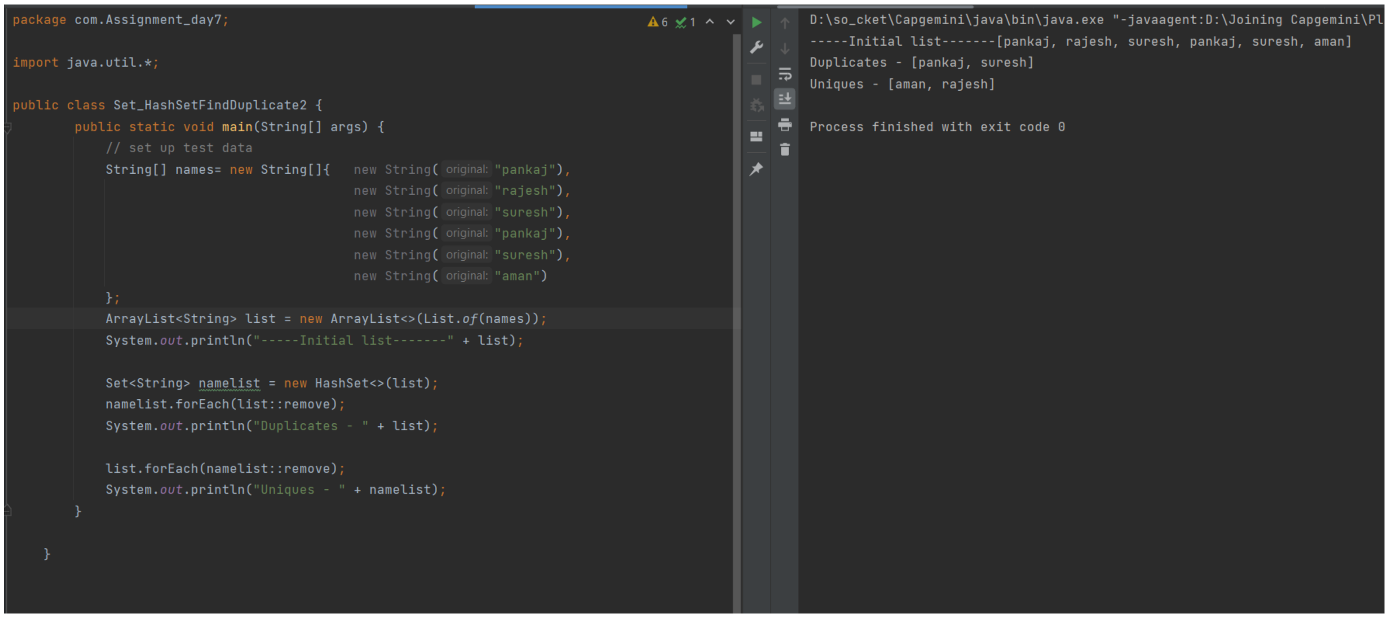
```
D:\so_cket\Capgemini\java\bin\java.exe "-javaagent:D:\Joining Capgemini\PluralSight\J
-----Initial list------- [pankaj, rajesh, suresh, pankaj]
Duplicate name detected - [pankaj]

Process finished with exit code 0
```

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""a
man") ),
 };
         }

 }
```

Your output should be as below.
-----Initial list-------
pankaj, rajesh, suresh, pankaj, suresh, aman
unique names : [aman, rajesh] duplicate
names : [suresh, pankaj]

Solution –

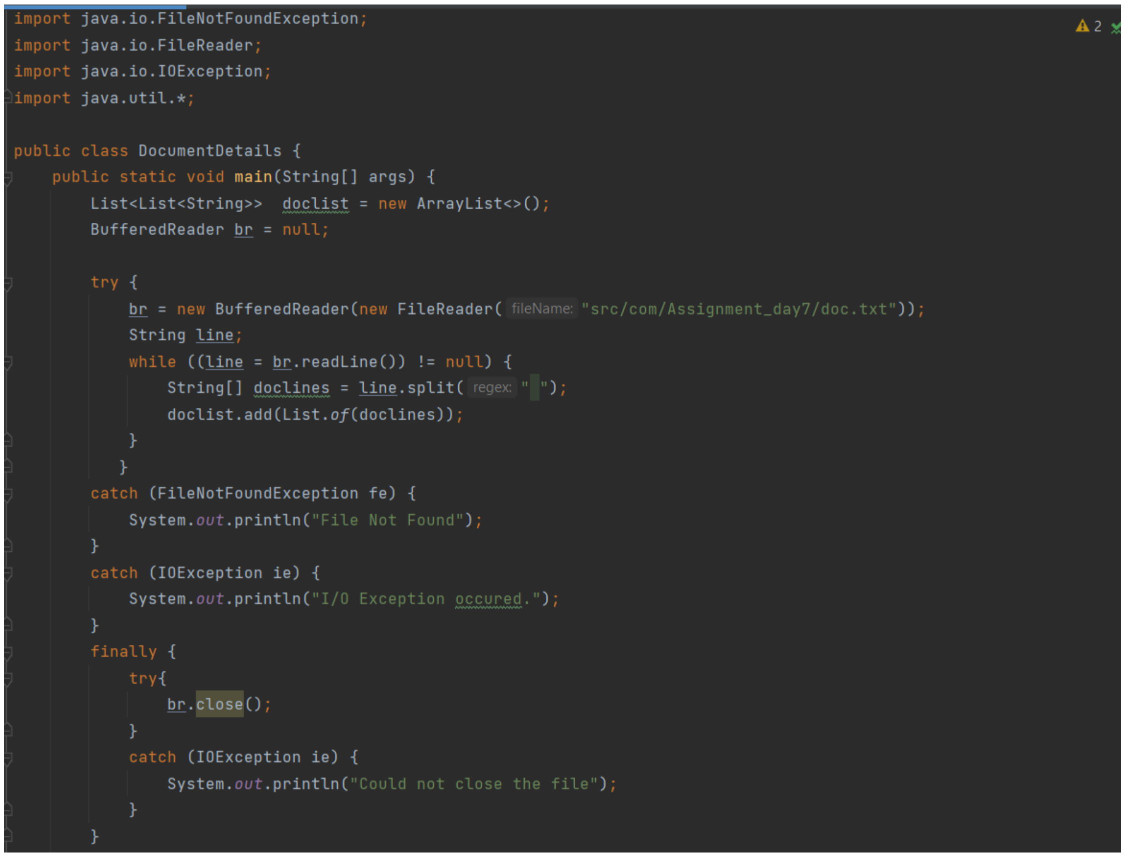


```java
package com.Assignment_day7;

import java.util.*;

public class Set_HashSetFindDuplicate2 {
        public static void main(String[] args) {
            // set up test data
            String[] names= new String[]{   new String( original: "pankaj"),
                                            new String( original: "rajesh"),
                                            new String( original: "suresh"),
                                            new String( original: "pankaj"),
                                            new String( original: "suresh"),
                                            new String( original: "aman")
            };
            ArrayList<String> list = new ArrayList<>(List.of(names));
            System.out.println("-----Initial list-------" + list);

            Set<String> namelist = new HashSet<>(list);
            namelist.forEach(list::remove);
            System.out.println("Duplicates - " + list);

            list.forEach(namelist::remove);
            System.out.println("Uniques - " + namelist);
        }

    }
```

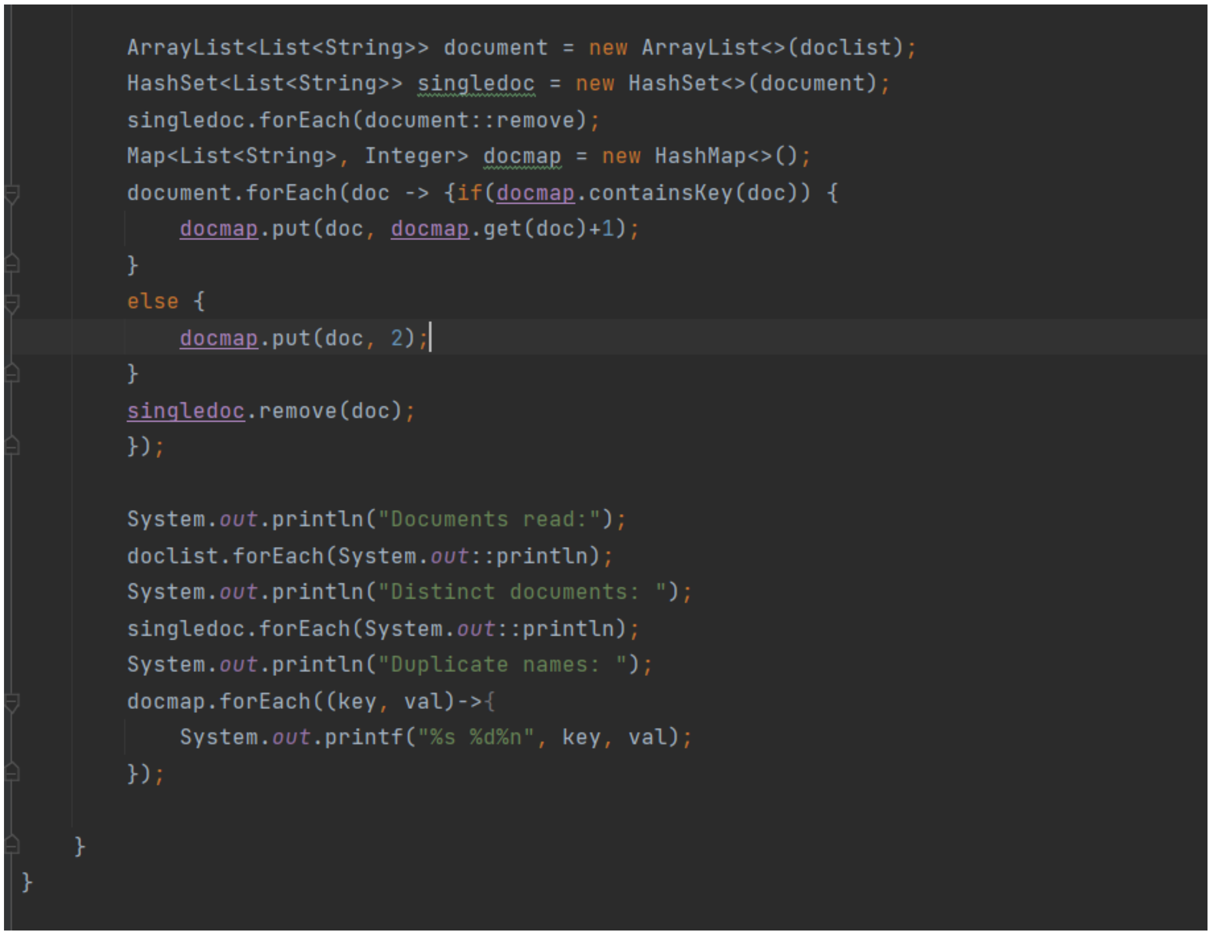
```
D:\so_cket\Capgemini\java\bin\java.exe "-javaagent:D:\Joining Capgemini\Pl
-----Initial list-------[pankaj, rajesh, suresh, pankaj, suresh, aman]
Duplicates - [pankaj, suresh]
Uniques - [aman, rajesh]

Process finished with exit code 0
```

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.

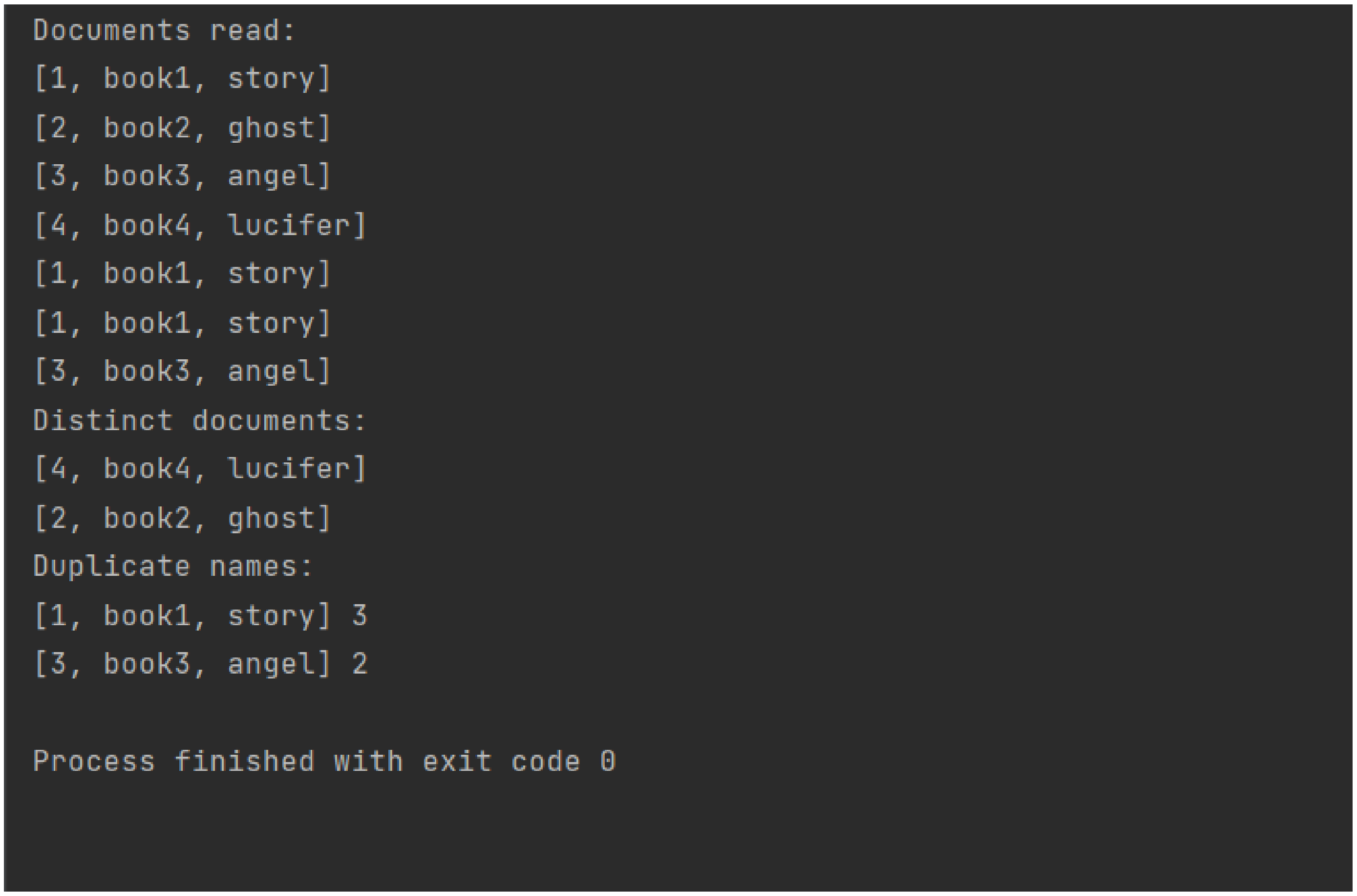
Solutions –



```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.*;

public class DocumentDetails {
    public static void main(String[] args) {
        List<List<String>>  doclist = new ArrayList<>();
        BufferedReader br = null;

        try {
            br = new BufferedReader(new FileReader( fileName: "src/com/Assignment_day7/doc.txt"));
            String line;
            while ((line = br.readLine()) != null) {
                String[] doclines = line.split( regex: " ");
                doclist.add(List.of(doclines));
            }
           }
        catch (FileNotFoundException fe) {
            System.out.println("File Not Found");
        }
        catch (IOException ie) {
            System.out.println("I/O Exception occured.");
        }
        finally {
            try{
                br.close();
            }
            catch (IOException ie) {
                System.out.println("Could not close the file");
            }
        }
```



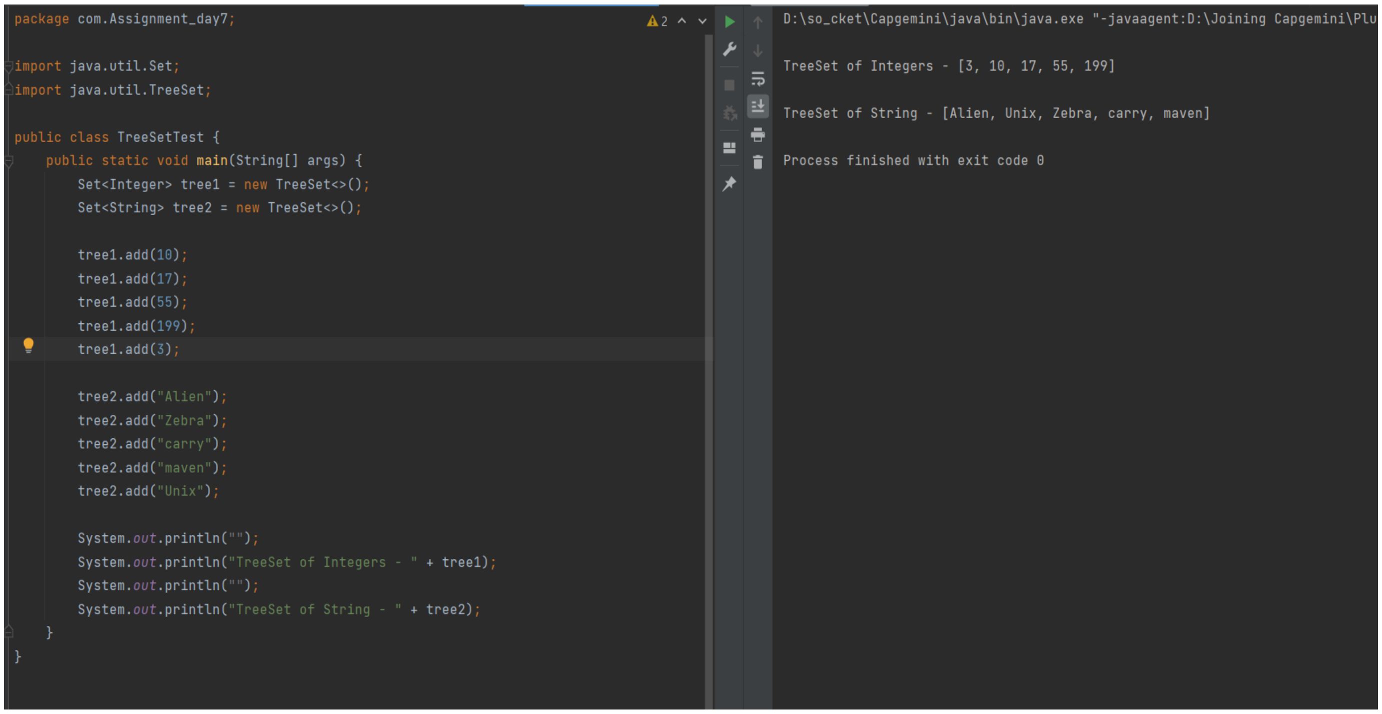
```java
        ArrayList<List<String>> document = new ArrayList<>(doclist);
        HashSet<List<String>> singledoc = new HashSet<>(document);
        singledoc.forEach(document::remove);
        Map<List<String>, Integer> docmap = new HashMap<>();
        document.forEach(doc -> {if(docmap.containsKey(doc)) {
            docmap.put(doc, docmap.get(doc)+1);
        }
        else {
            docmap.put(doc, 2);
        }
        singledoc.remove(doc);
        });

        System.out.println("Documents read:");
        doclist.forEach(System.out::println);
        System.out.println("Distinct documents: ");
        singledoc.forEach(System.out::println);
        System.out.println("Duplicate names: ");
        docmap.forEach((key, val)->{
            System.out.printf("%s %d%n", key, val);
        });

    }
}
```



```
Documents read:
[1, book1, story]
[2, book2, ghost]
[3, book3, angel]
[4, book4, lucifer]
[1, book1, story]
[1, book1, story]
[3, book3, angel]
Distinct documents:
[4, book4, lucifer]
[2, book2, ghost]
Duplicate names:
[1, book1, story] 3
[3, book3, angel] 2

Process finished with exit code 0
```

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 –



```java
package com.Assignment_day7;

import java.util.Set;
import java.util.TreeSet;

public class TreeSetTest {
    public static void main(String[] args) {
        Set<Integer> tree1 = new TreeSet<>();
        Set<String> tree2 = new TreeSet<>();

        tree1.add(10);
        tree1.add(17);
        tree1.add(55);
        tree1.add(199);
        tree1.add(3);

        tree2.add("Alien");
        tree2.add("Zebra");
        tree2.add("carry");
        tree2.add("maven");
        tree2.add("Unix");

        System.out.println("");
        System.out.println("TreeSet of Integers - " + tree1);
        System.out.println("");
        System.out.println("TreeSet of String - " + tree2);
    }
}
```

```
D:\so_cket\Capgemini\java\bin\java.exe "-javaagent:D:\Joining Capgemini\Plu

TreeSet of Integers - [3, 10, 17, 55, 199]

TreeSet of String - [Alien, Unix, Zebra, carry, maven]

Process finished with exit code 0
```

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;
importjava.util.HashMap; importjava.util.Map;
publicclassMap_HashMap {
  privatestaticfinal Integer ONE=newInteger(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

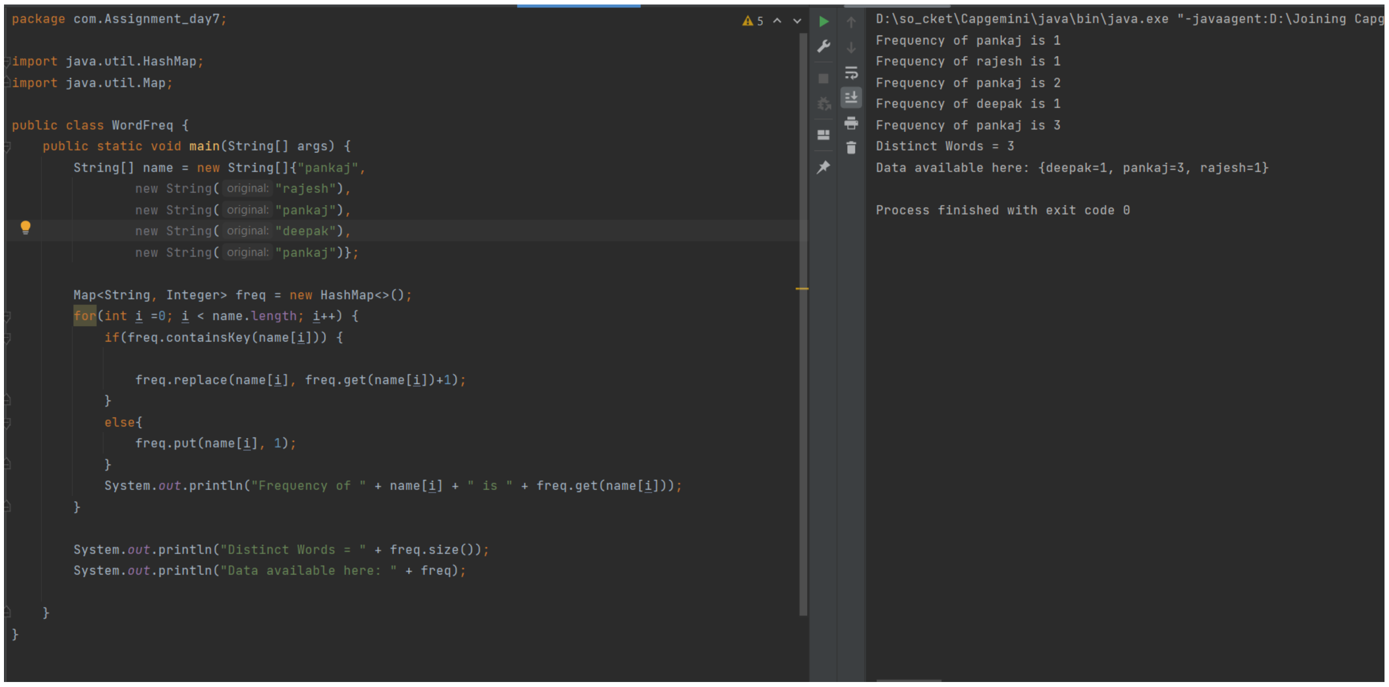
frequency of :deepak is : 1

frequency of :pankaj is : 3

3 distinct word detected

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

Solution –



```
package com.Assignment_day7;

import java.util.HashMap;
import java.util.Map;

public class WordFreq {
    public static void main(String[] args) {
        String[] name = new String[]{"pankaj",
                new String( original: "rajesh"),
                new String( original: "pankaj"),
                new String( original: "deepak"),
                new String( original: "pankaj")};

        Map<String, Integer> freq = new HashMap<>();
        for(int i =0; i < name.length; i++) {
            if(freq.containsKey(name[i])) {

                freq.replace(name[i], freq.get(name[i])+1);
            }
            else{
                freq.put(name[i], 1);
            }
            System.out.println("Frequency of " + name[i] + " is " + freq.get(name[i]));
        }

        System.out.println("Distinct Words = " + freq.size());
        System.out.println("Data available here: " + freq);

    }
}
```

```
D:\so_cket\Capgemini\java\bin\java.exe "-javaagent:D:\Joining Capg
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
Distinct Words = 3
Data available here: {deepak=1, pankaj=3, rajesh=1}

Process finished with exit code 0
```

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)

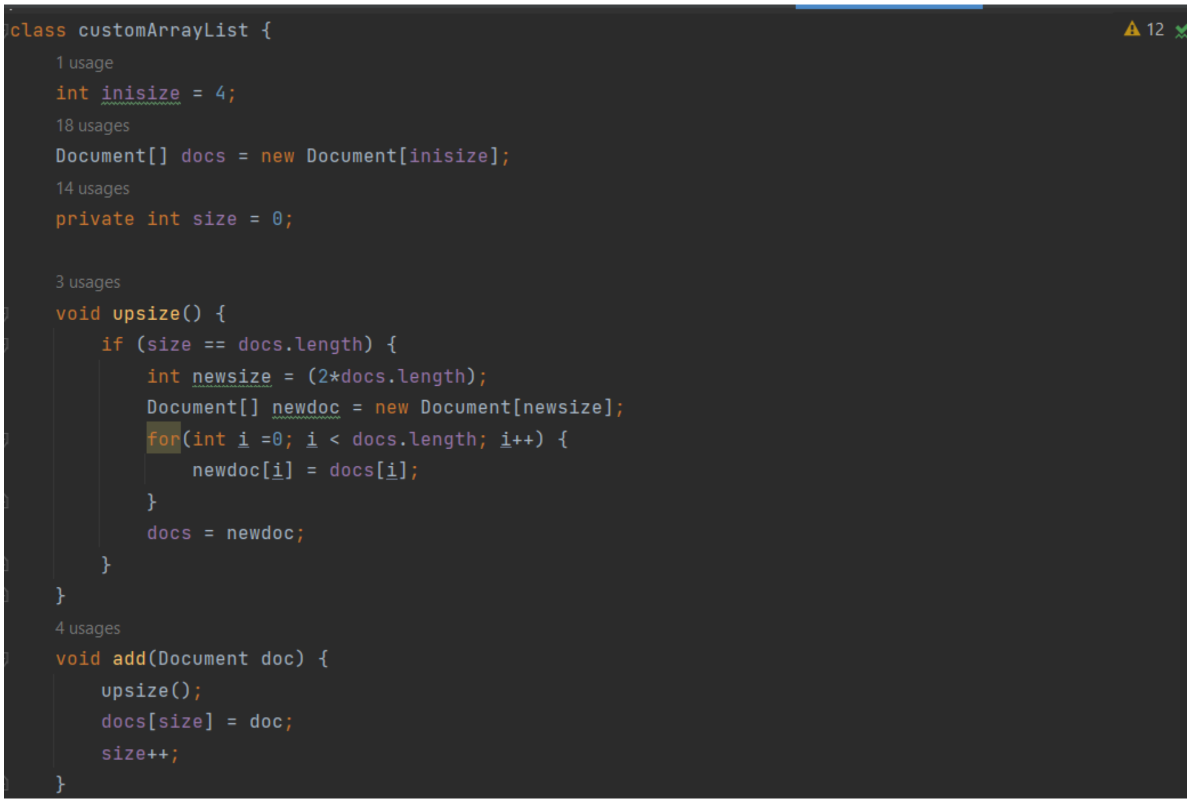
Limitations

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

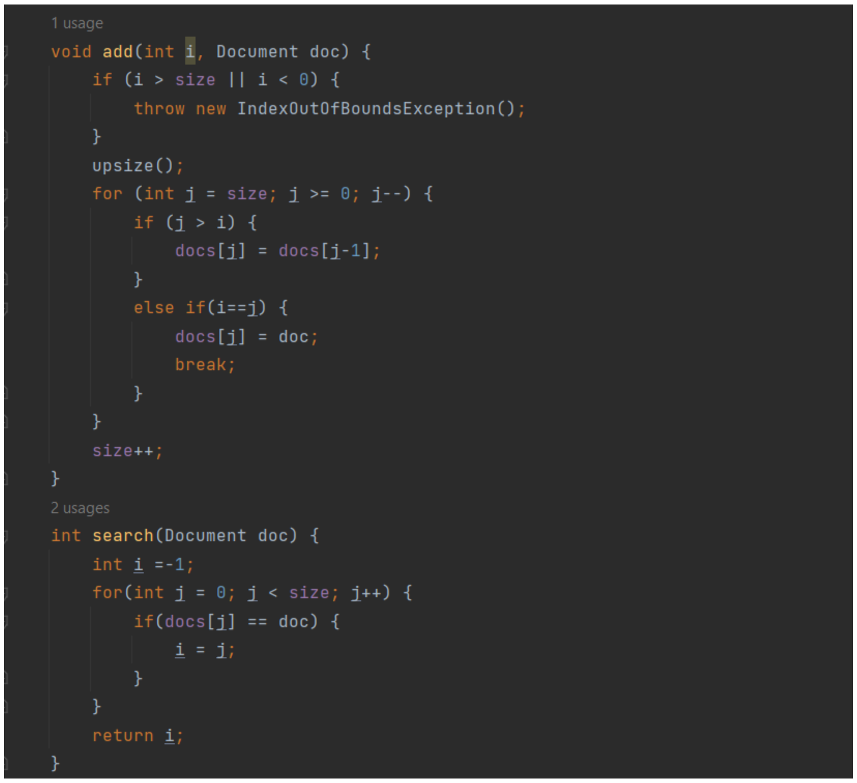
Perform below operations

- Adding
- Searching
- Removing
- Replacing
- Find total documents

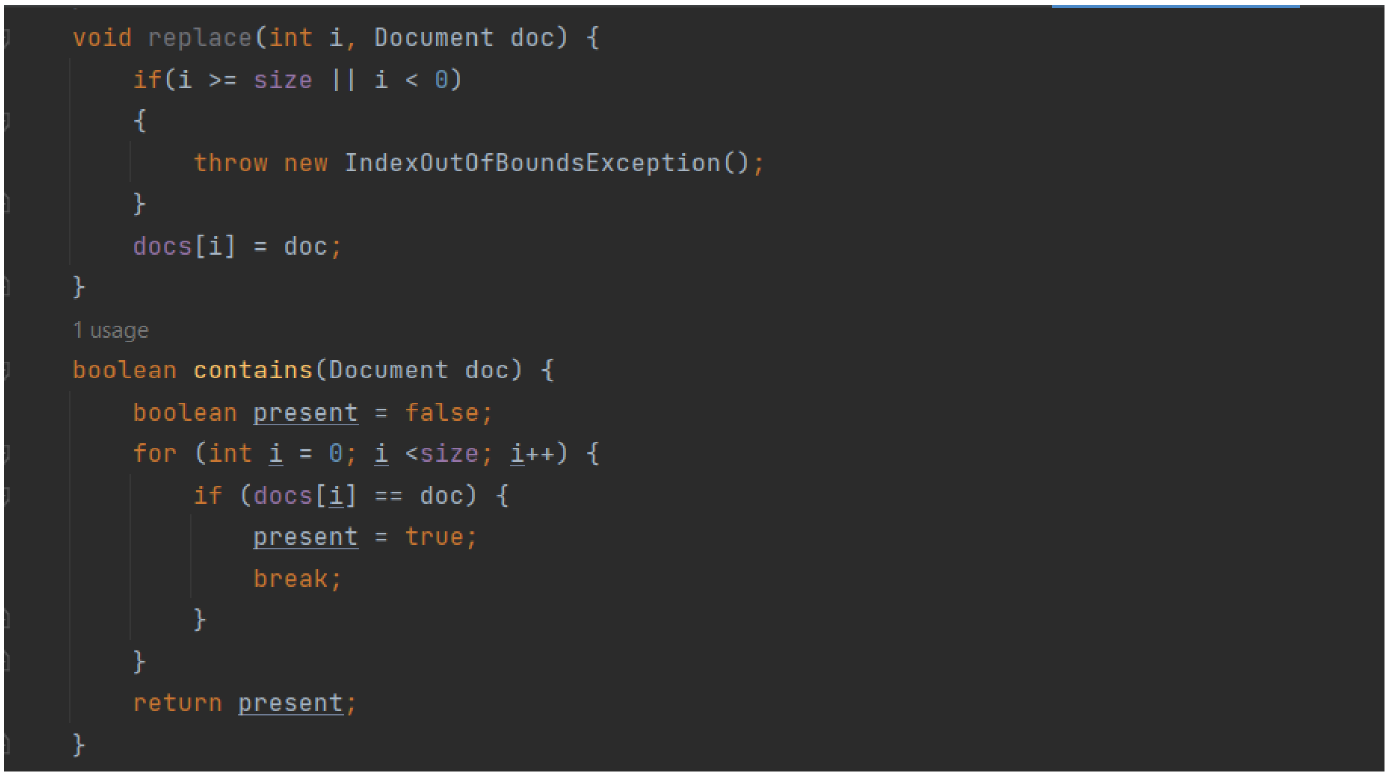
Solution –



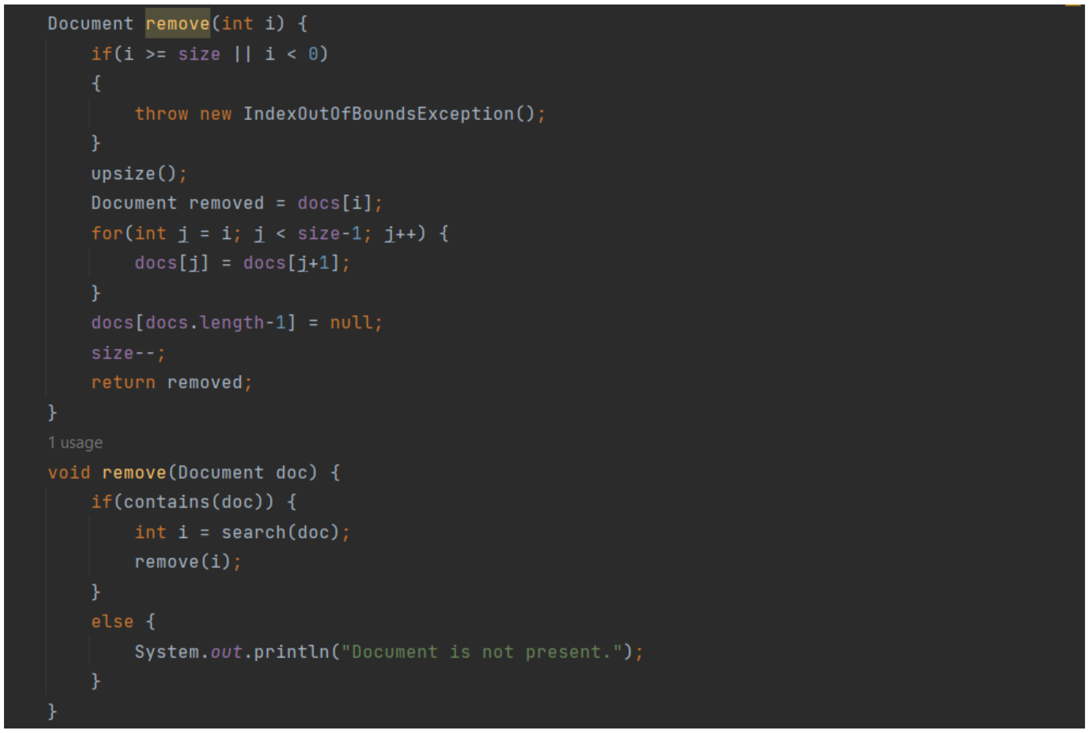
```
class customArrayList {
    1 usage
    int inisize = 4;
    18 usages
    Document[] docs = new Document[inisize];
    14 usages
    private int size = 0;

    3 usages
    void upsize() {
        if (size == docs.length) {
            int newsize = (2*docs.length);
            Document[] newdoc = new Document[newsize];
            for(int i =0; i < docs.length; i++) {
                newdoc[i] = docs[i];
            }
            docs = newdoc;
        }
    }
    4 usages
    void add(Document doc) {
        upsize();
        docs[size] = doc;
        size++;
    }
```



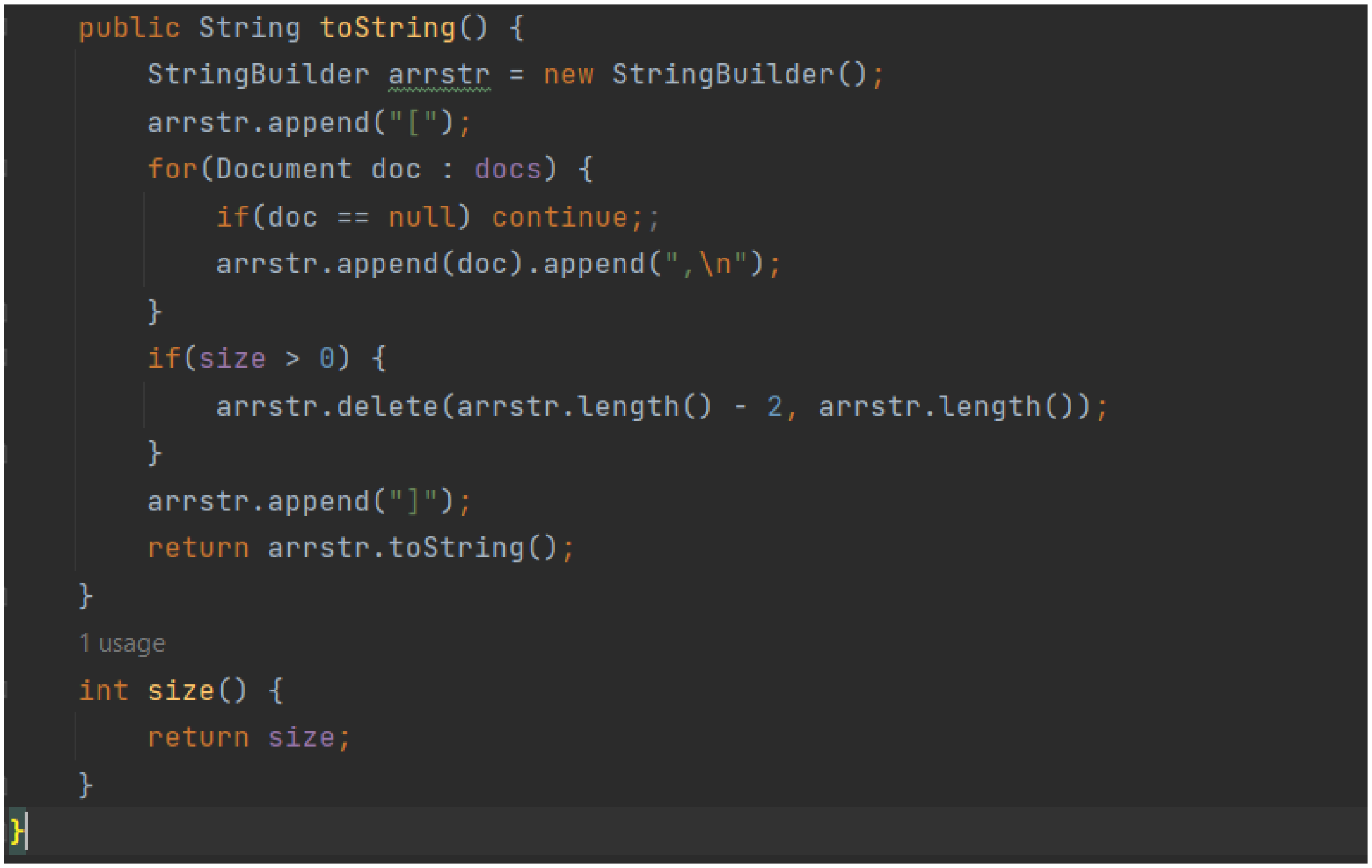
```
1 usage
void add(int i, Document doc) {
    if (i > size || i < 0) {
        throw new IndexOutOfBoundsException();
    }
    upsize();
    for (int j = size; j >= 0; j--) {
        if (j > i) {
            docs[j] = docs[j-1];
        }
        else if(i==j) {
            docs[j] = doc;
            break;
        }
    }
    size++;
}
2 usages
int search(Document doc) {
    int i =-1;
    for(int j = 0; j < size; j++) {
        if(docs[j] == doc) {
            i = j;
        }
    }
    return i;
}
```



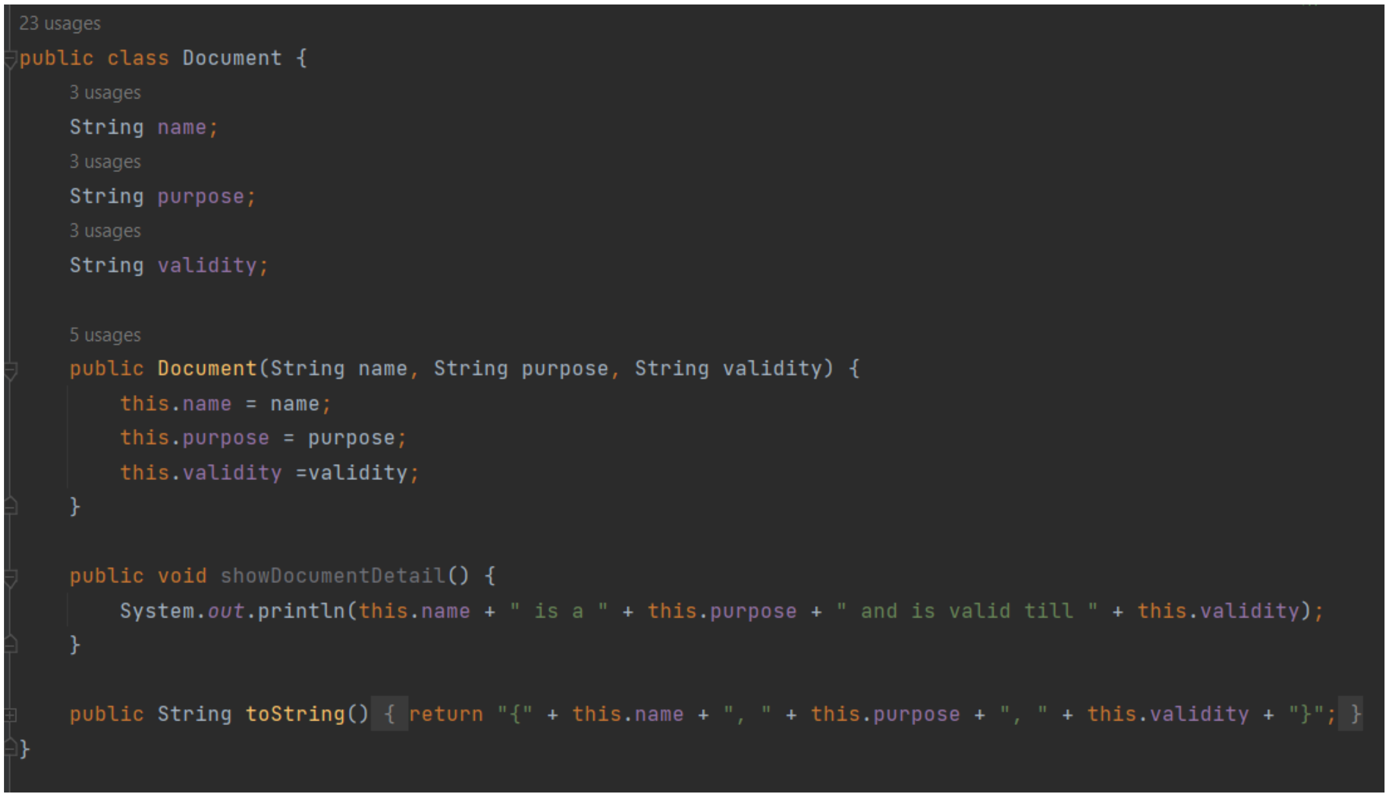
```
void replace(int i, Document doc) {
    if(i >= size || i < 0)
    {
        throw new IndexOutOfBoundsException();
    }
    docs[i] = doc;
}
1 usage
boolean contains(Document doc) {
    boolean present = false;
    for (int i = 0; i <size; i++) {
        if (docs[i] == doc) {
            present = true;
            break;
        }
    }
    return present;
}
```



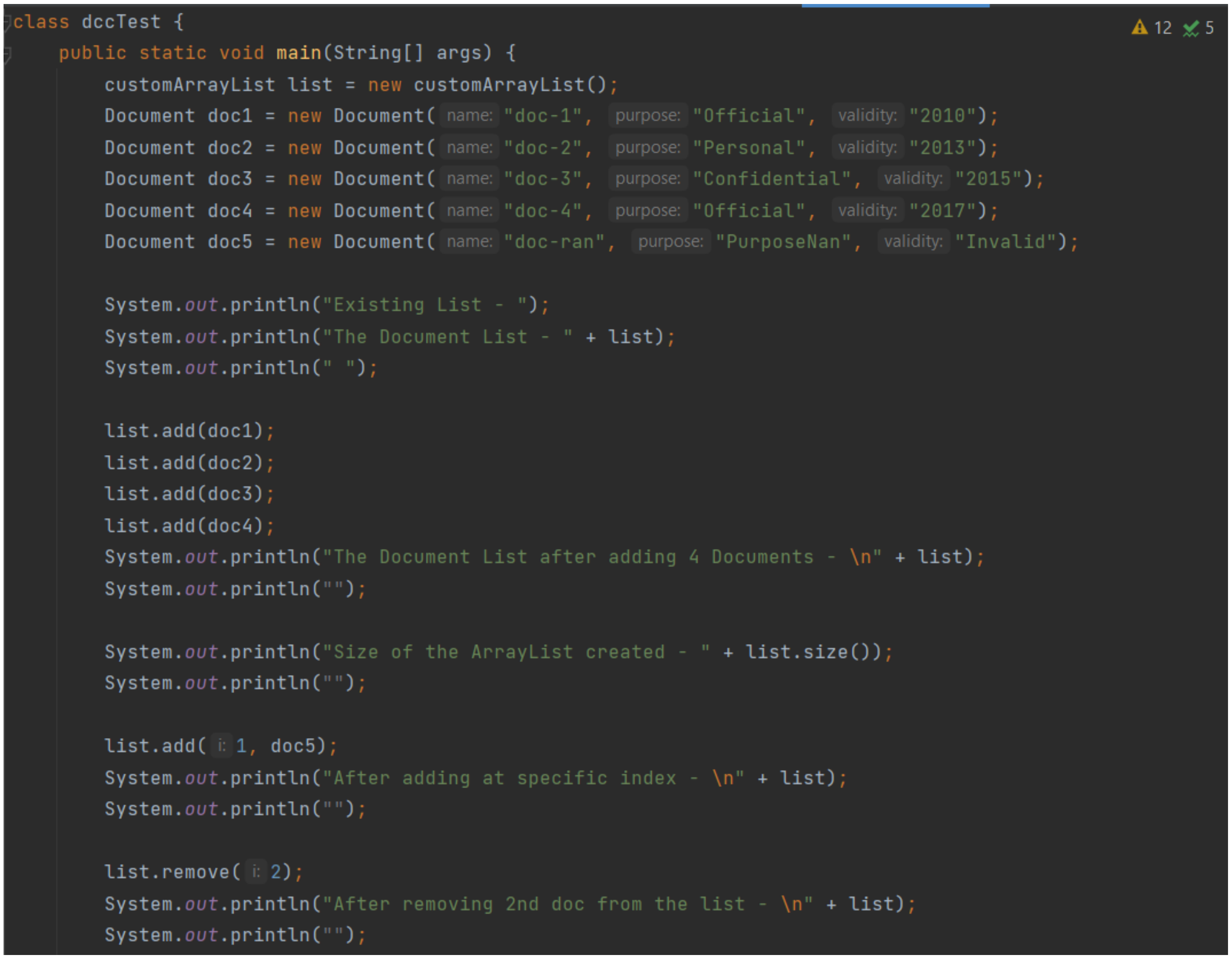
```
Document remove(int i) {
    if(i >= size || i < 0)
    {
        throw new IndexOutOfBoundsException();
    }
    upsize();
    Document removed = docs[i];
    for(int j = i; j < size-1; j++) {
        docs[j] = docs[j+1];
    }
    docs[docs.length-1] = null;
    size--;
    return removed;
}
1 usage
void remove(Document doc) {
    if(contains(doc)) {
        int i = search(doc);
        remove(i);
    }
    else {
        System.out.println("Document is not present.");
    }
}
```



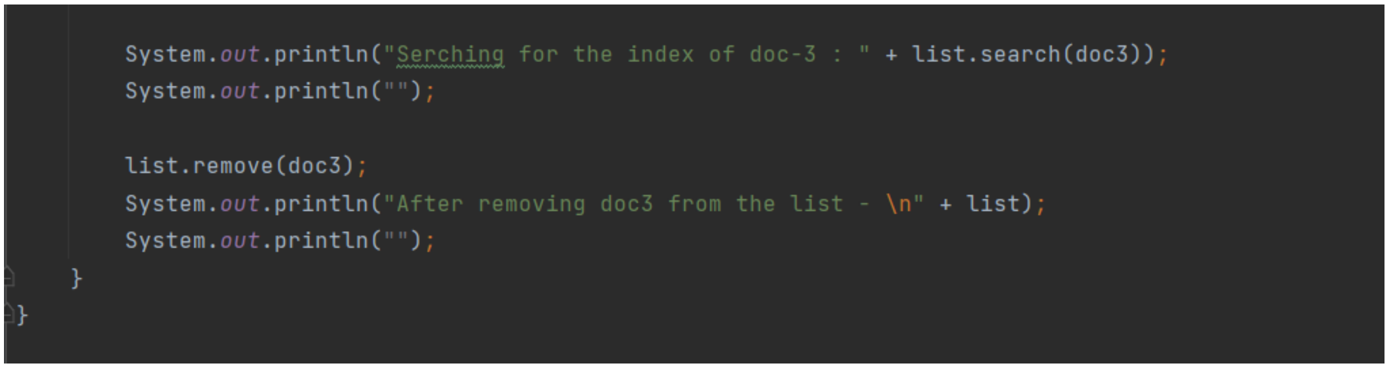
```
    public String toString() {
        StringBuilder arrstr = new StringBuilder();
        arrstr.append("[");
        for(Document doc : docs) {
            if(doc == null) continue;;
            arrstr.append(doc).append(",\n");
        }
        if(size > 0) {
            arrstr.delete(arrstr.length() - 2, arrstr.length());
        }
        arrstr.append("]");
        return arrstr.toString();
    }
    1 usage
    int size() {
        return size;
    }
}
```



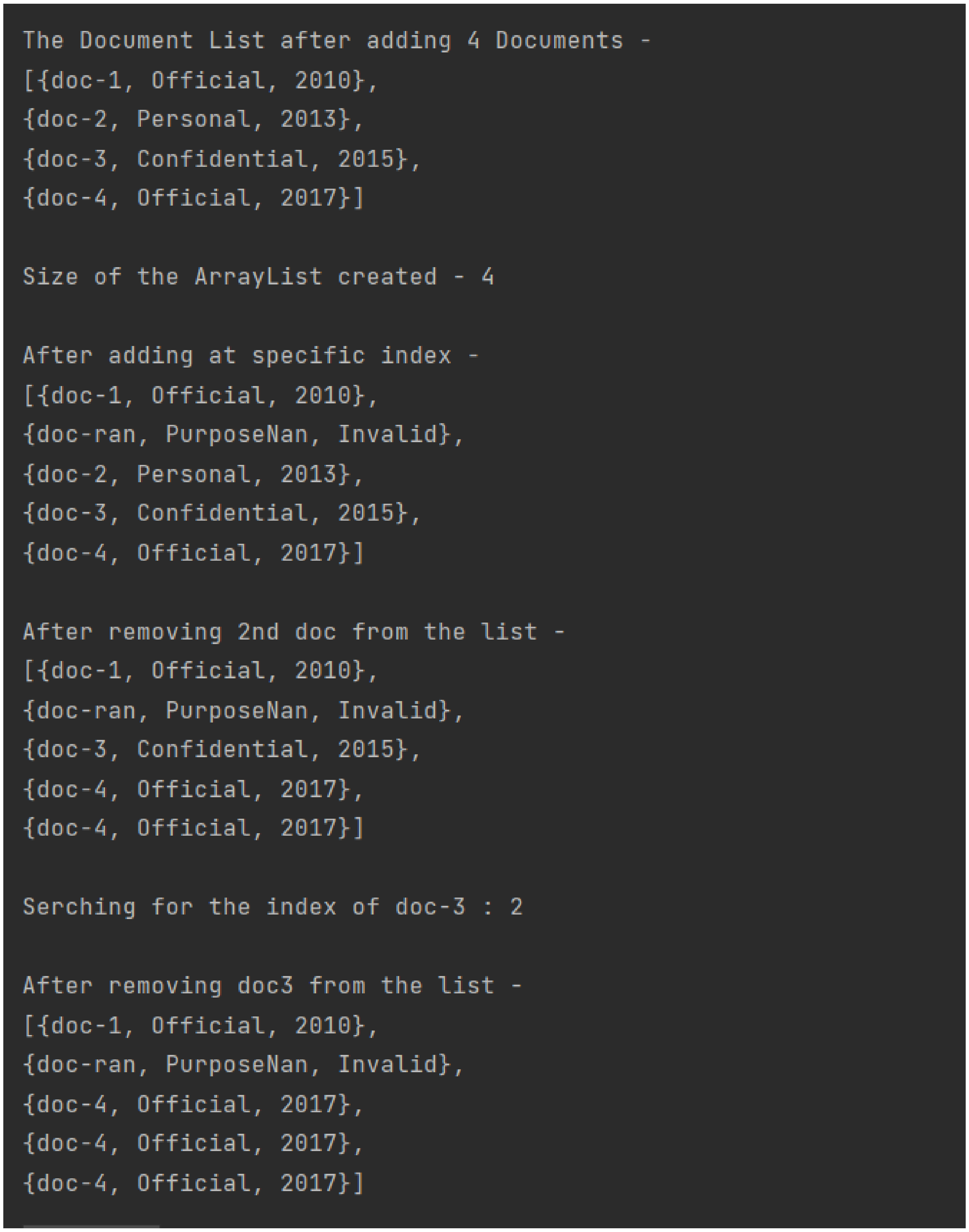
```
23 usages
public class Document {
    3 usages
    String name;
    3 usages
    String purpose;
    3 usages
    String validity;

    5 usages
    public Document(String name, String purpose, String validity) {
        this.name = name;
        this.purpose = purpose;
        this.validity =validity;
    }

    public void showDocumentDetail() {
        System.out.println(this.name + " is a " + this.purpose + " and is valid till " + this.validity);
    }

    public String toString() { return "{" + this.name + ", " + this.purpose + ", " + this.validity + "}"; }
}
```



```
class dccTest {                                                                              ⚠ 12 ✓ 5
    public static void main(String[] args) {
        customArrayList list = new customArrayList();
        Document doc1 = new Document( name: "doc-1",  purpose: "Official",  validity: "2010");
        Document doc2 = new Document( name: "doc-2",  purpose: "Personal",  validity: "2013");
        Document doc3 = new Document( name: "doc-3",  purpose: "Confidential",  validity: "2015");
        Document doc4 = new Document( name: "doc-4",  purpose: "Official",  validity: "2017");
        Document doc5 = new Document( name: "doc-ran",  purpose: "PurposeNan",  validity: "Invalid");

        System.out.println("Existing List - ");
        System.out.println("The Document List - " + list);
        System.out.println(" ");

        list.add(doc1);
        list.add(doc2);
        list.add(doc3);
        list.add(doc4);
        System.out.println("The Document List after adding 4 Documents - \n" + list);
        System.out.println("");

        System.out.println("Size of the ArrayList created - " + list.size());
        System.out.println("");

        list.add( i: 1, doc5);
        System.out.println("After adding at specific index - \n" + list);
        System.out.println("");

        list.remove( i: 2);
        System.out.println("After removing 2nd doc from the list - \n" + list);
        System.out.println("");
```



```
        System.out.println("Serching for the index of doc-3 : " + list.search(doc3));
        System.out.println("");

        list.remove(doc3);
        System.out.println("After removing doc3 from the list - \n" + list);
        System.out.println("");
    }
}
```



```
The Document List after adding 4 Documents -
[{doc-1, Official, 2010},
{doc-2, Personal, 2013},
{doc-3, Confidential, 2015},
{doc-4, Official, 2017}]

Size of the ArrayList created - 4

After adding at specific index -
[{doc-1, Official, 2010},
{doc-ran, PurposeNan, Invalid},
{doc-2, Personal, 2013},
{doc-3, Confidential, 2015},
{doc-4, Official, 2017}]

After removing 2nd doc from the list -
[{doc-1, Official, 2010},
{doc-ran, PurposeNan, Invalid},
{doc-3, Confidential, 2015},
{doc-4, Official, 2017},
{doc-4, Official, 2017}]

Serching for the index of doc-3 : 2

After removing doc3 from the list -
[{doc-1, Official, 2010},
{doc-ran, PurposeNan, Invalid},
{doc-4, Official, 2017},
{doc-4, Official, 2017},
{doc-4, Official, 2017}]
```

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

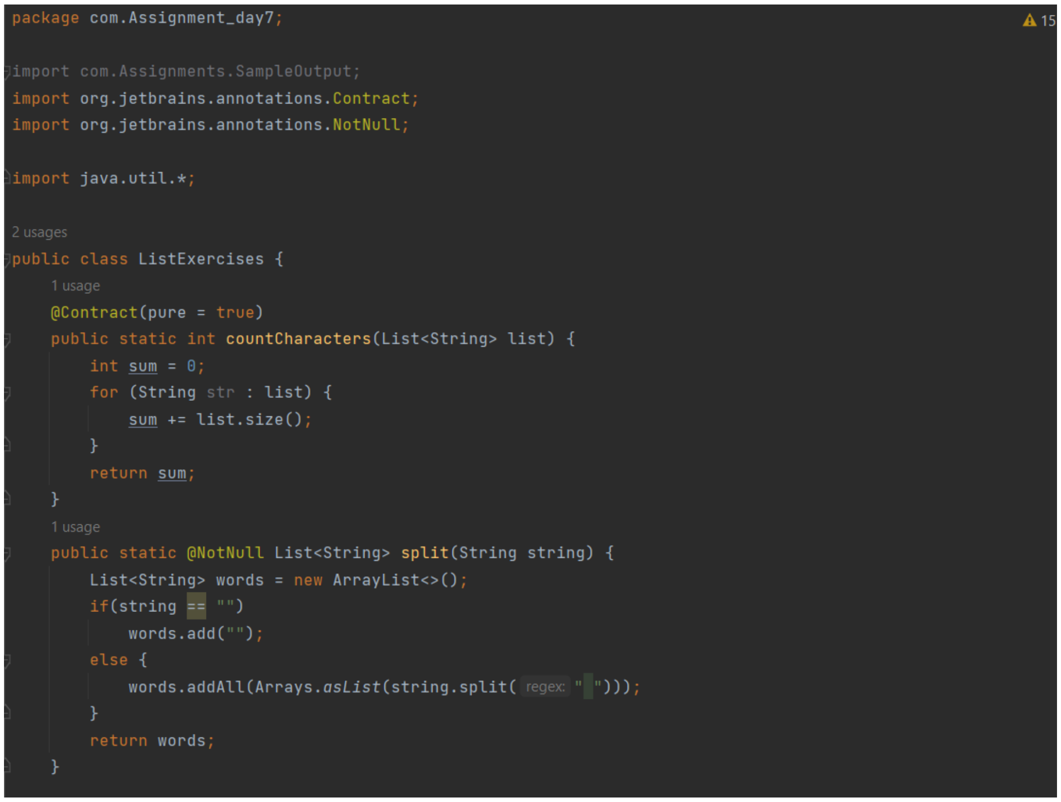
public static List<String> filterContaining(List<String> list, char c) {
return null;
}
	Returns a list of strings selected from a supplied list, which contain the character c. The returned list is in the same order as the original list, but it omits all strings that do not contain c. The original list 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

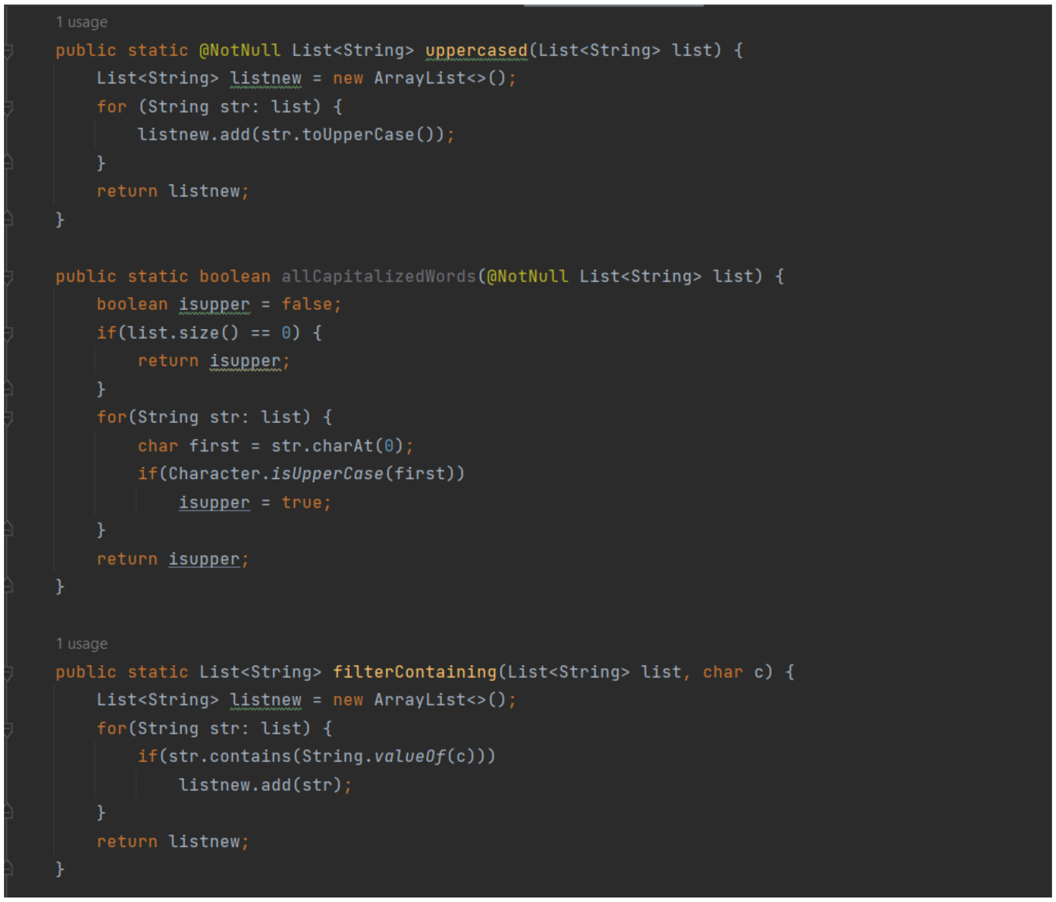
Class Name	ListExercisesTest

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

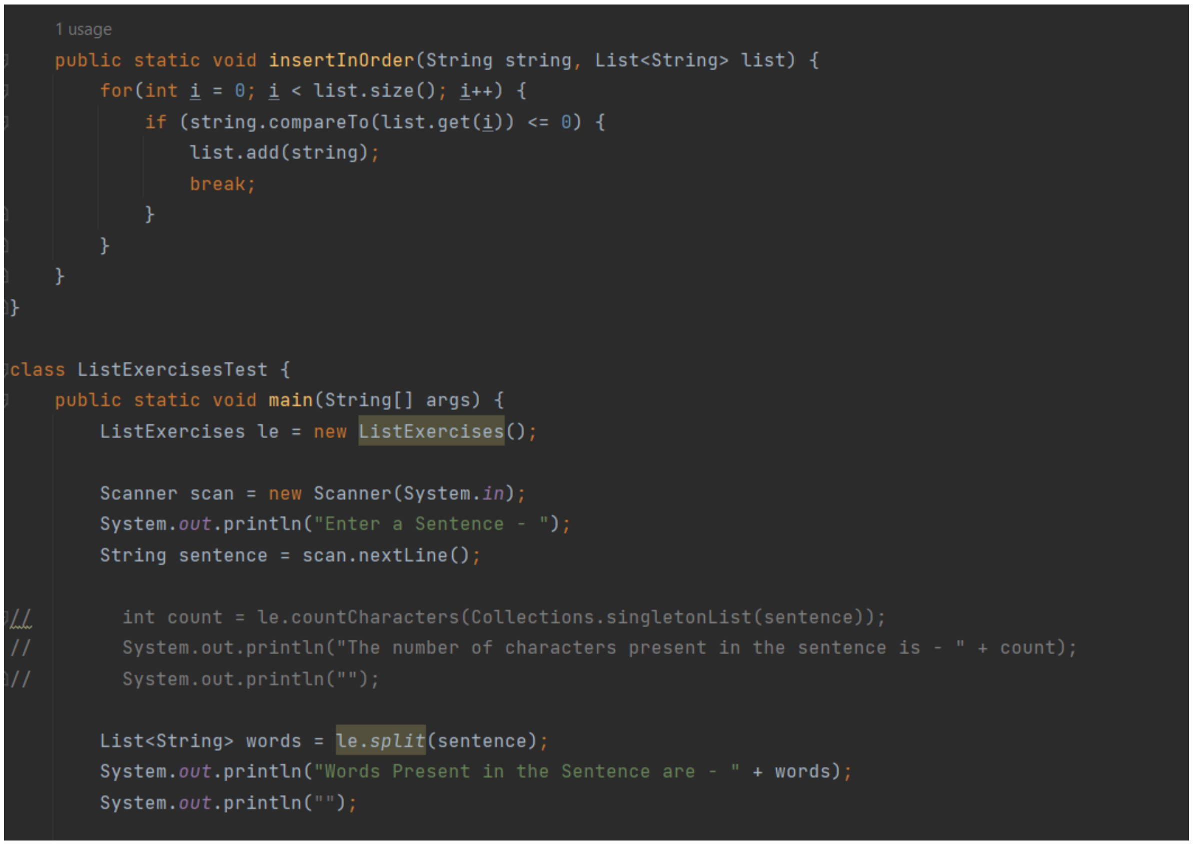
Solution –



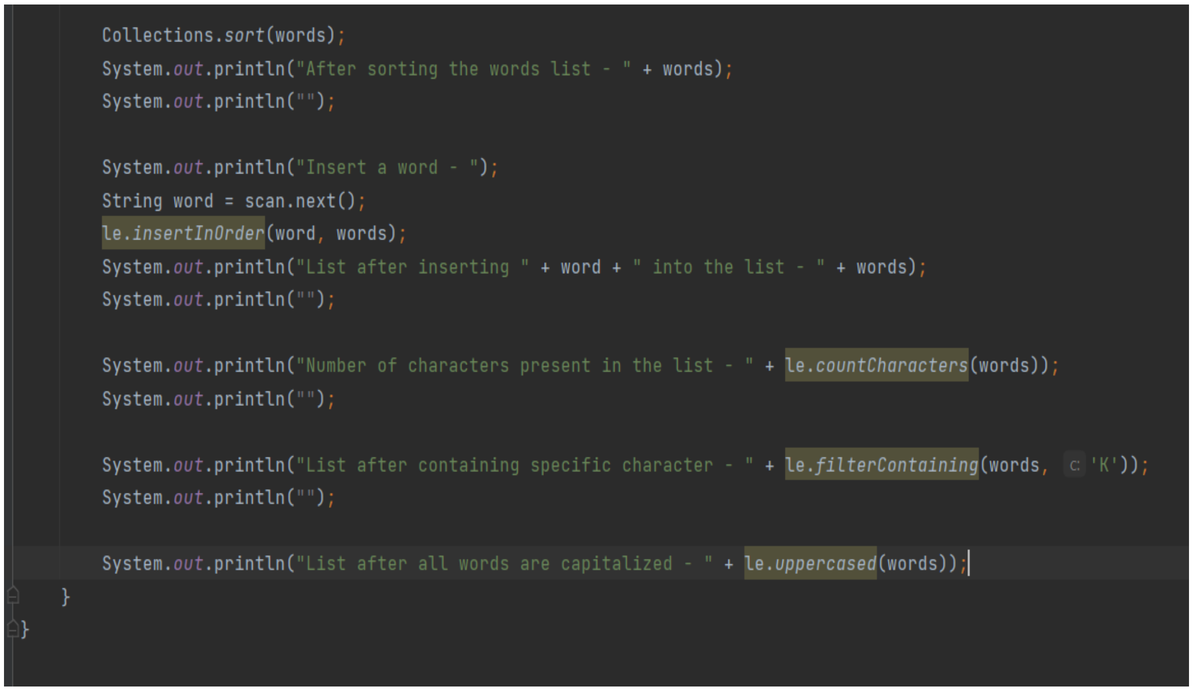
```
package com.Assignment_day7;

import com.Assignments.SampleOutput;
import org.jetbrains.annotations.Contract;
import org.jetbrains.annotations.NotNull;

import java.util.*;

2 usages
public class ListExercises {
    1 usage
    @Contract(pure = true)
    public static int countCharacters(List<String> list) {
        int sum = 0;
        for (String str : list) {
            sum += list.size();
        }
        return sum;
    }
    1 usage
    public static @NotNull List<String> split(String string) {
        List<String> words = new ArrayList<>();
        if(string == "")
            words.add("");
        else {
            words.addAll(Arrays.asList(string.split( regex: " ")));
        }
        return words;
    }
```



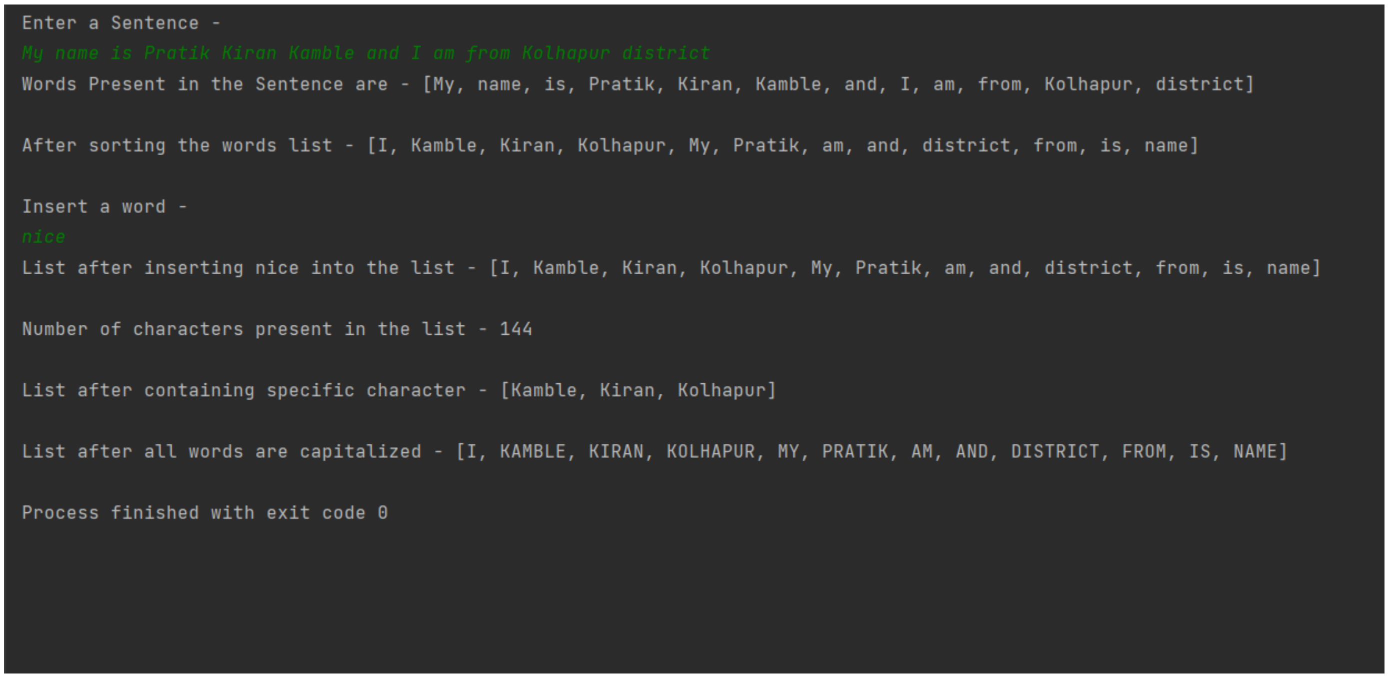
```
    1 usage
    public static @NotNull List<String> uppercased(List<String> list) {
        List<String> listnew = new ArrayList<>();
        for (String str: list) {
            listnew.add(str.toUpperCase());
        }
        return listnew;
    }

    public static boolean allCapitalizedWords(@NotNull List<String> list) {
        boolean isupper = false;
        if(list.size() == 0) {
            return isupper;
        }
        for(String str: list) {
            char first = str.charAt(0);
            if(Character.isUpperCase(first))
                isupper = true;
        }
        return isupper;
    }

    1 usage
    public static List<String> filterContaining(List<String> list, char c) {
        List<String> listnew = new ArrayList<>();
        for(String str: list) {
            if(str.contains(String.valueOf(c)))
                listnew.add(str);
        }
        return listnew;
    }
```



```
    1 usage
    public static void insertInOrder(String string, List<String> list) {
        for(int i = 0; i < list.size(); i++) {
            if (string.compareTo(list.get(i)) <= 0) {
                list.add(string);
                break;
            }
        }
    }
}

class ListExercisesTest {
    public static void main(String[] args) {
        ListExercises le = new ListExercises();

        Scanner scan = new Scanner(System.in);
        System.out.println("Enter a Sentence - ");
        String sentence = scan.nextLine();

//        int count = le.countCharacters(Collections.singletonList(sentence));
//        System.out.println("The number of characters present in the sentence is - " + count);
//        System.out.println("");

        List<String> words = le.split(sentence);
        System.out.println("Words Present in the Sentence are - " + words);
        System.out.println("");
```



```
        Collections.sort(words);
        System.out.println("After sorting the words list - " + words);
        System.out.println("");

        System.out.println("Insert a word - ");
        String word = scan.next();
        le.insertInOrder(word, words);
        System.out.println("List after inserting " + word + " into the list - " + words);
        System.out.println("");

        System.out.println("Number of characters present in the list - " + le.countCharacters(words));
        System.out.println("");

        System.out.println("List after containing specific character - " + le.filterContaining(words, c: 'K'));
        System.out.println("");

        System.out.println("List after all words are capitalized - " + le.uppercased(words));
    }
}
```



```
Enter a Sentence -
My name is Pratik Kiran Kamble and I am from Kolhapur district
Words Present in the Sentence are - [My, name, is, Pratik, Kiran, Kamble, and, I, am, from, Kolhapur, district]

After sorting the words list - [I, Kamble, Kiran, Kolhapur, My, Pratik, am, and, district, from, is, name]

Insert a word -
nice
List after inserting nice into the list - [I, Kamble, Kiran, Kolhapur, My, Pratik, am, and, district, from, is, name]

Number of characters present in the list - 144

List after containing specific character - [Kamble, Kiran, Kolhapur]

List after all words are capitalized - [I, KAMBLE, KIRAN, KOLHAPUR, MY, PRATIK, AM, AND, DISTRICT, FROM, IS, NAME]

Process finished with exit code 0
```