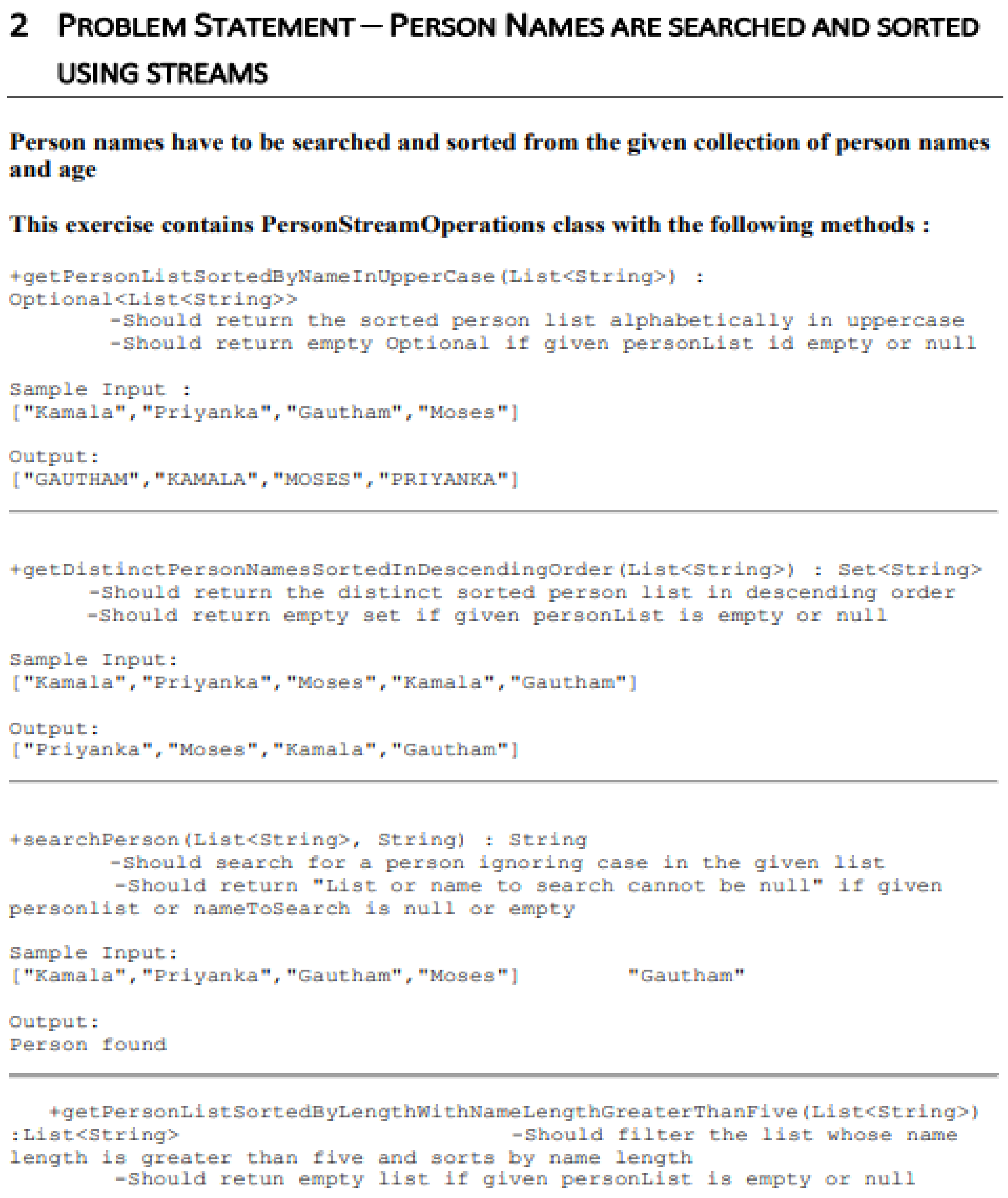
JAVA - J2EE Batch 2

Name – Aman Yadav

E-mail : prakashaman5@gmail.com

Phone: +919519131321

Assignment-10



2 Problem Statement – Person Names are searched and sorted using streams

Person names have to be searched and sorted from the given collection of person names and age

This exercise contains PersonStreamOperations class with the following methods :

```
+getPersonListSortedByNameInUpperCase(List<String>) :
Optional<List<String>>
        -Should return the sorted person list alphabetically in uppercase
        -Should return empty Optional if given personList id empty or null

Sample Input :
["Kamala","Priyanka","Gautham","Moses"]

Output:
["GAUTHAM","KAMALA","MOSES","PRIYANKA"]
```

```
+getDistinctPersonNamesSortedInDescendingOrder(List<String>) : Set<String>
      -Should return the distinct sorted person list in descending order
      -Should return empty set if given personList is empty or null

Sample Input:
["Kamala","Priyanka","Moses","Kamala","Gautham"]

Output:
["Priyanka","Moses","Kamala","Gautham"]
```

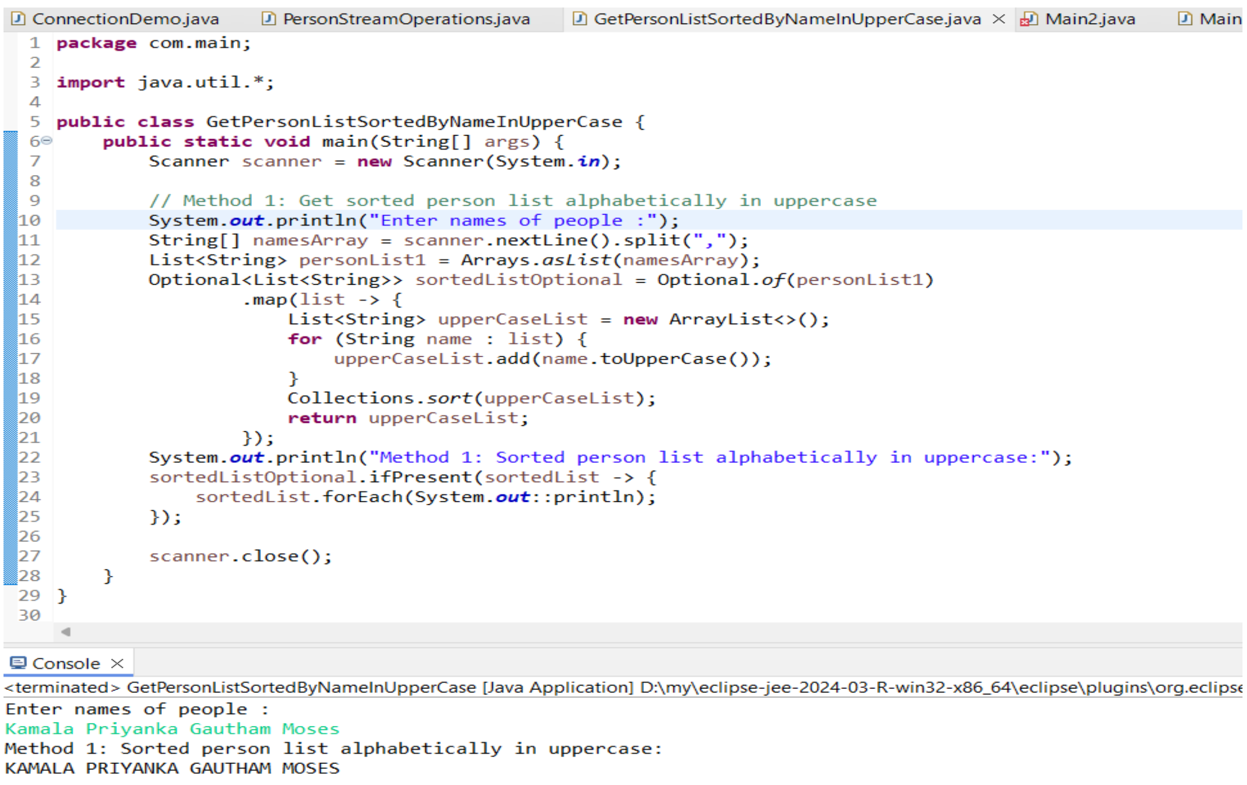
```
+searchPerson(List<String>, String) : String
        -Should search for a person ignoring case in the given list
        -Should return "List or name to search cannot be null" if given
personlist or nameToSearch is null or empty

Sample Input:
["Kamala","Priyanka","Gautham","Moses"]        "Gautham"

Output:
Person found
```

```
   +getPersonListSortedByLengthWithNameLengthGreaterThanFive(List<String>)
:List<String>                         -Should filter the list whose name
length is greater than five and sorts by name length
        -Should retun empty list if given personList is empty or null
```

```
Sample Input:
["Kamala","Priyanka","Gautham","Moses"]

Output:
["Kamala","Gautham","Priyanka"]
```

```
+getPersonByMaxAge(Map<String, Integer>) : String
       -Should return the person name having maximum age from the given map
      -Should return "Give proper input not null" if given map is null or
empty

Sample Input:
{"Gautham"=30,"Latha"=56,""Punith"=45}

Output:
Latha
```

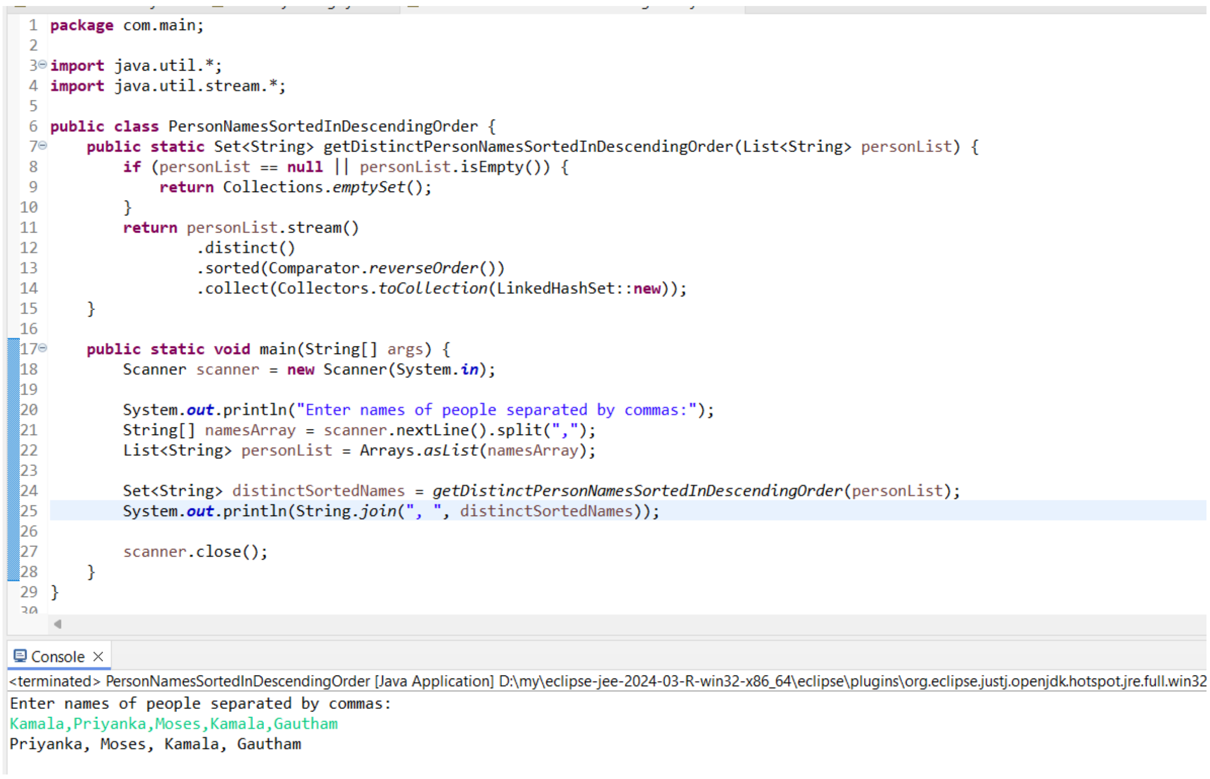
→



ConnectionDemo.java
PersonStreamOperations.java
GetPersonListSortedByNameInUpperCase.java
Main2.java
Main
Console

```java
package com.main;

import java.util.*;

public class GetPersonListSortedByNameInUpperCase {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Method 1: Get sorted person list alphabetically in uppercase
        System.out.println("Enter names of people :");
        String[] namesArray = scanner.nextLine().split(",");
        List<String> personList1 = Arrays.asList(namesArray);
        Optional<List<String>> sortedListOptional = Optional.of(personList1)
                .map(list -> {
                    List<String> upperCaseList = new ArrayList<>();
                    for (String name : list) {
                        upperCaseList.add(name.toUpperCase());
                    }
                    Collections.sort(upperCaseList);
                    return upperCaseList;
                });
        System.out.println("Method 1: Sorted person list alphabetically in uppercase:");
        sortedListOptional.ifPresent(sortedList -> {
            sortedList.forEach(System.out::println);
        });

        scanner.close();
    }
}
```

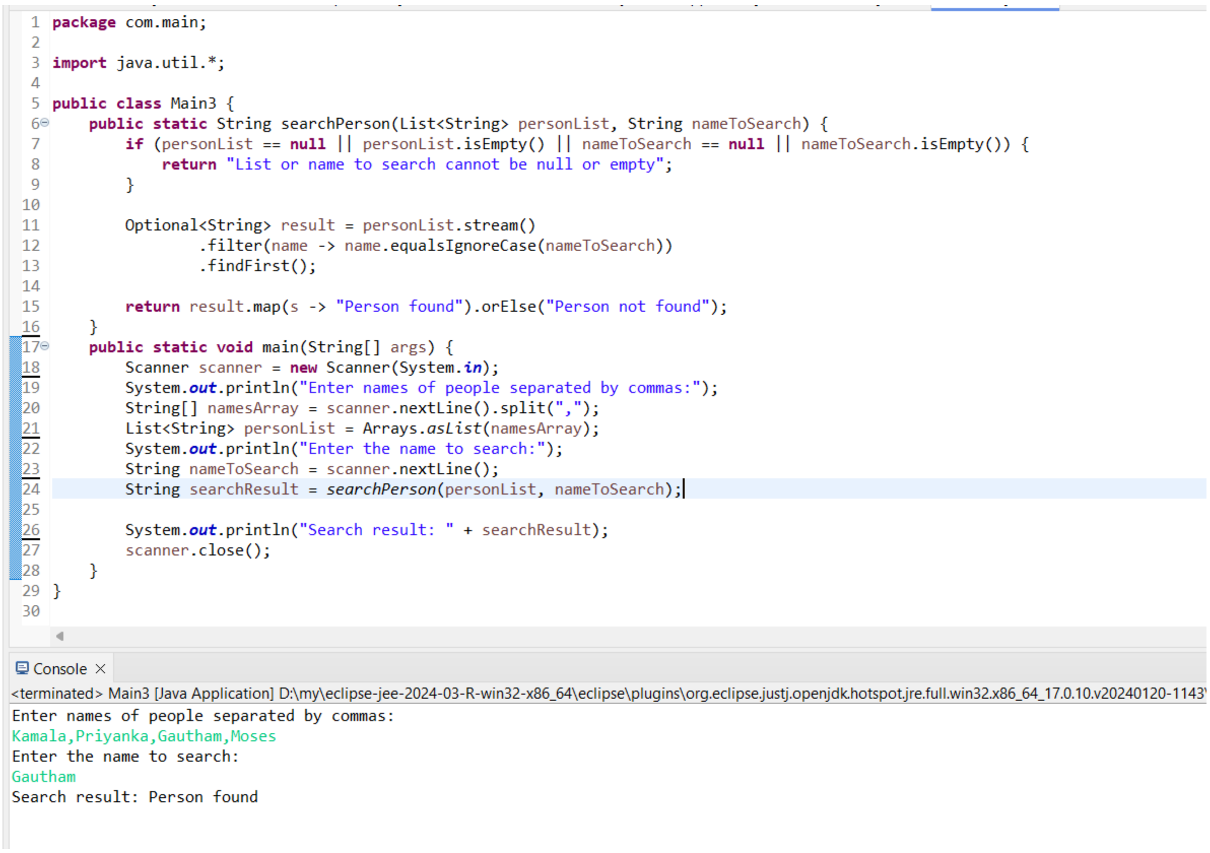
```
<terminated> GetPersonListSortedByNameInUpperCase [Java Application] D:\my\eclipse-jee-2024-03-R-win32-x86_64\eclipse\plugins\org.eclipse
Enter names of people :
Kamala Priyanka Gautham Moses
Method 1: Sorted person list alphabetically in uppercase:
KAMALA PRIYANKA GAUTHAM MOSES
```



```java
package com.main;

import java.util.*;
import java.util.stream.*;

public class PersonNamesSortedInDescendingOrder {
    public static Set<String> getDistinctPersonNamesSortedInDescendingOrder(List<String> personList) {
        if (personList == null || personList.isEmpty()) {
            return Collections.emptySet();
        }
        return personList.stream()
                .distinct()
                .sorted(Comparator.reverseOrder())
                .collect(Collectors.toCollection(LinkedHashSet::new));
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter names of people separated by commas:");
        String[] namesArray = scanner.nextLine().split(",");
        List<String> personList = Arrays.asList(namesArray);

        Set<String> distinctSortedNames = getDistinctPersonNamesSortedInDescendingOrder(personList);
        System.out.println(String.join(", ", distinctSortedNames));

        scanner.close();
    }
}
```

Console

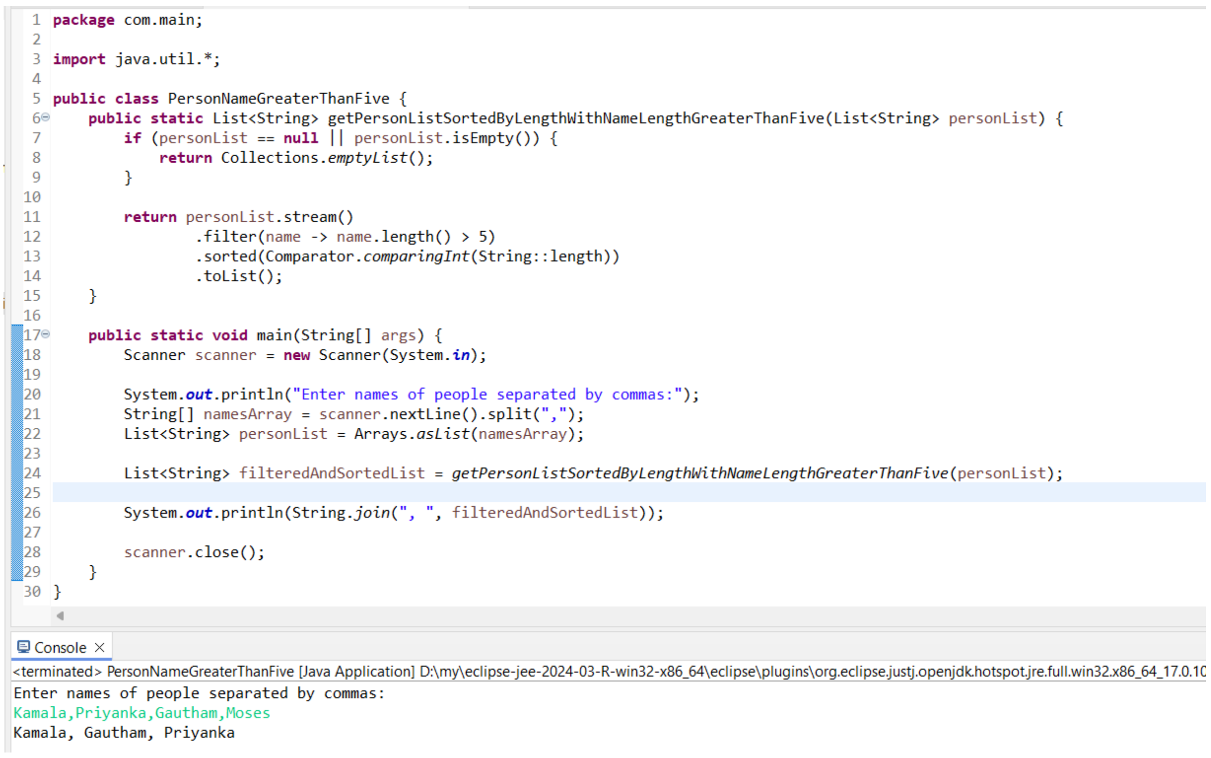
```
<terminated> PersonNamesSortedInDescendingOrder [Java Application] D:\my\eclipse-jee-2024-03-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32
Enter names of people separated by commas:
Kamala,Priyanka,Moses,Kamala,Gautham
Priyanka, Moses, Kamala, Gautham
```



```java
package com.main;

import java.util.*;

public class Main3 {
    public static String searchPerson(List<String> personList, String nameToSearch) {
        if (personList == null || personList.isEmpty() || nameToSearch == null || nameToSearch.isEmpty()) {
            return "List or name to search cannot be null or empty";
        }

        Optional<String> result = personList.stream()
                .filter(name -> name.equalsIgnoreCase(nameToSearch))
                .findFirst();

        return result.map(s -> "Person found").orElse("Person not found");
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter names of people separated by commas:");
        String[] namesArray = scanner.nextLine().split(",");
        List<String> personList = Arrays.asList(namesArray);
        System.out.println("Enter the name to search:");
        String nameToSearch = scanner.nextLine();
        String searchResult = searchPerson(personList, nameToSearch);

        System.out.println("Search result: " + searchResult);
        scanner.close();
    }
}
```

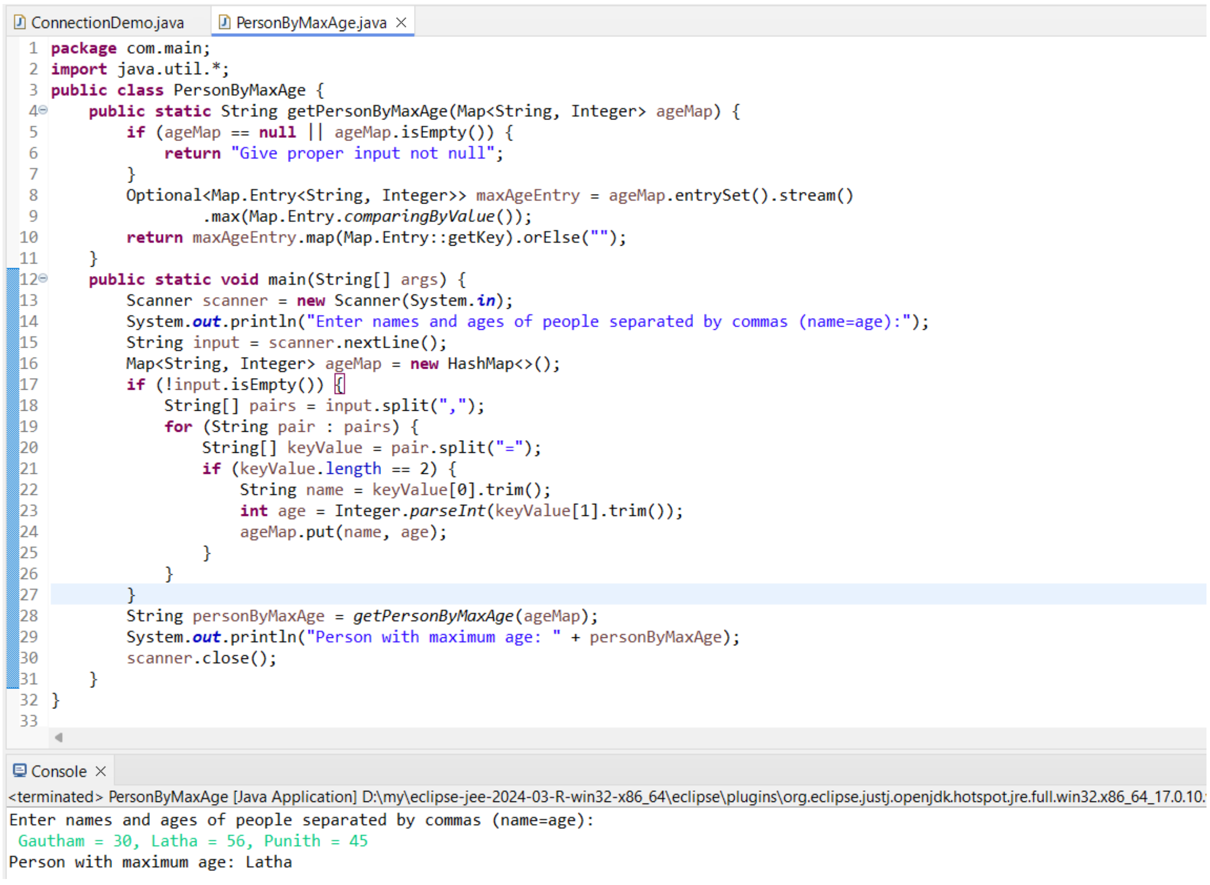
Console

```
<terminated> Main3 [Java Application] D:\my\eclipse-jee-2024-03-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.10.v20240120-1143
Enter names of people separated by commas:
Kamala,Priyanka,Gautham,Moses
Enter the name to search:
Gautham
Search result: Person found
```



```java
package com.main;

import java.util.*;

public class PersonNameGreaterThanFive {
    public static List<String> getPersonListSortedByLengthWithNameLengthGreaterThanFive(List<String> personList) {
        if (personList == null || personList.isEmpty()) {
            return Collections.emptyList();
        }

        return personList.stream()
                .filter(name -> name.length() > 5)
                .sorted(Comparator.comparingInt(String::length))
                .toList();
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter names of people separated by commas:");
        String[] namesArray = scanner.nextLine().split(",");
        List<String> personList = Arrays.asList(namesArray);

        List<String> filteredAndSortedList = getPersonListSortedByLengthWithNameLengthGreaterThanFive(personList);

        System.out.println(String.join(", ", filteredAndSortedList));

        scanner.close();
    }
}
```

```
<terminated> PersonNameGreaterThanFive [Java Application] D:\my\eclipse-jee-2024-03-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.10
Enter names of people separated by commas:
Kamala,Priyanka,Gautham,Moses
Kamala, Gautham, Priyanka
```



```java
package com.main;
import java.util.*;
public class PersonByMaxAge {
    public static String getPersonByMaxAge(Map<String, Integer> ageMap) {
        if (ageMap == null || ageMap.isEmpty()) {
            return "Give proper input not null";
        }
        Optional<Map.Entry<String, Integer>> maxAgeEntry = ageMap.entrySet().stream()
                .max(Map.Entry.comparingByValue());
        return maxAgeEntry.map(Map.Entry::getKey).orElse("");
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter names and ages of people separated by commas (name=age):");
        String input = scanner.nextLine();
        Map<String, Integer> ageMap = new HashMap<>();
        if (!input.isEmpty()) {
            String[] pairs = input.split(",");
            for (String pair : pairs) {
                String[] keyValue = pair.split("=");
                if (keyValue.length == 2) {
                    String name = keyValue[0].trim();
                    int age = Integer.parseInt(keyValue[1].trim());
                    ageMap.put(name, age);
                }
            }
        }
        String personByMaxAge = getPersonByMaxAge(ageMap);
        System.out.println("Person with maximum age: " + personByMaxAge);
        scanner.close();
    }
}

```

```
<terminated> PersonByMaxAge [Java Application] D:\my\eclipse-jee-2024-03-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.10.
Enter names and ages of people separated by commas (name=age):
 Gautham = 30, Latha = 56, Punith = 45
Person with maximum age: Latha
```