Assignment 10

PERSON NAMES ARE SEARCHED AND SORTED USING STREAMS:

CODE:

```java
package Assignment10;
import java.util.*;
import java.util.stream.Collectors;

public class PersonStreamOperations {

    public static Optional<List<String>> getPersonListSortedByNameInUpperCase(List<String> personList) {  1 usage
        if (personList == null || personList.isEmpty()) {
            return Optional.empty();
        }
        List<String> sortedList = personList.stream()
                .map(String::toUpperCase)
                .sorted()
                .collect(Collectors.toList());
        return Optional.of(sortedList);
    }

    public static Set<String> getDistinctPersonNamesSortedInDescendingOrder(List<String> personList) {  1 usage
        if (personList == null || personList.isEmpty()) {
            return Collections.emptySet();
        }
        Set<String> distinctSortedSet = personList.stream()
                .distinct()
                .sorted(Comparator.reverseOrder())
                .collect(Collectors.toCollection(LinkedHashSet::new));
        return distinctSortedSet;
    }
```

```java
    public static String searchPerson(List<String> personList, String nameToSearch) {  1 usage
        if (personList == null || personList.isEmpty() || nameToSearch == null || nameToSearch.isEmpty()) {
            return "List or name to search cannot be null";
        }
        boolean found = personList.stream()
                .anyMatch(name -> name.equalsIgnoreCase(nameToSearch));
        return found ? "Person found" : "Person not found";
    }

    public static List<String> getPersonListSortedByLengthWithNameLengthGreaterThanFive(List<String> personList) {  1 usa
        if (personList == null || personList.isEmpty()) {
            return Collections.emptyList();
        }
        List<String> filteredList = personList.stream()
                .filter(name -> name.length() > 5)
                .sorted(Comparator.comparingInt(String::length))
                .collect(Collectors.toList());
        return filteredList;
    }
    public static String getPersonByMaxAge(Map<String, Integer> ageMap) {  1 usage
        if (ageMap == null || ageMap.isEmpty()) {
            return "Give proper input not null";
        }
        return ageMap.entrySet().stream()  Stream<Entry<...>>
                .max(Map.Entry.comparingByValue())  Optional<Entry<...>>
                .map(Map.Entry::getKey)  Optional<String>
                .orElse( other: null);
    }
```

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

    System.out.print("Enter the number of person names: ");
    int numNames = scanner.nextInt();
    scanner.nextLine();
    List<String> personList = new ArrayList<>();
    for (int i = 0; i < numNames; i++) {
        System.out.print("Enter person name " + (i + 1) + ": ");
        String name = scanner.nextLine();
        personList.add(name);
    }
    System.out.println("Person List: " + personList);
    Optional<List<String>> sortedNames = getPersonListSortedByNameInUpperCase(personList);
    sortedNames.ifPresent(names -> {
        System.out.println(names);
    });

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

    System.out.print("Enter the name to search: ");
    String nameToSearch = scanner.nextLine();
    String searchResult = searchPerson(personList, nameToSearch);
    System.out.println(searchResult);
```

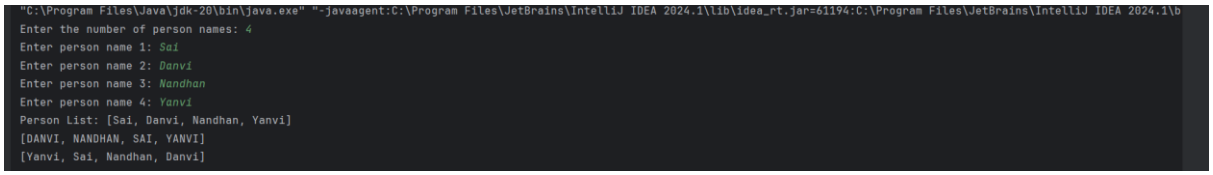
```java

    List<String> filteredNames = getPersonListSortedByLengthWithNameLengthGreaterThanFive(personList);
    System.out.println(filteredNames);

    Map<String, Integer> ageMap = new HashMap<>();
    System.out.println("Enter key-value pairs separated by commas (press Enter to stop):");
    while (true) {
        String input = scanner.nextLine();
        if (input.isEmpty()) {
            break;
        }
        String[] parts = input.split( regex: ",");
        if (parts.length != 2) {
            System.out.println("Invalid input format. Please try again.");
            continue;
        }
        String key = parts[0].trim();
        int value = Integer.parseInt(parts[1].trim());
        ageMap.put(key, value);
    }

    System.out.println("Map: " + ageMap);
    String maxAgePerson = getPersonByMaxAge(ageMap);
    System.out.println(maxAgePerson);
    scanner.close();
}
}
```

Output1:



Output2:

Output3: