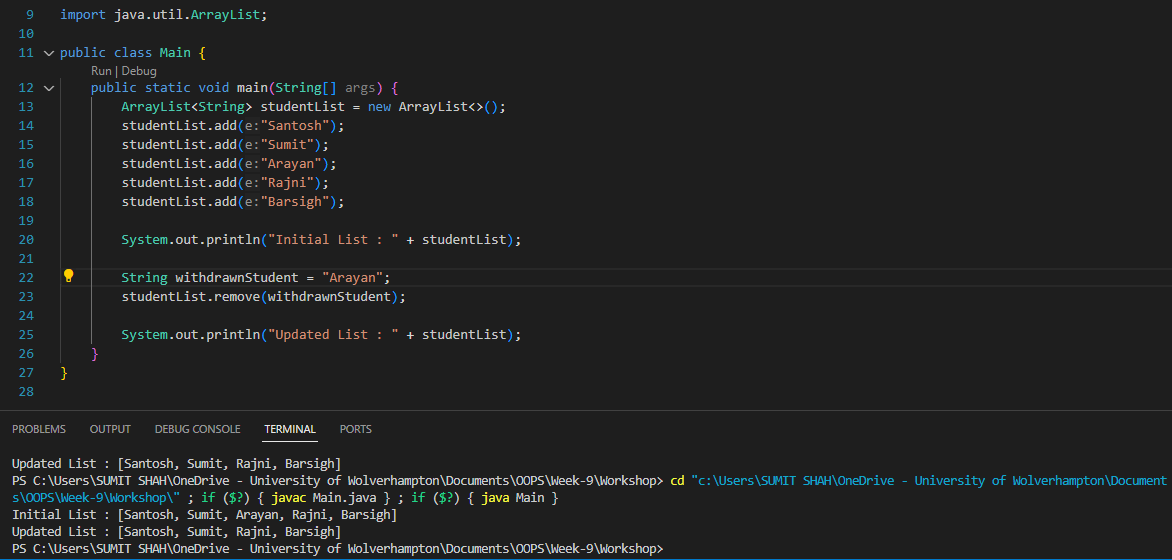
**ArrayList and LinkedList:**

1. Create an ArrayList to store the names of students in a class. Add, remove, and print the list of students.

* Initialize an empty ArrayList to store examinee names.
* Add the names of five examinee participating in the exam to the ArrayList.
* Remove the name of the examinee who withdrew from the exam.
* Print the updated list of participants.



1. Write a program to insert elements into the linked list at the first and last positions. Also check if the linked list is empty or not.

A screen shot of a computer

Description automatically generated

1. Rotate the elements of an ArrayList to the right by a given number of positions. For example, if the ArrayList is [1, 2, 3, 4, 5] and you rotate it by 2 positions, the result should be [4, 5, 1, 2, 3].

A screen shot of a computer

Description automatically generated

1. Write a program to declare a linkedList, colors to store String. Insert five colors into the linked list.
2. Iterate and print all the colors.
3. Check if “Red” exists in the linkedList or not.
4. Shuffle the elements of the list and print them.
5. Print the LinkedList in ascending order

A screen shot of a computer

Description automatically generated

A computer screen shot of text

Description automatically generated

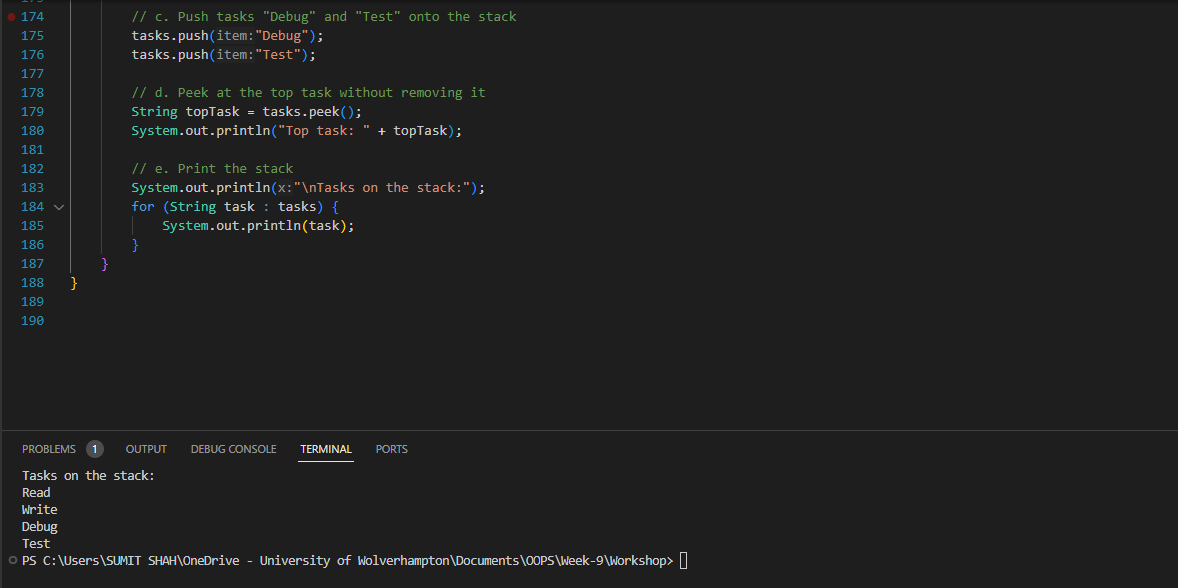
**A black screen with a black background

Description automatically generated**

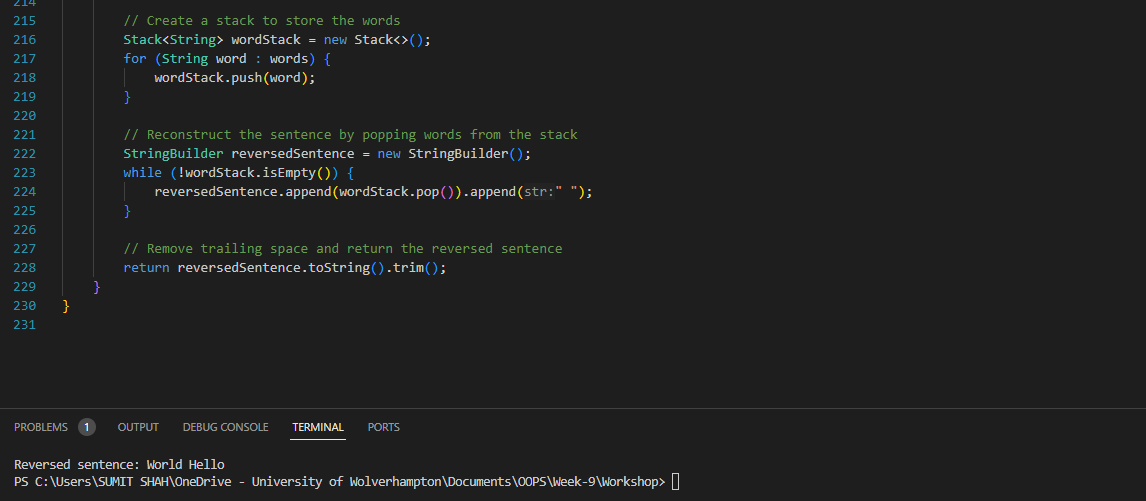
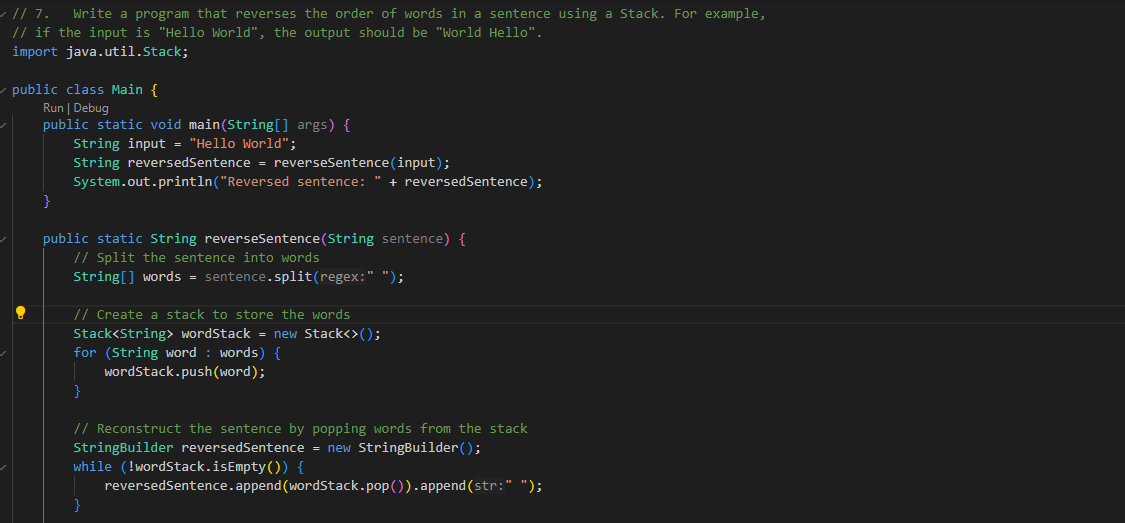
**Stack:**

1. Create a Stack to manage a sequence of tasks. Implement the following operations:
2. Push the tasks "Read", "Write", and "Code" onto the stack.
3. Pop a task from the stack.
4. Push tasks "Debug" and "Test" onto the stack.
5. Peek at the top task without removing it.
6. Print the stack.

A screenshot of a computer

Description automatically generated

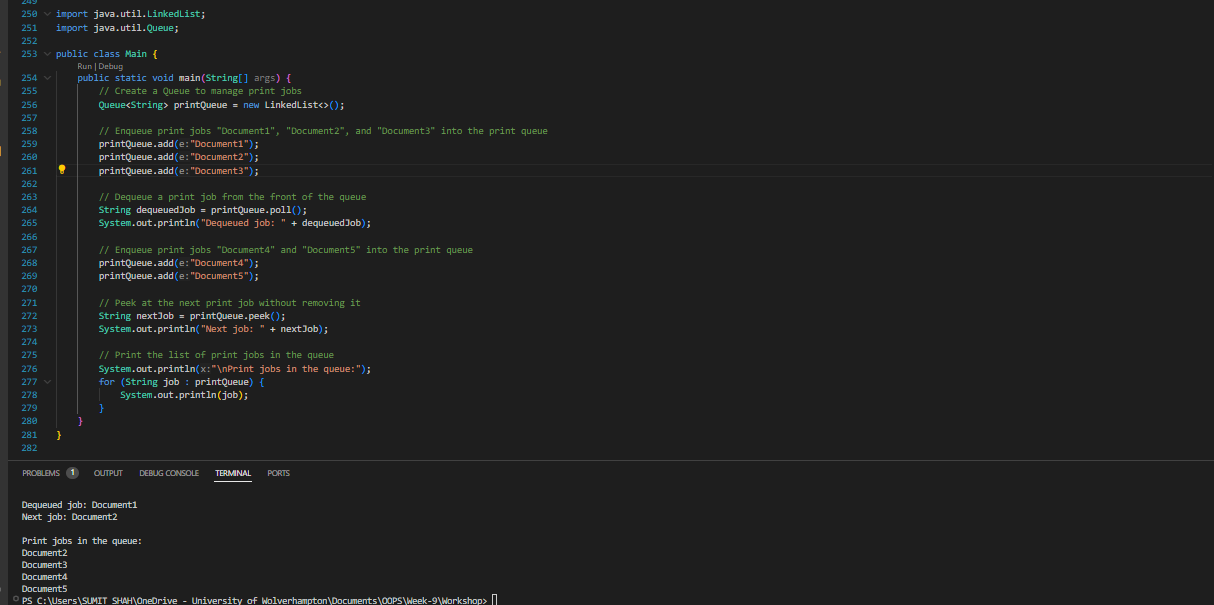
1. Write a program that reverses the order of words in a sentence using a Stack. For example, if the input is "Hello World", the output should be "World Hello".



**Queue**

1. Imagine a scenario where a printer is managing print jobs. Create a Queue to handle these print jobs. Implement the following operations:

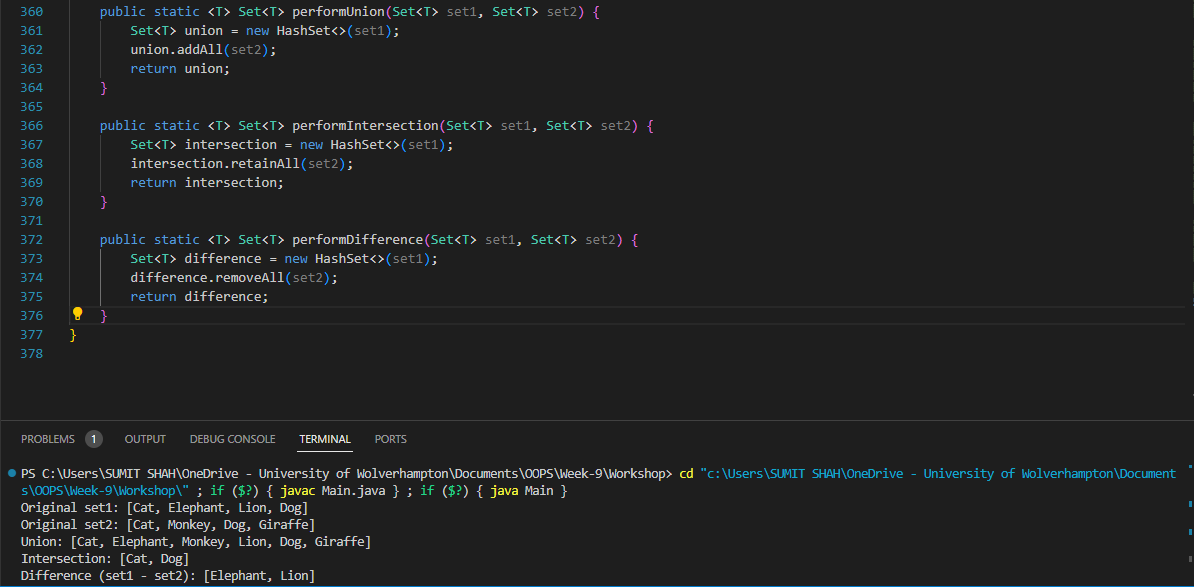
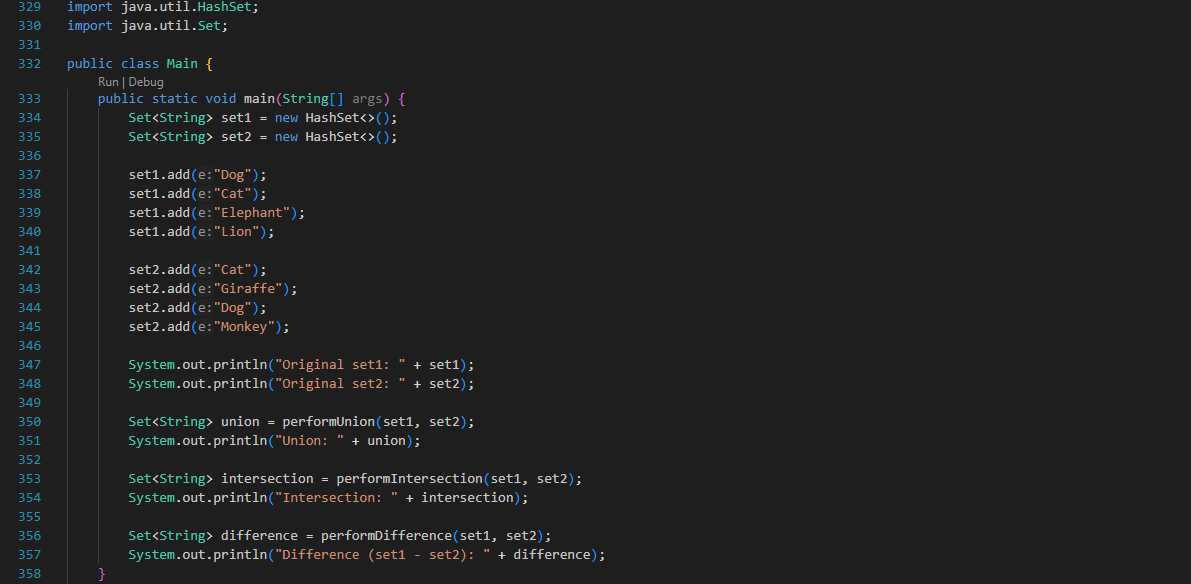
* Enqueue print jobs "Document1", "Document2", and "Document3" into the print queue.
* Dequeue a print job from the front of the queue.
* Enqueue print jobs "Document4" and "Document5" into the print queue.
* Peek at the next print job without removing it.
* Print the list of print jobs in the queue.

****

**Set Operations**

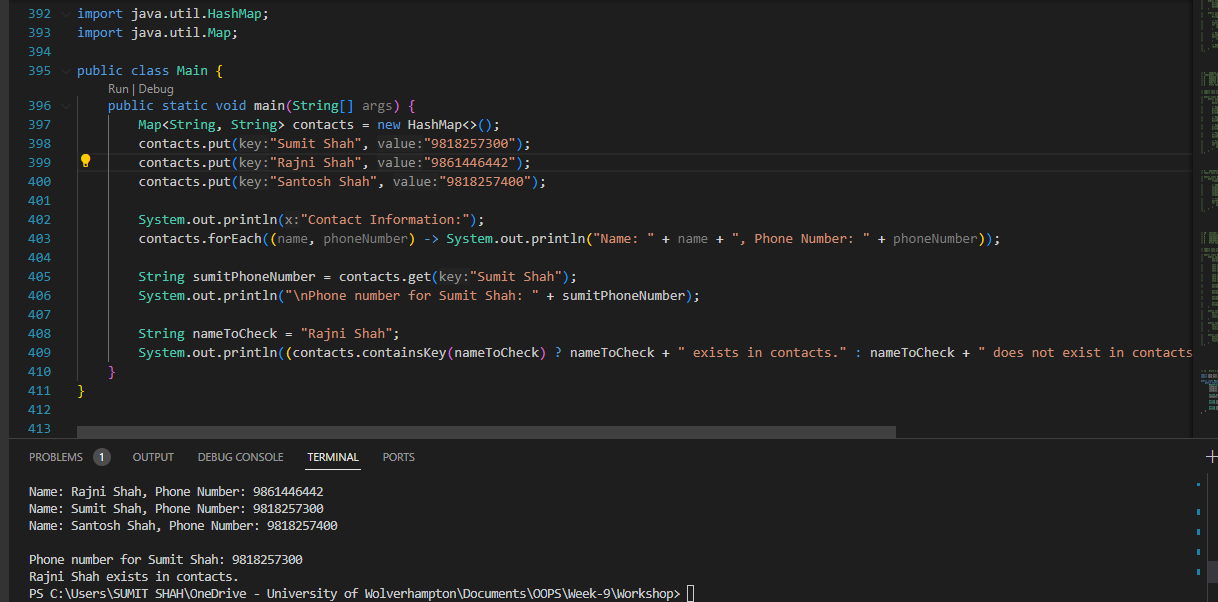
1. Implement a TreeSet to store unique names in alphabetical order.
2. Consider a scenario where you have two sets, each representing a group of animals. Implement a Java program to perform set operations (Union, Intersection, and Difference) on these sets:

* Initialize two HashSet objects: **set1** with elements "Dog," "Cat," "Elephant," and "Lion," and **set2** with elements "Cat," "Giraffe," "Dog," and "Monkey."
* Implement a method performUnion that takes two sets and returns their union.
* Implement a method performIntersection that takes two sets and returns their intersection.
* Implement a method performDifference that takes two sets and returns the difference of the first set from the second set.
* Print the original sets, the union, intersection, and difference of the sets.

****

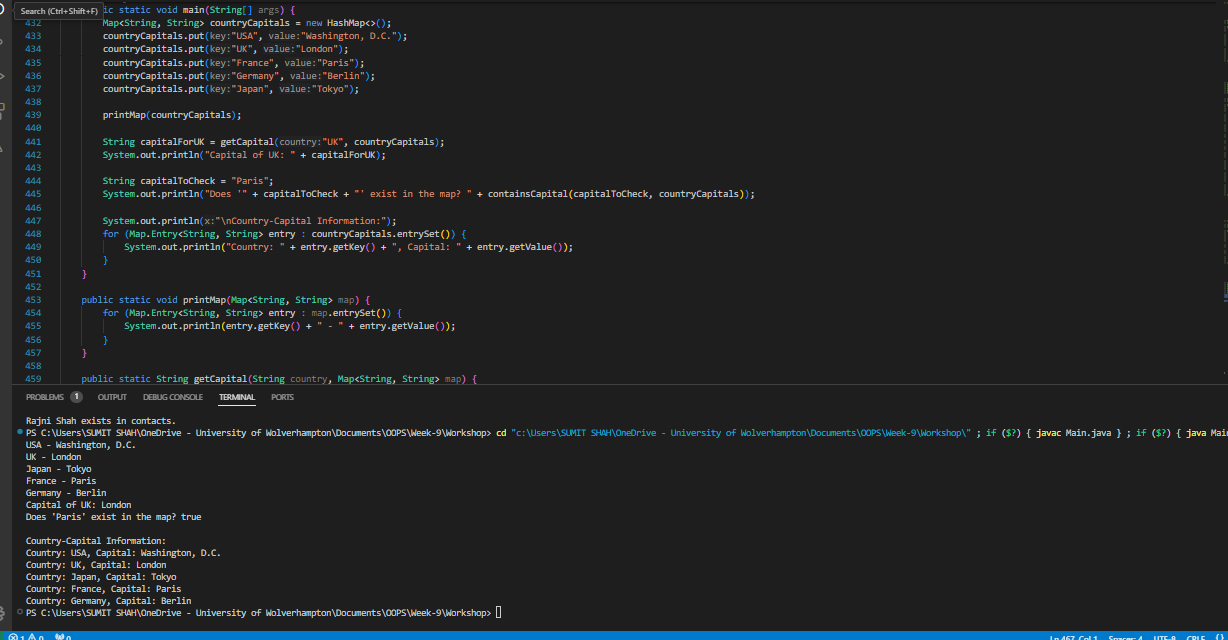
**Map(HashMap, LinkedHashMap, TreeMap):**

1. Write a program that uses a HashMap to store contact information (name and phone number).



1. Imagine a scenario where you are managing information about countries and their capitals using a HashMap. Perform the following tasks:

* Initialize a HashMap called countryCapitals to store the capitals of different countries. Add at least five country-capital pairs.
* Implement a method called printMap that takes a HashMap and prints all the key-value pairs.
* Implement a method called getCapital that takes a country name as a parameter and returns its capital from the countryCapitals map.
* Implement a method called containsCapital that takes a capital name as a parameter and returns whether that capital exists in the countryCapitals map.
* Iterate through the countryCapitals map and print each country and its capital.



**Collection Algorithm**

**Sorting**

1. Write a program that sorts an array of integers using the sort() method. Also try sorting in reverse order.

A screenshot of a computer program

Description automatically generated

1. Write a program that sorts an array list of string of colors using the sort() method. Also try sorting in reverse order.

A screenshot of a computer program

Description automatically generated

**Binary search**

1. Write a program to initialize an ArrayList with a set of integers. Implement a binary search algorithm to find a particular integer.

