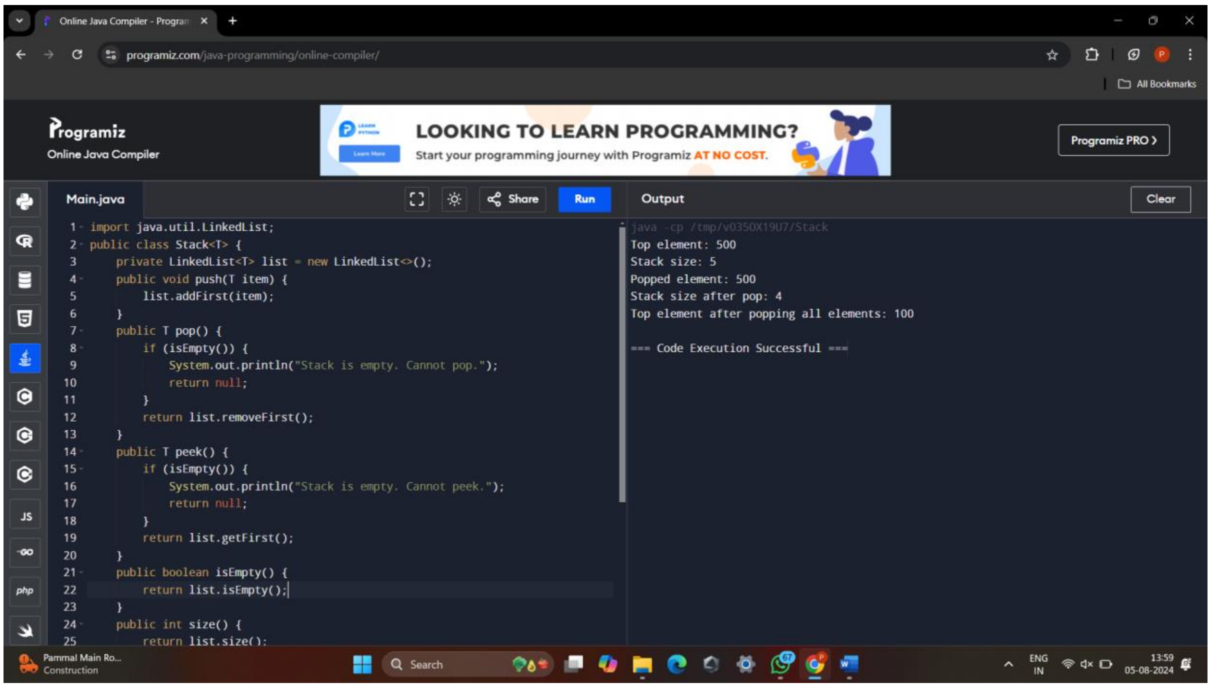
1. import java.util.LinkedList;

```
public class Stack<T> {
    private LinkedList<T> list = new LinkedList<>();
    public void push(T item) {
        list.addFirst(item);
    }
    public T pop() {
        if (isEmpty()) {
            System.out.println("Stack is empty. Cannot pop.");
            return null;
        }
        return list.removeFirst();
    }
    public T peek() {
        if (isEmpty()) {
            System.out.println("Stack is empty. Cannot peek.");
            return null;
        }
        return list.getFirst();
    }
    public boolean isEmpty() {
        return list.isEmpty();
    }
    public int size() {
        return list.size();
    }

    public static void main(String[] args) {
        Stack<Integer> stack = new Stack<>();
        stack.push(200);
        stack.push(100);
```

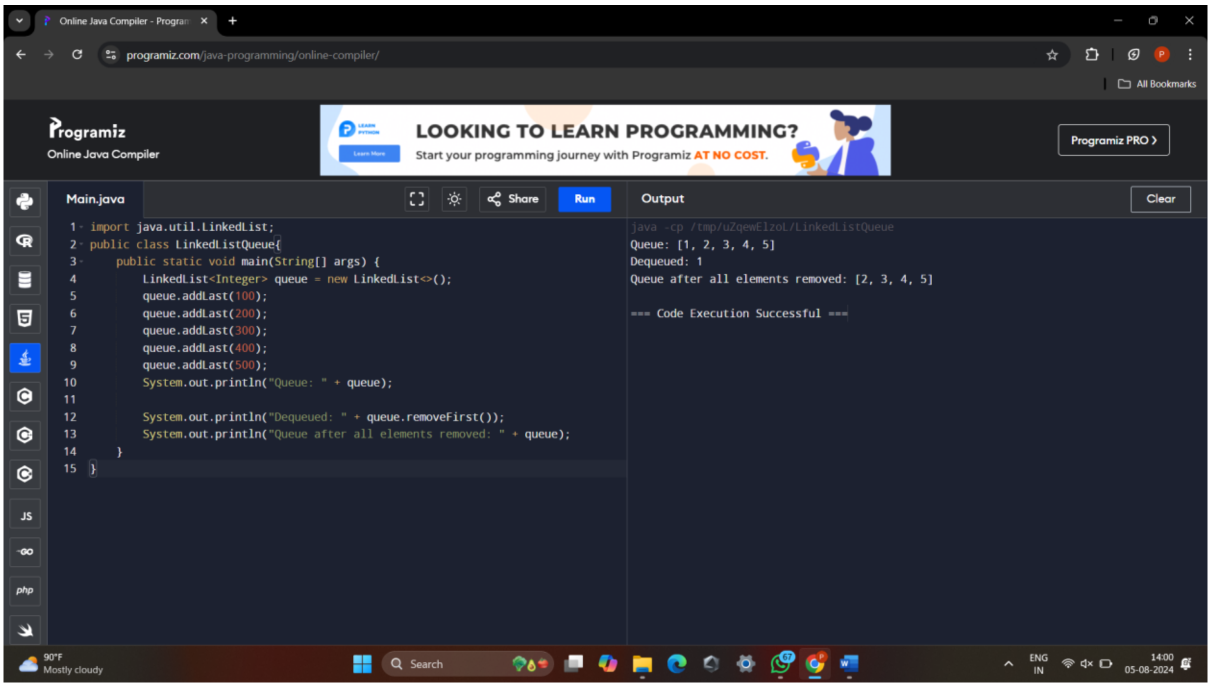
```java
        stack.push(600);
        stack.push(300);
        stack.push(500);
        System.out.println("Top element: " + stack.peek());
        System.out.println("Stack size: " + stack.size());
        System.out.println("Popped element: " + stack.pop());
        System.out.println("Stack size after pop: " + stack.size());
        stack.pop();
        stack.pop();
        System.out.println("Top element after popping all elements: " + stack.peek());
    }
}
```



```java
import java.util.LinkedList;
public class Stack<T> {
    private LinkedList<T> list = new LinkedList<>();
    public void push(T item) {
        list.addFirst(item);
    }
    public T pop() {
        if (isEmpty()) {
            System.out.println("Stack is empty. Cannot pop.");
            return null;
        }
        return list.removeFirst();
    }
    public T peek() {
        if (isEmpty()) {
            System.out.println("Stack is empty. Cannot peek.");
            return null;
        }
        return list.getFirst();
    }
    public boolean isEmpty() {
        return list.isEmpty();
    }
    public int size() {
        return list.size();
```

```
java -cp /tmp/v0350X19U7/Stack
Top element: 500
Stack size: 5
Popped element: 500
Stack size after pop: 4
Top element after popping all elements: 100

=== Code Execution Successful ===
```

2.
```java
import java.util.LinkedList;
public class LinkedListQueue{
    public static void main(String[] args) {
        LinkedList<Integer> queue = new LinkedList<>();
        queue.addLast(100);
        queue.addLast(200);
        queue.addLast(300);
        queue.addLast(400);
        queue.addLast(500);
        System.out.println("Queue: " + queue);


        System.out.println("Dequeued: " + queue.removeFirst());
        System.out.println("Queue after all elements removed: " + queue);
    }
}
```


Online Java Compiler - Program
programiz.com/java-programming/online-compiler/
All Bookmarks
Programiz
Online Java Compiler
LOOKING TO LEARN PROGRAMMING?
Start your programming journey with Programiz AT NO COST.
Programiz PRO ›
Main.java
Share
Run
Output
Clear

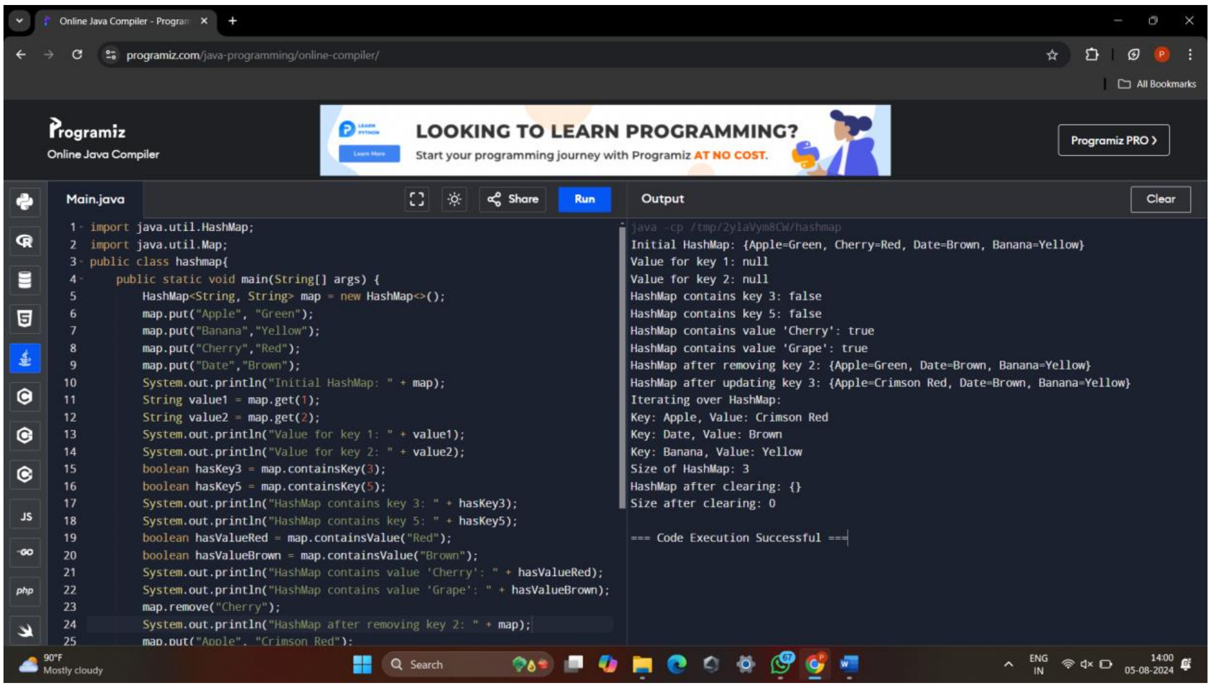
```java
import java.util.LinkedList;
public class LinkedListQueue{
    public static void main(String[] args) {
        LinkedList<Integer> queue = new LinkedList<>();
        queue.addLast(100);
        queue.addLast(200);
        queue.addLast(300);
        queue.addLast(400);
        queue.addLast(500);
        System.out.println("Queue: " + queue);

        System.out.println("Dequeued: " + queue.removeFirst());
        System.out.println("Queue after all elements removed: " + queue);
    }
}
```

```
java -cp /tmp/uZqewElzoL/LinkedListQueue
Queue: [1, 2, 3, 4, 5]
Dequeued: 1
Queue after all elements removed: [2, 3, 4, 5]

=== Code Execution Successful ===
```

3. import java.util.HashMap;

```
import java.util.Map;
public class hashmap{
    public static void main(String[] args) {
        HashMap<String, String> map = new HashMap<>();
        map.put("Apple", "Green");
        map.put("Banana","Yellow");
        map.put("Cherry","Red");
        map.put("Date","Brown");
        System.out.println("Initial HashMap: " + map);
        String value1 = map.get(1);
        String value2 = map.get(2);
        System.out.println("Value for key 1: " + value1);
        System.out.println("Value for key 2: " + value2);


        boolean hasKey3 = map.containsKey(3);
        boolean hasKey5 = map.containsKey(5);
        System.out.println("HashMap contains key 3: " + hasKey3);
        System.out.println("HashMap contains key 5: " + hasKey5);


        boolean hasValueRed = map.containsValue("Red");
        boolean hasValueBrown = map.containsValue("Brown");
        System.out.println("HashMap contains value 'Cherry': " + hasValueRed);
        System.out.println("HashMap contains value 'Grape': " + hasValueBrown);


        map.remove("Cherry");
        System.out.println("HashMap after removing key 2: " + map);
```

```java
        map.put("Apple", "Crimson Red");
        System.out.println("HashMap after updating key 3: " + map);



        System.out.println("Iterating over HashMap:");
        for (Map.Entry<String, String> entry : map.entrySet()) {
            System.out.println("Key: " + entry.getKey() + ", Value: " + entry.getValue());
        }



        int size = map.size();
        System.out.println("Size of HashMap: " + size);


        map.clear();
        System.out.println("HashMap after clearing: " + map);
        System.out.println("Size after clearing: " + map.size());
    }
}
```



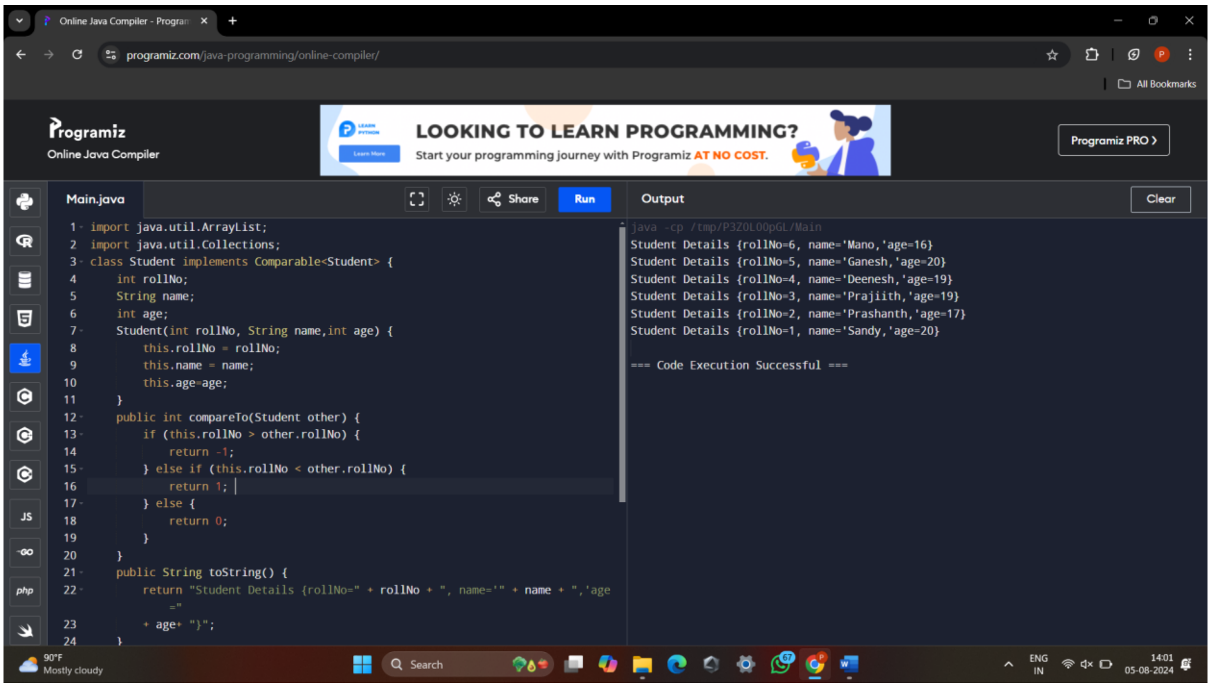
```java
import java.util.HashMap;
import java.util.Map;
public class hashmap{
    public static void main(String[] args) {
        HashMap<String, String> map = new HashMap<>();
        map.put("Apple", "Green");
        map.put("Banana","Yellow");
        map.put("Cherry","Red");
        map.put("Date","Brown");
        System.out.println("Initial HashMap: " + map);
        String value1 = map.get(1);
        String value2 = map.get(2);
        System.out.println("Value for key 1: " + value1);
        System.out.println("Value for key 2: " + value2);
        boolean hasKey3 = map.containsKey(3);
        boolean hasKey5 = map.containsKey(5);
        System.out.println("HashMap contains key 3: " + hasKey3);
        System.out.println("HashMap contains key 5: " + hasKey5);
        boolean hasValueRed = map.containsValue("Red");
        boolean hasValueBrown = map.containsValue("Brown");
        System.out.println("HashMap contains value 'Cherry': " + hasValueRed);
        System.out.println("HashMap contains value 'Grape': " + hasValueBrown);
        map.remove("Cherry");
        System.out.println("HashMap after removing key 2: " + map);
        map.put("Apple", "Crimson Red");
```

```
java -cp /tmp/2ylaVym8CW/hashmap
Initial HashMap: {Apple=Green, Cherry=Red, Date=Brown, Banana=Yellow}
Value for key 1: null
Value for key 2: null
HashMap contains key 3: false
HashMap contains key 5: false
HashMap contains value 'Cherry': true
HashMap contains value 'Grape': true
HashMap after removing key 2: {Apple=Green, Date=Brown, Banana=Yellow}
HashMap after updating key 3: {Apple=Crimson Red, Date=Brown, Banana=Yellow}
Iterating over HashMap:
Key: Apple, Value: Crimson Red
Key: Date, Value: Brown
Key: Banana, Value: Yellow
Size of HashMap: 3
HashMap after clearing: {}
Size after clearing: 0

=== Code Execution Successful ===
```

Online Java Compiler - Programiz
programiz.com/java-programming/online-compiler/
Programiz
Online Java Compiler
LOOKING TO LEARN PROGRAMMING?
Start your programming journey with Programiz AT NO COST.
Programiz PRO >
Main.java
Share
Run
Output
Clear
90°F
Mostly cloudy
Search
ENG IN
14:00
05-08-2024

```
4. import java.util.ArrayList;
import java.util.Collections;
class Student implements Comparable<Student> {
   int rollNo;
   String name;
   int age;
   Student(int rollNo, String name,int age) {
      this.rollNo = rollNo;
      this.name = name;
      this.age=age;
   }
   public int compareTo(Student other) {
      if (this.rollNo > other.rollNo) {
         return -1;
      } else if (this.rollNo < other.rollNo) {
         return 1;
      } else {
         return 0;
      }
   }
   public String toString() {
      return "Student Details {rollNo=" + rollNo + ", name='" + name + "','age="
      + age+ "}";
   }
}

public class Main {
   public static void main(String[] args) {
      ArrayList<Student> students = new ArrayList<>();
      students.add(new Student(3, "Prajiith",19));
      students.add(new Student(1, "Sandy",20));
```

```
        students.add(new Student(2, "Prashanth",17));
        students.add(new Student(6, "Mano",16));
        students.add(new Student(4, "Deenesh",19));
        students.add(new Student(5, "Ganesh",20));

        Collections.sort(students);

        for (Student student : students) {
            System.out.println(student);
        }
    }
}
```


Online Java Compiler - Program
programiz.com/java-programming/online-compiler/
All Bookmarks
Programiz
Online Java Compiler
LOOKING TO LEARN PROGRAMMING?
Start your programming journey with Programiz AT NO COST.
Programiz PRO
Main.java
Share
Run
Output
Clear
90°F
Mostly cloudy
Search
ENG IN
14:01
05-08-2024

```
import java.util.ArrayList;
import java.util.Collections;
class Student implements Comparable<Student> {
    int rollNo;
    String name;
    int age;
    Student(int rollNo, String name,int age) {
        this.rollNo = rollNo;
        this.name = name;
        this.age=age;
    }
    public int compareTo(Student other) {
        if (this.rollNo > other.rollNo) {
            return -1;
        } else if (this.rollNo < other.rollNo) {
            return 1;
        } else {
            return 0;
        }
    }
    public String toString() {
        return "Student Details {rollNo=" + rollNo + ", name='" + name + ",'age
            ="
        + age+ "}";
    }
```

```
java -cp /tmp/P3ZOLOOpGL/Main
Student Details {rollNo=6, name='Mano,'age=16}
Student Details {rollNo=5, name='Ganesh,'age=20}
Student Details {rollNo=4, name='Deenesh,'age=19}
Student Details {rollNo=3, name='Prajiith,'age=19}
Student Details {rollNo=2, name='Prashanth,'age=17}
Student Details {rollNo=1, name='Sandy,'age=20}

=== Code Execution Successful ===
```