6.Implement the following Data structures in Java

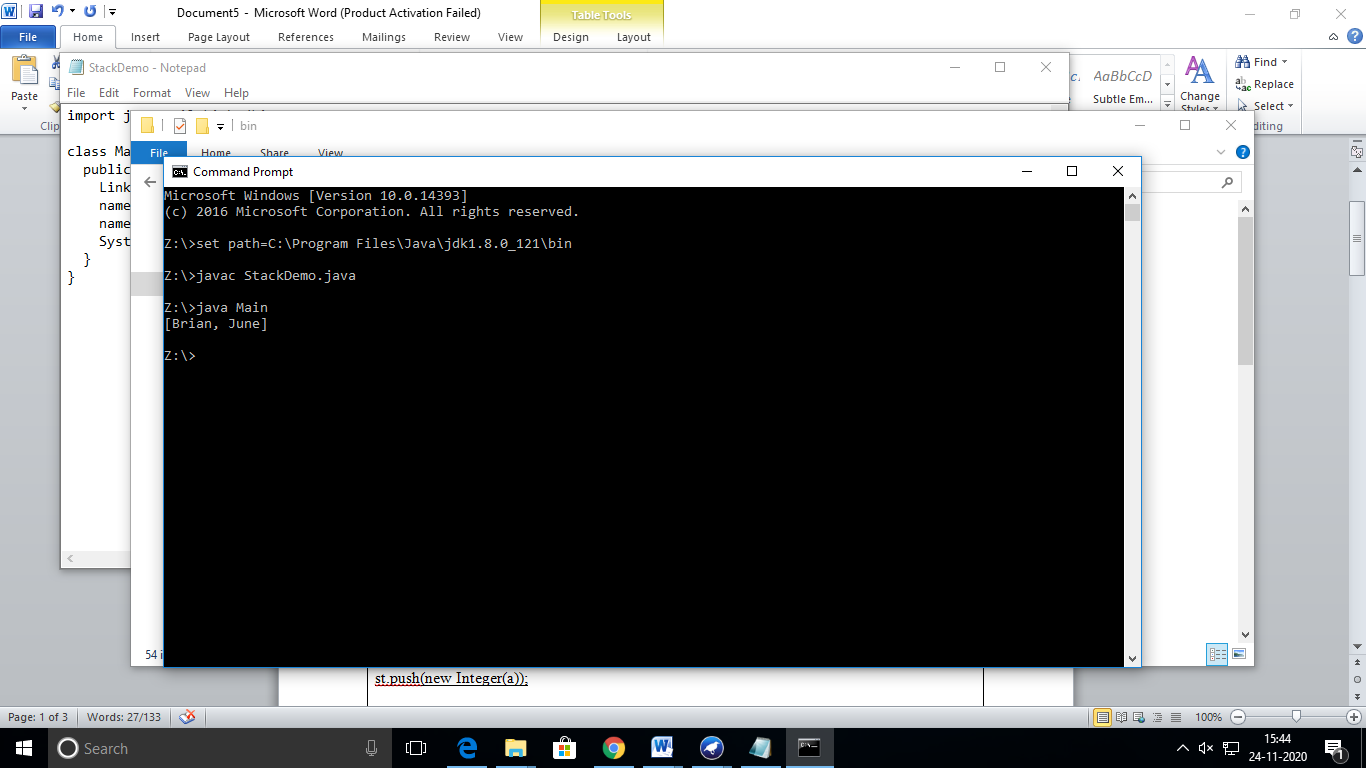
a)Linked Lists b) Stacks c) Queues d) Set e) Map

a)Linked Lists :

program:

|  |
| --- |
| import java.util.LinkedList;  class Main {  public static void main(String[] args) {  LinkedList<String> names = new LinkedList<String>();  names.add("Brian");  names.add("June");  System.out.println(names); // This will output [Brian, June]  }  } |

Output:

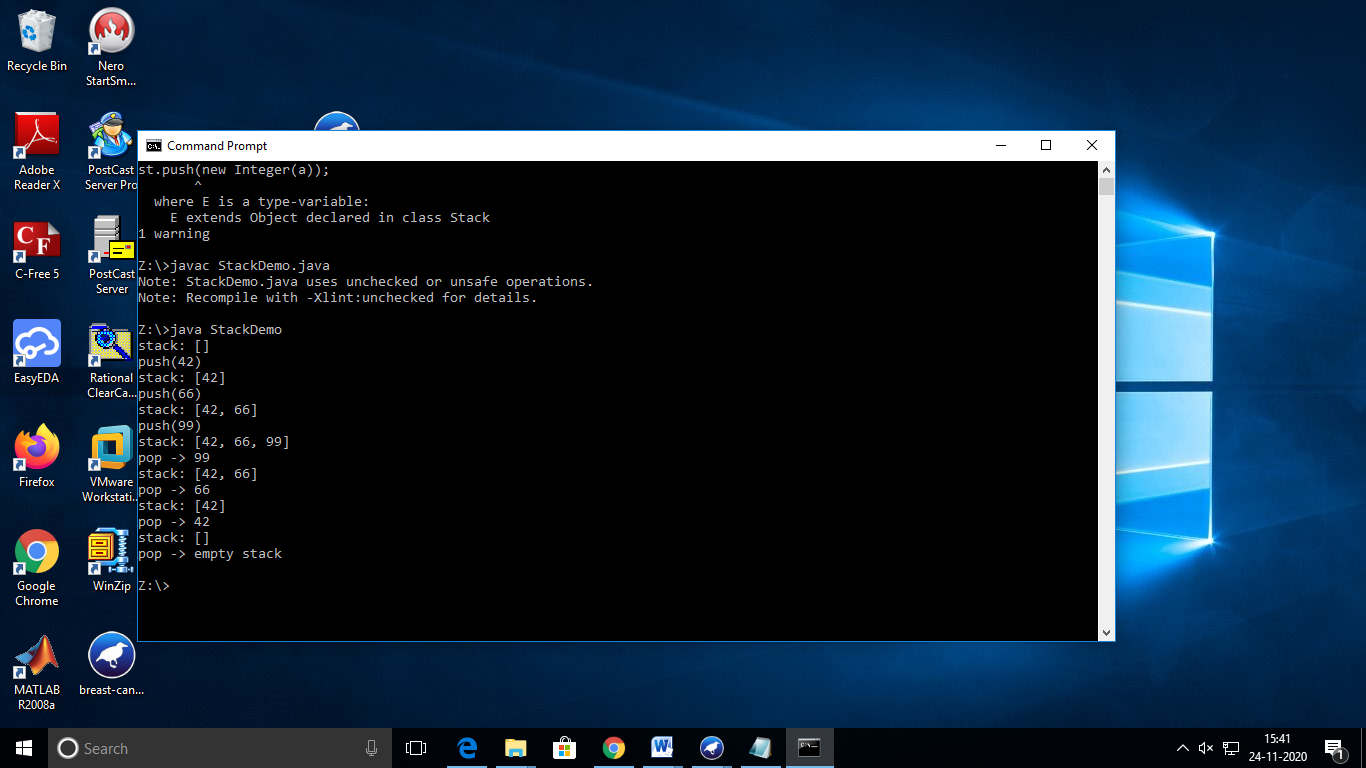


b) Stacks:

prgrams:

|  |
| --- |
| import java.util.\*;  public class StackDemo {  static void showpush(Stack st, int a) {  st.push(new Integer(a));  System.out.println("push(" + a + ")");  System.out.println("stack: " + st);  }  static void showpop(Stack st) {  System.out.print("pop -> ");  Integer a = (Integer) st.pop();  System.out.println(a);  System.out.println("stack: " + st);  }  public static void main(String args[]) {  Stack st = new Stack();  System.out.println("stack: " + st);  showpush(st, 42);  showpush(st, 66);  showpush(st, 99);  showpop(st);  showpop(st);  showpop(st);  try {  showpop(st);  }catch (EmptyStackException e) {  System.out.println("empty stack");  }  }  } |

Output:

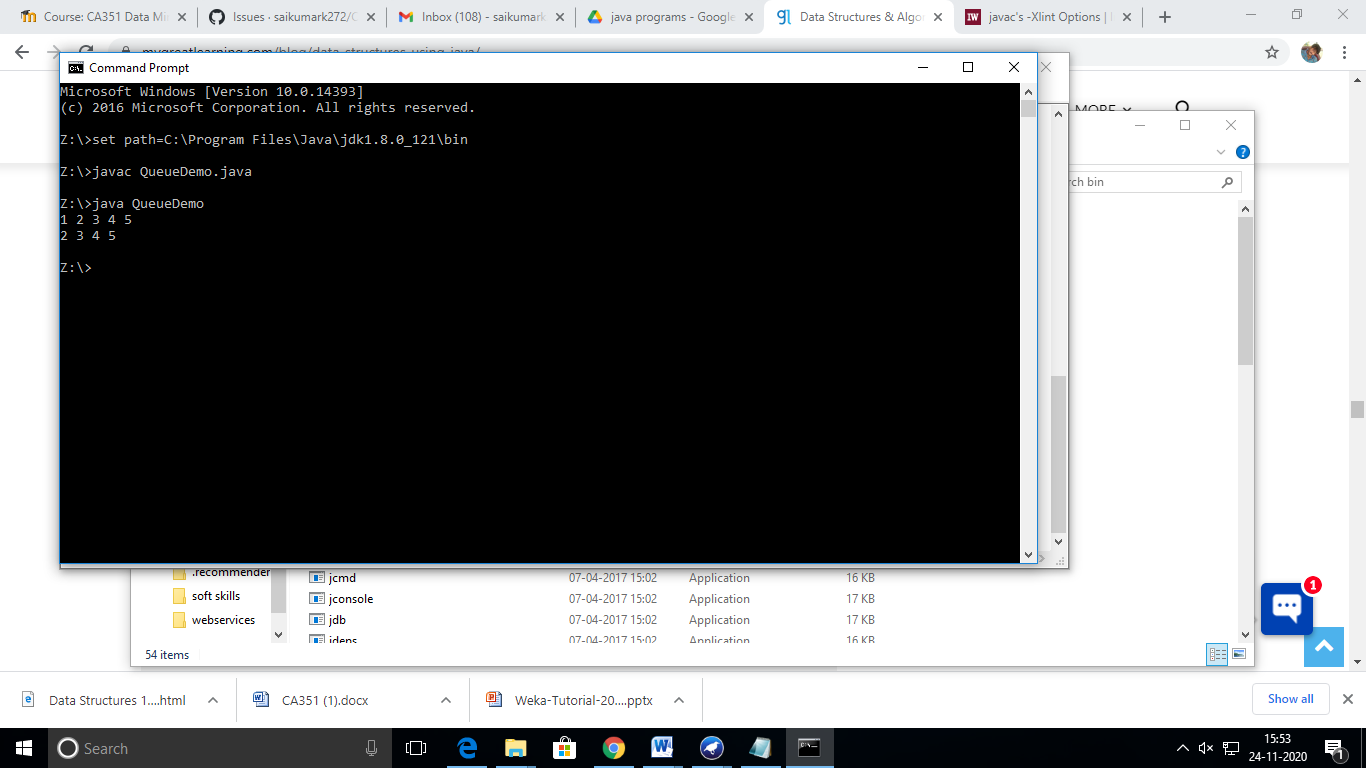


c) Queues :

program:

|  |
| --- |
| import java.util.\*;  class Queue{  int front;  int rear;  int[] arr;    Queue()  {  front=rear=-1;  arr=new int[10];  }    void enqueue(int a)  {  if(rear==arr.length-1)  System.out.println("overflow");  else  arr[++rear]=a;    if(front==-1)  front++;  }    int dequeue()  {  int x=-1;  if(front==-1)  System.out.println("underflow");  else  x=arr[front++];  if(rear==0)  rear--;  return x;  }    void display()  {  for(int i=front;i<=rear;i++)  System.out.print(arr[i]+" ");  System.out.println();  }  }  public class QueueDemo{  public static void main(String[] args)  {  Queue ob=new Queue();  ob.enqueue(1);  ob.enqueue(2);  ob.enqueue(3);  ob.enqueue(4);  ob.enqueue(5);  ob.display();  ob.dequeue();  ob.display();  }  } |

Output:

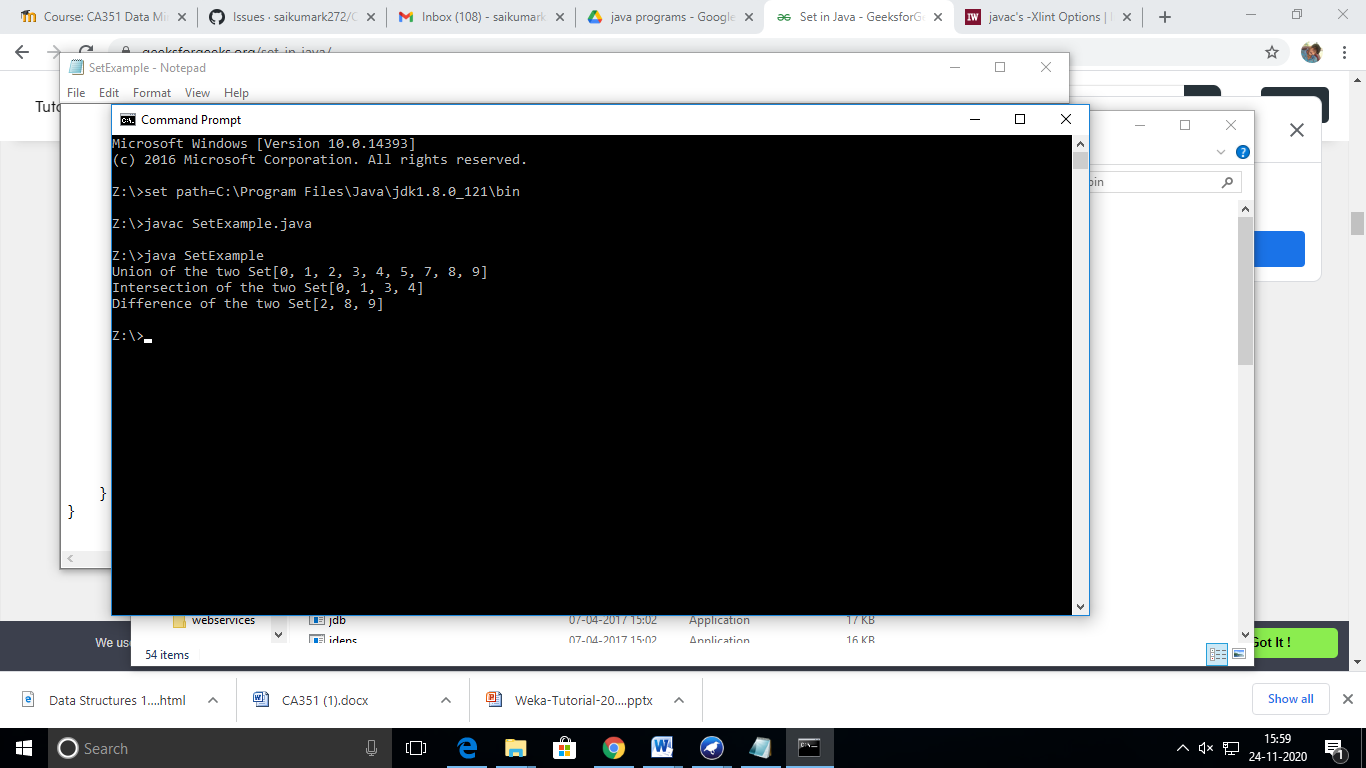


d) Set:

program:

|  |
| --- |
| import java.util.\*;  public class SetExample  {  public static void main(String args[])  {  Set<Integer> a = new HashSet<Integer>();  a.addAll(Arrays.asList(new Integer[] {1, 3, 2, 4, 8, 9, 0}));  Set<Integer> b = new HashSet<Integer>();  b.addAll(Arrays.asList(new Integer[] {1, 3, 7, 5, 4, 0, 7, 5}));    // To find union  Set<Integer> union = new HashSet<Integer>(a);  union.addAll(b);  System.out.print("Union of the two Set");  System.out.println(union);    // To find intersection  Set<Integer> intersection = new HashSet<Integer>(a);  intersection.retainAll(b);  System.out.print("Intersection of the two Set");  System.out.println(intersection);    // To find the symmetric difference  Set<Integer> difference = new HashSet<Integer>(a);  difference.removeAll(b);  System.out.print("Difference of the two Set");  System.out.println(difference);  }  } |

Output:



e) Map:

program:

|  |
| --- |
| import java.awt.Color;  import java.util.HashMap;  import java.util.Map;  import java.util.Set;  public class MapDemo  {  public static void main(String[] args)  {  Map<String, Color> favoriteColors = new HashMap<String, Color>(); favoriteColors.put("sai", Color.BLUE); favoriteColors.put("Ram", Color.GREEN); favoriteColors.put("krishna", Color.RED);  favoriteColors.put("narayana", Color.BLUE); // Print all keys and values in the map  Set<String> keySet = favoriteColors.keySet(); for (String key : keySet)  {  Color value = favoriteColors.get(key);  System.out.println(key + " : " + value);  }  }  } |

Output:

