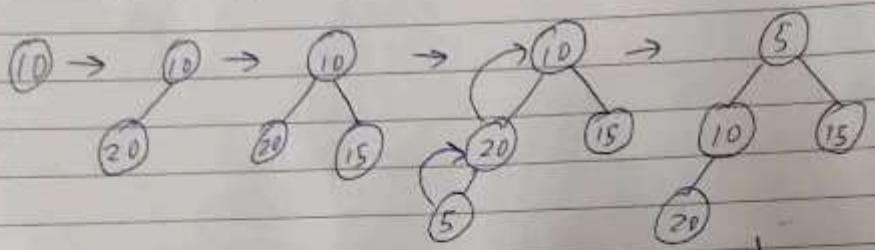
Assignment-8
Collections-2

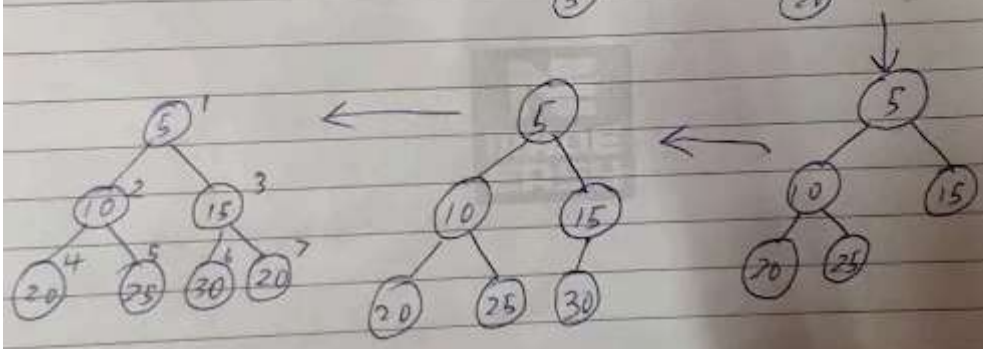
Q1. In the following Java program, two PriorityQueues are created—one for Integer and one for Character with a reverse order comparator.

```
public class PriorityQueueImplementation {

    public static void main(String[] args)
    {

        PriorityQueue<Integer> pQueue = new PriorityQueue<Integer>();
        pQueue.add(10);
        pQueue.add(20);
        pQueue.add(15);
        pQueue.add(5);
        pQueue.add(25);
        pQueue.add(30);
        pQueue.add(20);

        PriorityQueue<Character> pqc = new PriorityQueue<>(Comparator.reverseOrder());
        pqc.add('B');
        pqc.add('C');
        pqc.add('A');
        pqc.add('B');
        pqc.add('b');
        pqc.add('a');
        pqc.add('d');
        pqc.add('c');

        System.out.println("Character Priority Queue: " + pqc);

        System.out.println("Integer Priority Queue: " + pQueue);

        pqc.offer('F');
        pqc.remove('d');
        System.out.println("Character Priority Queue after applying offer & remove: " + pqc);

        System.out.println("Priority Queue After Poll: " + pqc);

        System.out.println("Priority Queue After Peek: " + pqc);

        System.out.println("Priority Queue After Element: " + pqc);
    }
}
```

8 Priority queue makes min heap and ~~return~~ return it when priority queue object is called.

Nos. added →
10, 20, 15, 5, 25, 30, 20



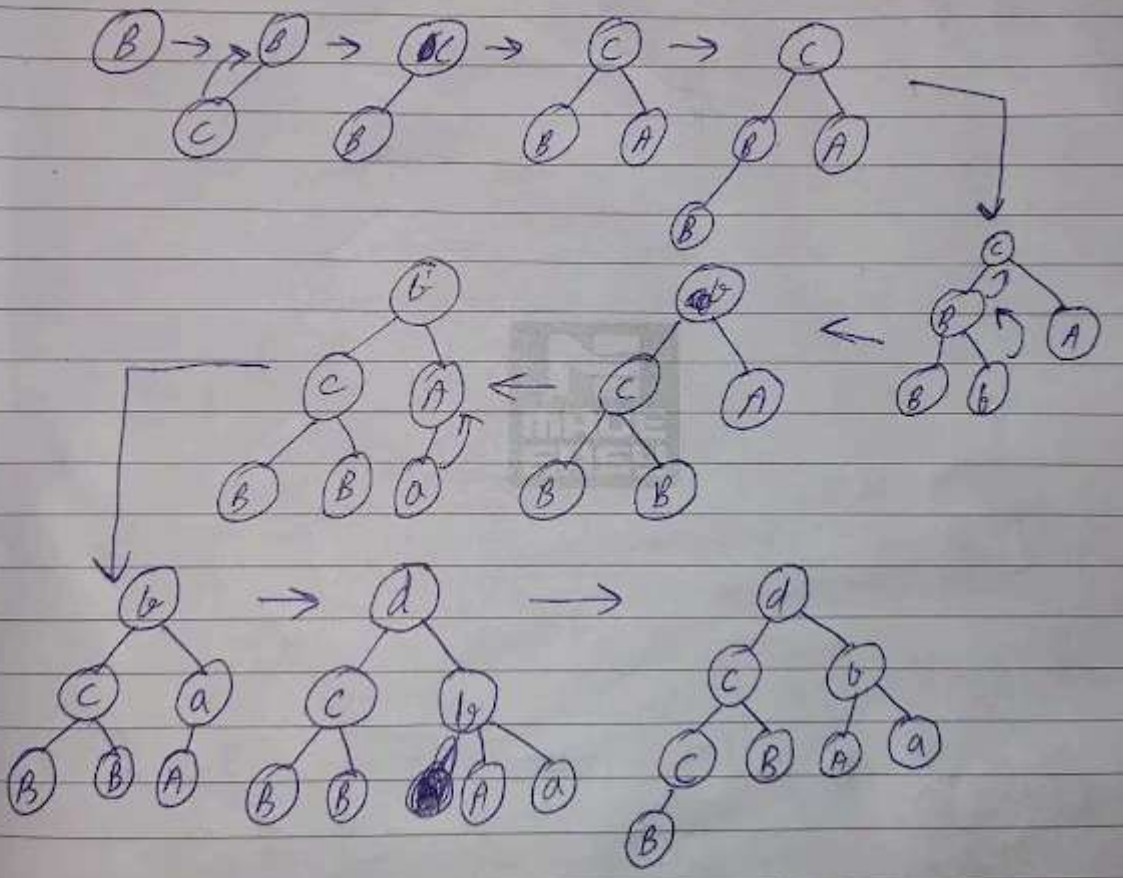


[5, 10, 15, 20, 25, 30, 20]

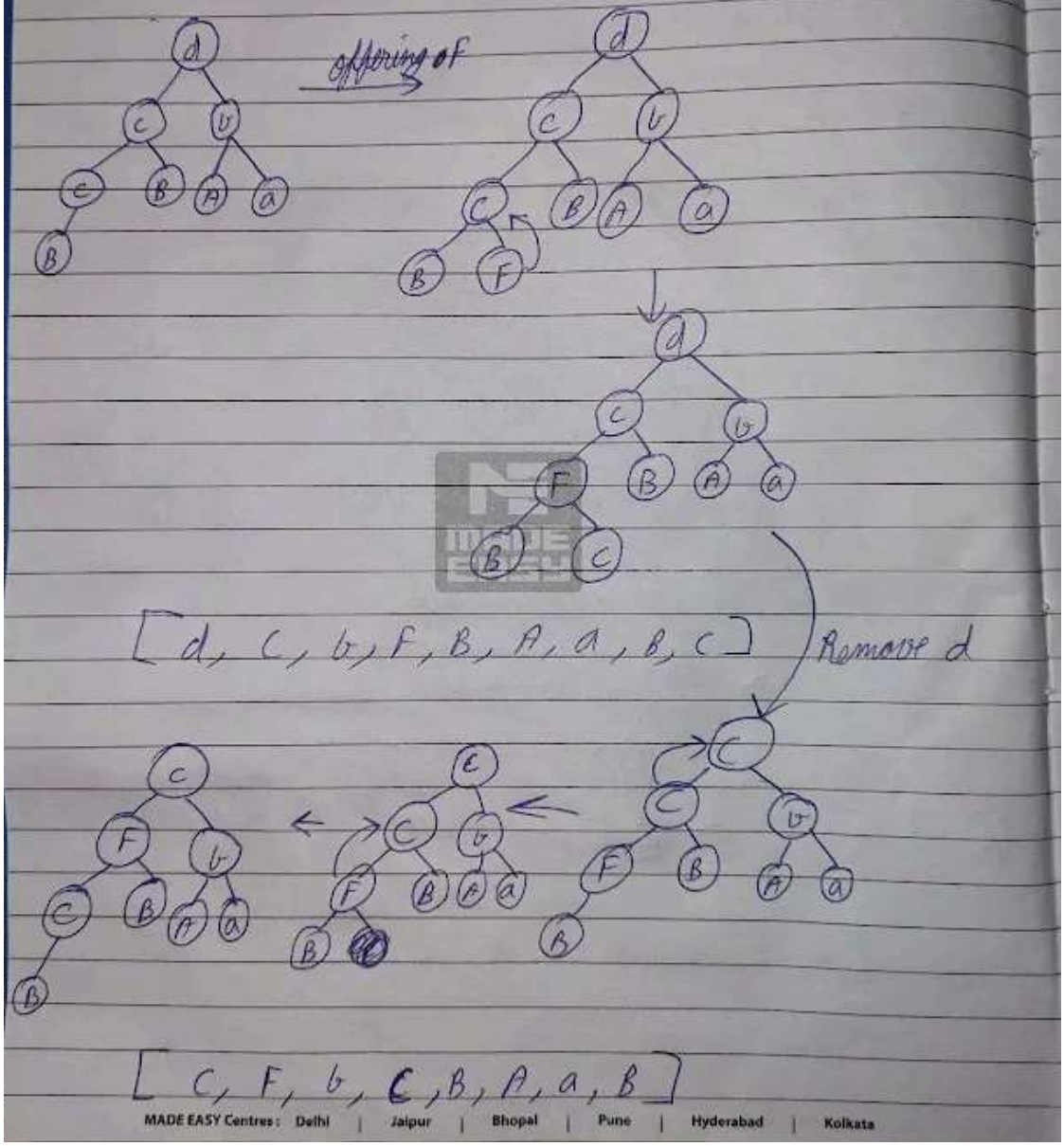
Characters added →

B, C, A, B, b, a, d, c

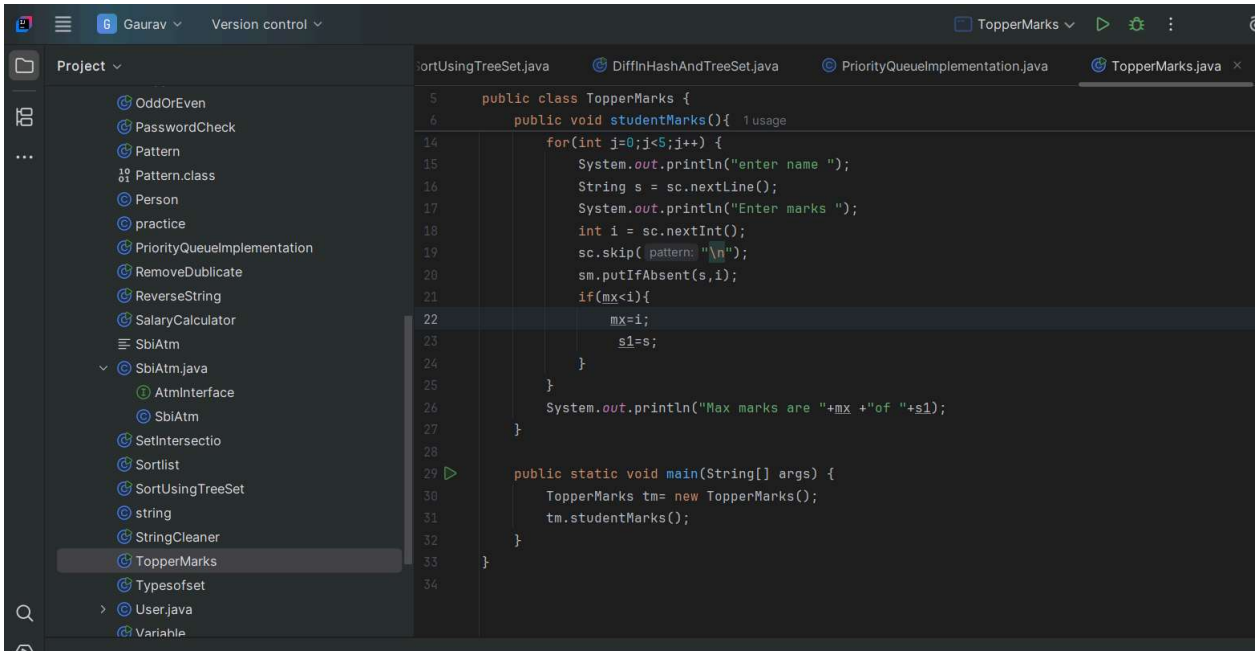
as we have used comparator to reverse order the priority key will make max heap.

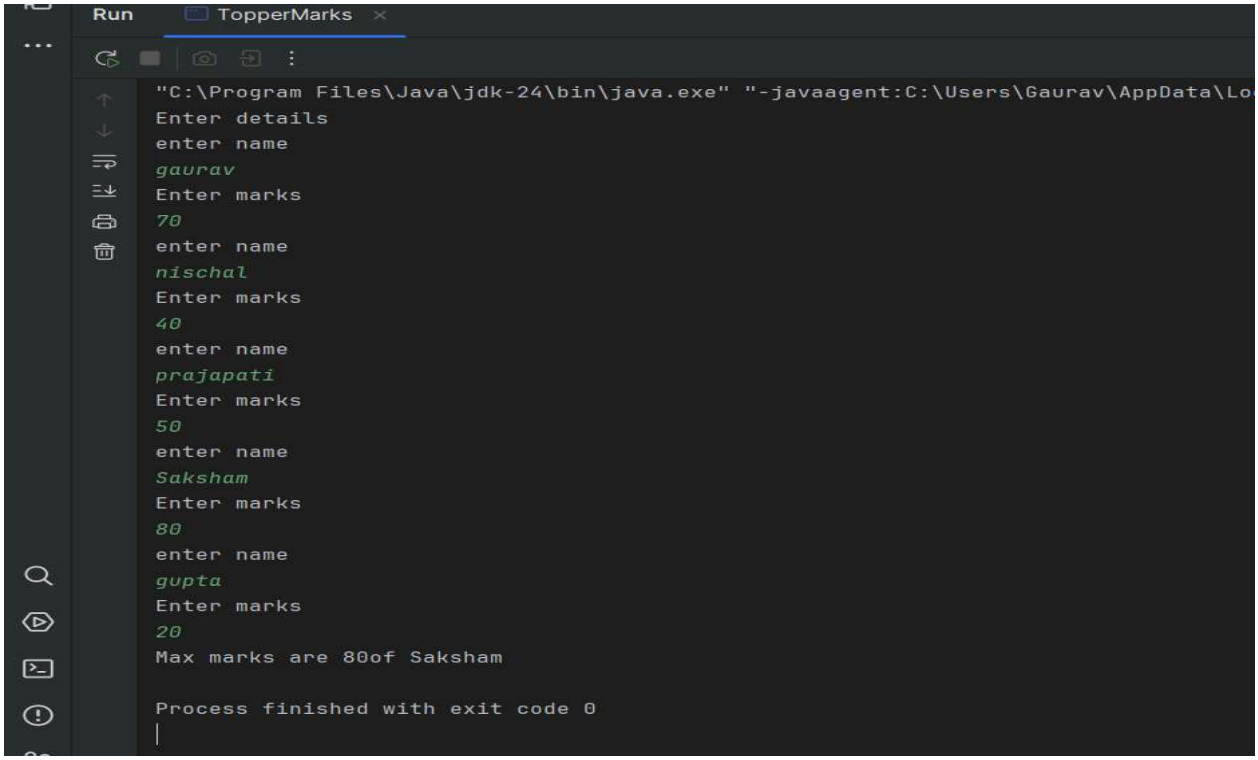


[d, C, b, C, B, A, a, B]

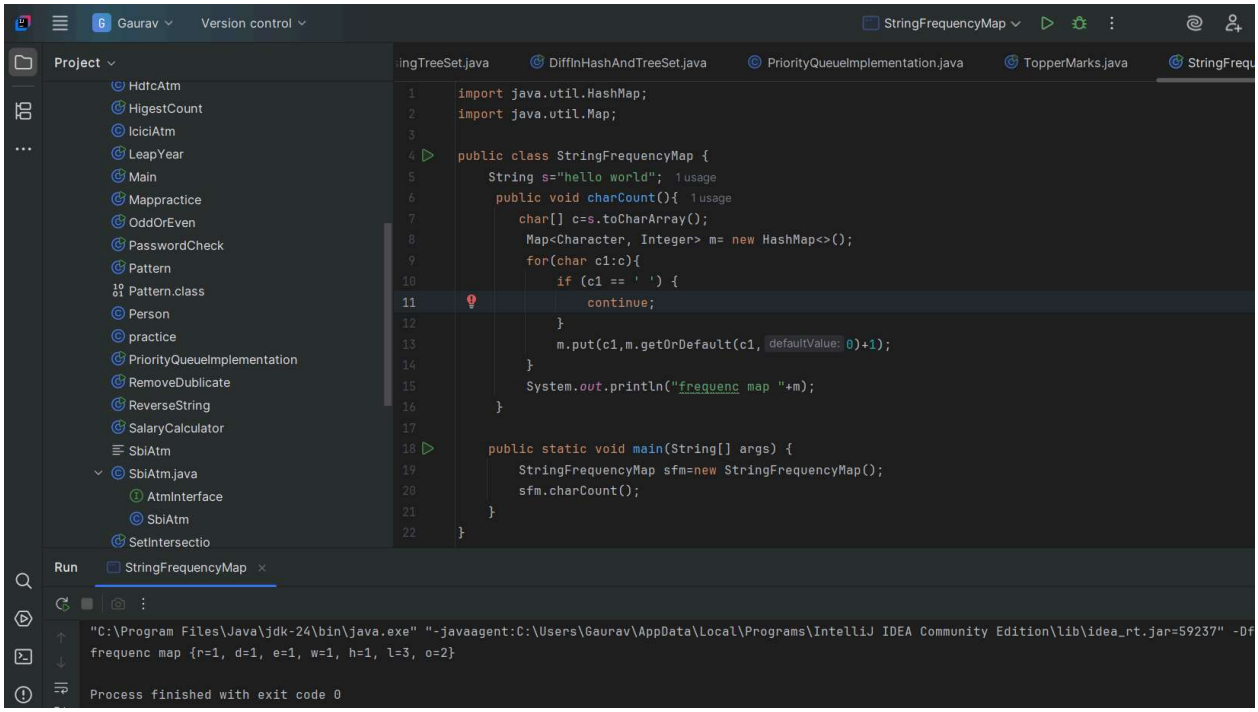

offering of
[d, C, b, F, B, A, a, B, C]
Remove d
[C, F, b, C, B, A, a, B]

Q2.WAP that stores student names and their marks, adds 5 students, and print the topper student name(i.e., the one with the highest marks):

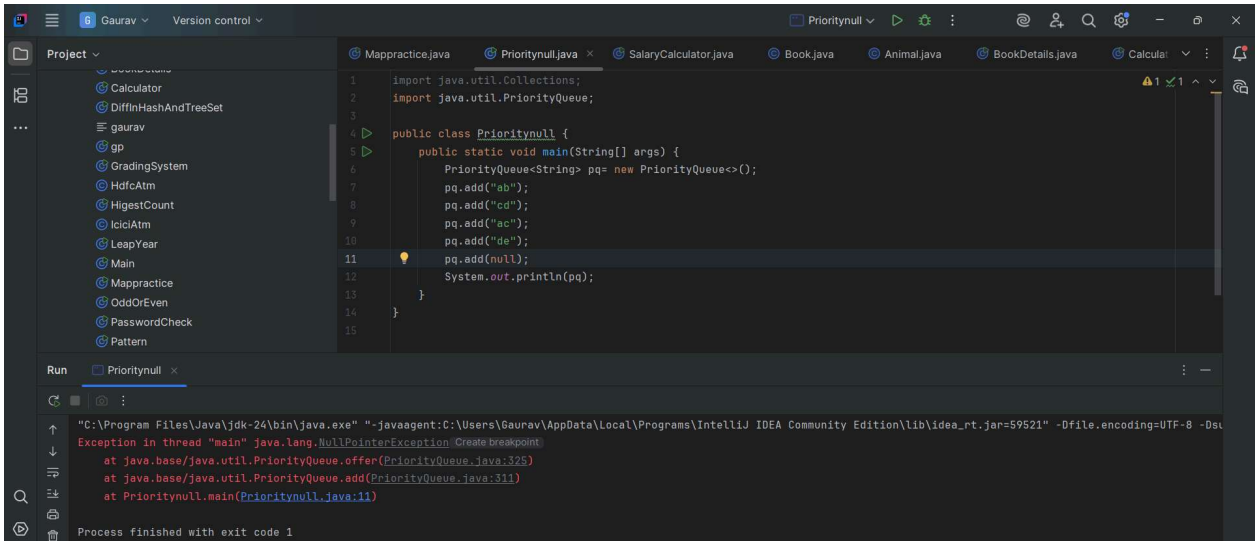




Q3.Write a method that returns a Map of character frequency in a string.
Take Input as: hello world
Output should be:
h : 1
e : 1
l : 3
o : 2
w : 1
r : 1
d : 1



Q4.Can you store null elements in a PriorityQueue? Explain with a coding example.

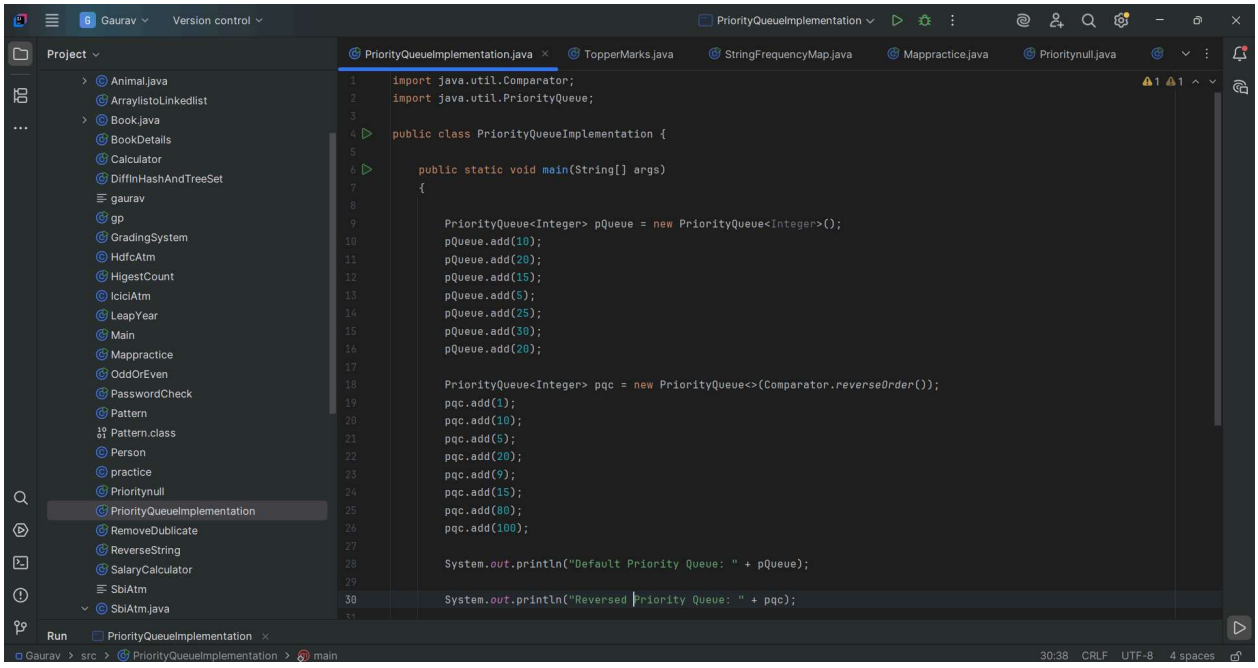


Priority Queue does not accept null, as priority queue makes a min heep which requires comparison but if we provide null which cannot be compared leads to problem in comparison due to which heap formation is not passible.

Q5.What is the default ordering in a PriorityQueue if no comparator is provided? Explain with the help of coding example

Ans. Priority queue uses the min heap by default which leads element to ordered in ascending order.

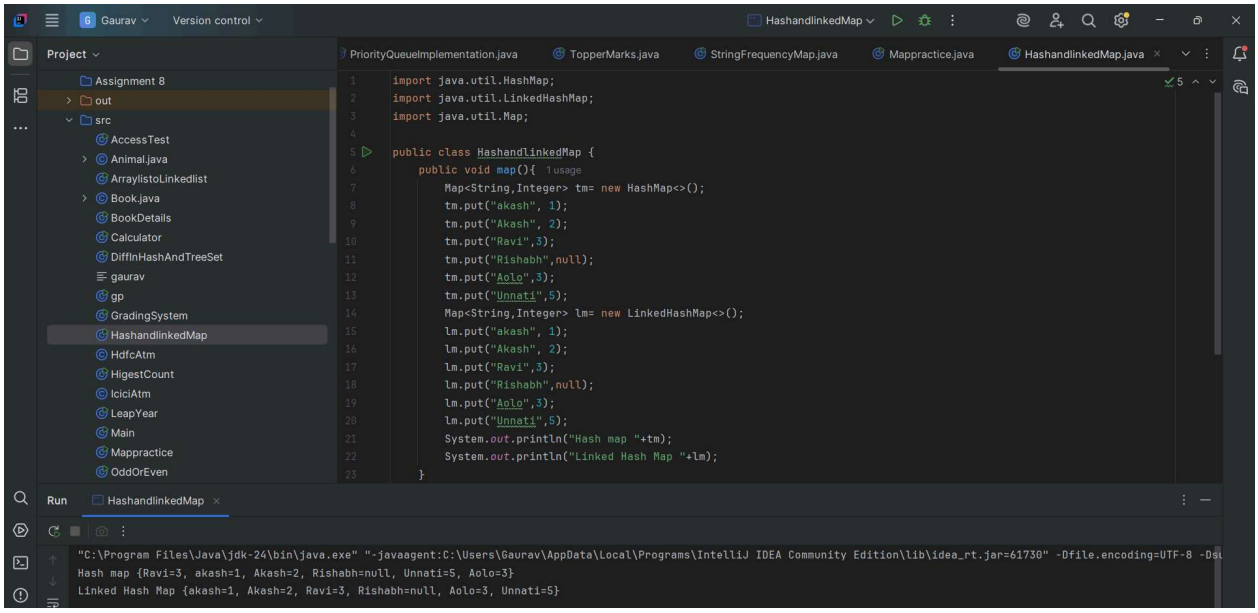
To reverse the order means to arrange in decreasing order we need to use comparator or collection.reverse().



Q6.In which cases would you prefer LinkedHashMap over HashMap? Explain with a coding example

Ans. The main difference between LinkedHashMap and HashMap is that Linked hash map maintain the insertion order whereas hash map do not maintain the insertion order. We can use the linked hashmap when we need to maintain the insertion order.

For example we can use linked hash map to maintain the transaction history like type of transaction and amount it will only maintain last debit and credit.We can also use it to implement LRU cash.



Q7.What is a Queue? How is it different from a Stack?

Ans. Queue is basically a liner data structure which is based on first in first out concept. The element which is entered first is taken out first.

The main difference between stack and queue is that stack is lifo(Last in First out) and queue is fifo(first in first out).

In Java queues implement Queue Interface of collections whereas stack implements list interface.