Name – Yash Daga Roll Number – 20BCE7323

Lab-12

Write a program to perform Insert, Delete, and Search operations on a Hash table using division method. Based on the user choice , 1. Linear Probing, 2.Quadratic Probing, resolve the collisions.

Code:

```
import java.util.*;

class HashTable {


	static int maxSize = 7;

	// Function to print an array

	static void printArray(int arr[])

	{

		for (int i = 0; i < arr.length; i++) {

			System.out.print(arr[i] + " ");

		}

	}


	private static int hash(int key)

  {

    return key % maxSize;

  }


	public static void insertLinearProbing(int[] hashTable, int num)

  {

    int tmp = hash(num);

    int i = tmp;
```

```
    do {
        if (hashTable[i] == -1) {
                                hashTable[i] = num;
            return;
        }
        i = (i + 1) % maxSize;
    }
    while (i != tmp);
}

      public static void deleteLinearProbing(int[] hashTable, int del)
{
    int i = hash(del);
                while(hashTable[i] != del) {
                        i = (i + 1) % maxSize;
                }
                hashTable[i] = -1;

                for(i = (i + 1) % maxSize; hashTable[i] != -1; i = (i + 1) % maxSize) {
                        int tmp = hashTable[i];
                        hashTable[i] = -1;
                        insertLinearProbing(hashTable, tmp);
                }
}

      public static boolean searchLinearProbing(int[] hashTable, int search)
```

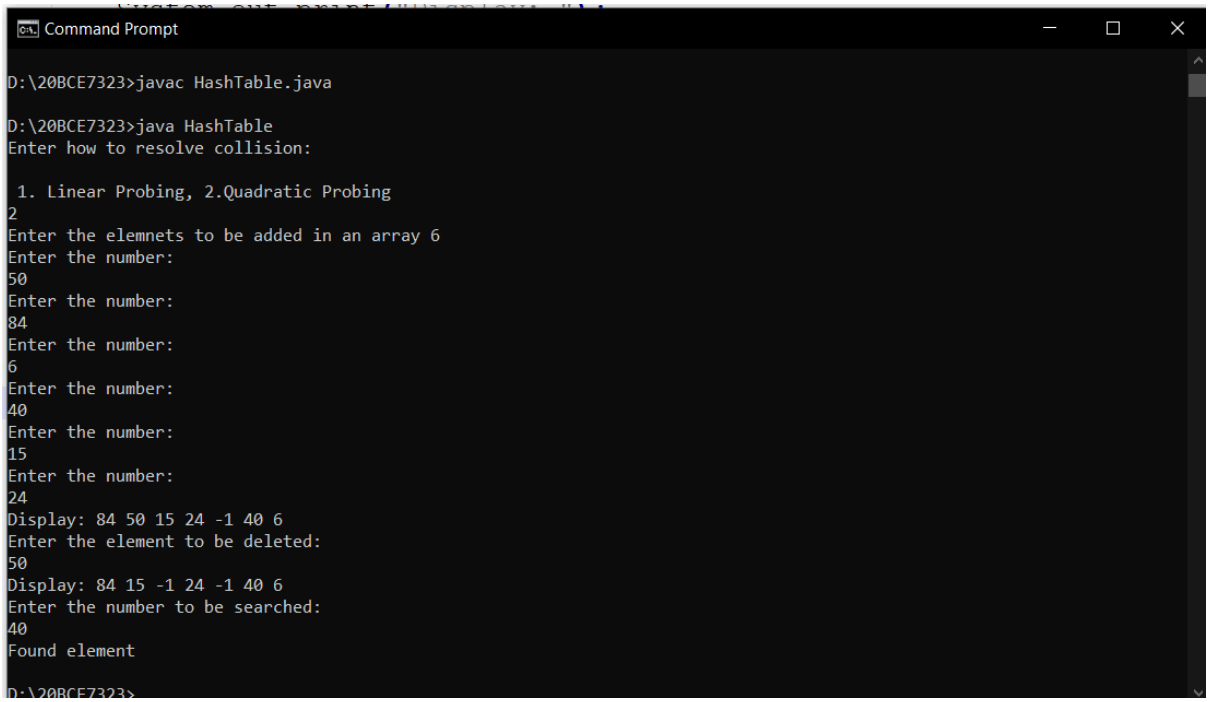
```
{
   int i = hash(search);
   while (hashTable[i] != -1) {
      if (hashTable[i] == search)
         return true;
      i = (i + 1) % maxSize;
   }
   return false;
}

      public static void insertQuadraticProbing(int[] hashTable, int num)
{
   int tmp = hash(num);
   int i = tmp, h = 1;
   do
      {
                              if(hashTable[i] == -1) {
                                       hashTable[i] = num;
                                       return;
                              }
         i = (i + h * h++) % maxSize;
      }
   while (i != tmp);
}

      public static void deleteQuadraticProbing(int[] hashTable, int del)
```

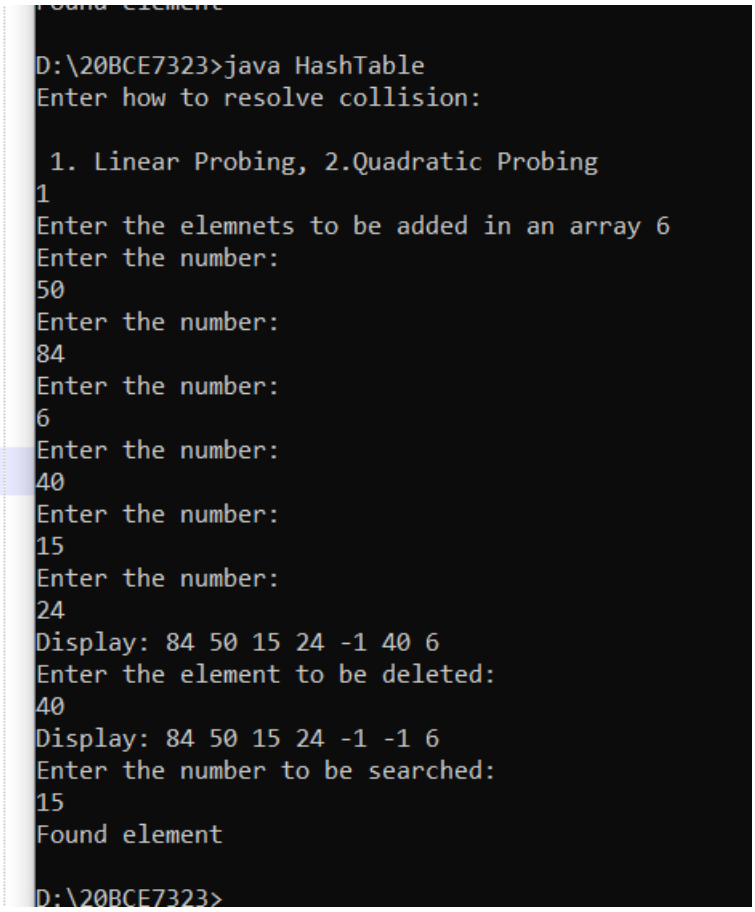
```
    {
            int i = hash(del), h = 1;
            while(hashTable[i] != del) {
                i = (i + h * h++) % maxSize;
            }
            hashTable[i] = -1;

            for(i = (i + h * h++) % maxSize; hashTable[i] != -1; i = (i + h * h++) %
maxSize) {
                int tmp = hashTable[i];
                hashTable[i] = -1;
                insertQuadraticProbing(hashTable, tmp);
            }
    }

        public static boolean searchQuadraticProbing(int[] hashTable, int search)
    {
       int i = hash(search), h = 1;
       while (hashTable[i] != -1)
          {
             if (hashTable[i] == search)
                return true;
             i = (i + h * h++) % maxSize;
          }
       return false;
    }
```

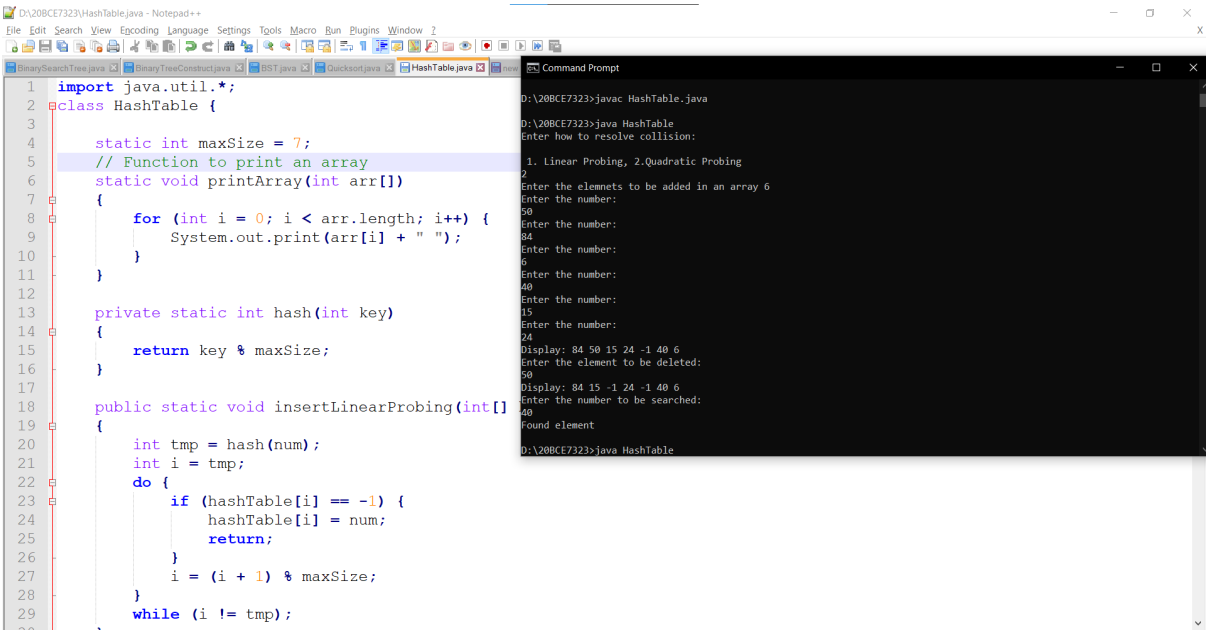
```
// Driver code
public static void main(String args[])
{
	Scanner sc = new Scanner(System.in);
	System.out.println("Enter how to resolve collision: \n" );
	System.out.println(" 1. Linear Probing, 2.Quadratic Probing");
	int choice = sc.nextInt();
	System.out.print("Enter the elemnets to be added in an array ");
	int n = sc.nextInt();
	int hashTable[] = new int[maxSize];
	// Initializing the hash table
	for (int i = 0; i < maxSize; i++) {
		hashTable[i] = -1;
	}
	if(choice == 1) {
		// Linear Probing
		for(int i = 0; i < n; i++) {
			System.out.println("Enter the number: ");
			int num = sc.nextInt();
			insertLinearProbing(hashTable, num);
		}
		System.out.print("Display: ");
		printArray(hashTable);
		System.out.println("\nEnter the element to be deleted: ");
		int del = sc.nextInt();
		deleteLinearProbing(hashTable, del);
```

```
            System.out.print("Display: ");
            printArray(hashTable);
            System.out.println("\nEnter the number to be searched: ");
            int search = sc.nextInt();
            boolean found = searchLinearProbing(hashTable, search);
            if(found) {
                System.out.println("Found element");
            } else {
                System.out.println("Not Found");
            }
        } else if(choice == 2) {
            // Quadratic probing
            for(int i = 0; i < n; i++) {
                System.out.println("Enter the number: ");
                int num = sc.nextInt();
                insertQuadraticProbing(hashTable, num);
            }
            System.out.print("Display: ");
            printArray(hashTable);
            System.out.println("\nEnter the element to be deleted: ");
            int del = sc.nextInt();
            deleteQuadraticProbing(hashTable, del);
            System.out.print("Display: ");
            printArray(hashTable);
            System.out.println("\nEnter the number to be searched: ");
            int search = sc.nextInt();
```

```
                    boolean found = searchQuadraticProbing(hashTable,
search);
                    if(found) {
                        System.out.println("Found element");
                    } else {
                        System.out.println("Not Found");
                    }
                } else {
                    System.out.println("Invalid choice");
                }
            }
}
```

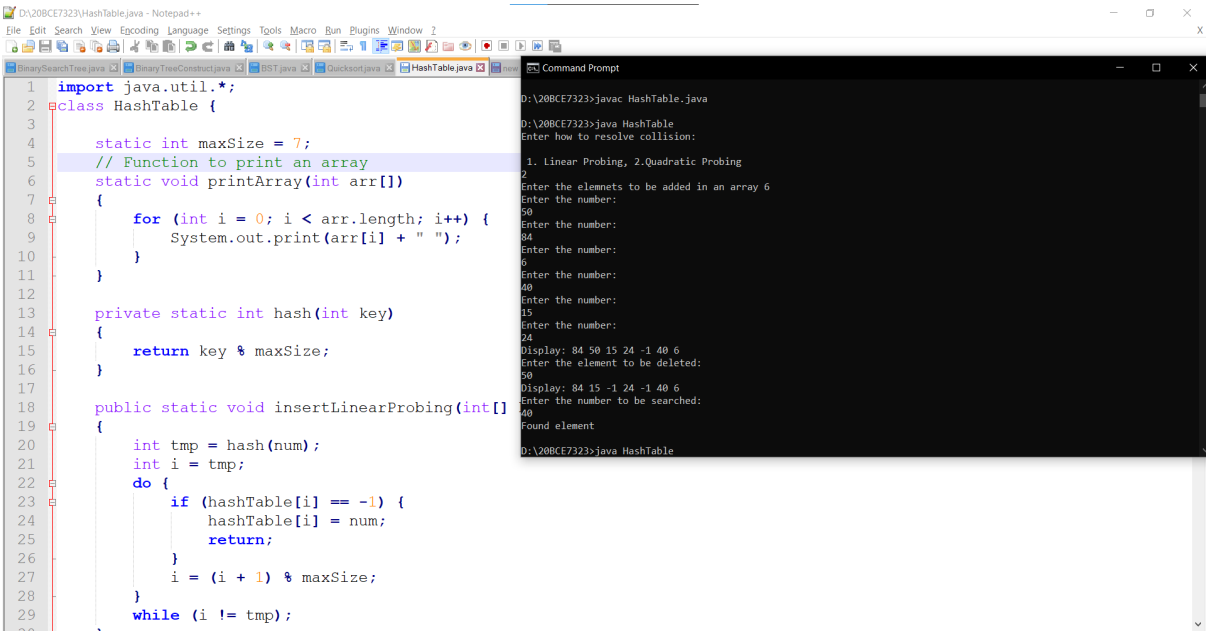

Command Prompt
D:\20BCE7323>javac HashTable.java
D:\20BCE7323>java HashTable
Enter how to resolve collision:
1. Linear Probing, 2.Quadratic Probing
2
Enter the elemnets to be added in an array 6
Enter the number:
50
Enter the number:
84
Enter the number:
6
Enter the number:
40
Enter the number:
15
Enter the number:
24
Display: 84 50 15 24 -1 40 6
Enter the element to be deleted:
50
Display: 84 15 -1 24 -1 40 6
Enter the number to be searched:
40
Found element
D:\20BCE7323>


D:\20BCE7323>java HashTable
Enter how to resolve collision:
1. Linear Probing, 2.Quadratic Probing
1
Enter the elemnets to be added in an array 6
Enter the number:
50
Enter the number:
84
Enter the number:
6
Enter the number:
40
Enter the number:
15
Enter the number:
24
Display: 84 50 15 24 -1 40 6
Enter the element to be deleted:
40
Display: 84 50 15 24 -1 -1 6
Enter the number to be searched:
15
Found element
D:\20BCE7323>



```java
import java.util.*;
class HashTable {

    static int maxSize = 7;
    // Function to print an array
    static void printArray(int arr[])
    {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
    }

    private static int hash(int key)
    {
        return key % maxSize;
    }

    public static void insertLinearProbing(int[]
    {
        int tmp = hash(num);
        int i = tmp;
        do {
            if (hashTable[i] == -1) {
                hashTable[i] = num;
                return;
            }
            i = (i + 1) % maxSize;
        }
        while (i != tmp);
```

```
Command Prompt

D:\20BCE7323>javac HashTable.java

D:\20BCE7323>java HashTable
Enter how to resolve collision:

 1. Linear Probing, 2.Quadratic Probing
2
Enter the elemnets to be added in an array 6
Enter the number:
50
Enter the number:
84
Enter the number:
6
Enter the number:
40
Enter the number:
15
Enter the number:
24
Display: 84 50 15 24 -1 40 6
Enter the element to be deleted:
50
Display: 84 15 -1 24 -1 40 6
Enter the number to be searched:
40
Found element

D:\20BCE7323>java HashTable
```



```java
import java.util.*;
class HashTable {

    static int maxSize = 7;
    // Function to print an array
    static void printArray(int arr[])
    {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
    }

    private static int hash(int key)
    {
        return key % maxSize;
    }

    public static void insertLinearProbing(int[]
    {
        int tmp = hash(num);
        int i = tmp;
        do {
            if (hashTable[i] == -1) {
                hashTable[i] = num;
                return;
            }
            i = (i + 1) % maxSize;
        }
        while (i != tmp);
```

```
Command Prompt

D:\20BCE7323>javac HashTable.java

D:\20BCE7323>java HashTable
Enter how to resolve collision:

 1. Linear Probing, 2.Quadratic Probing
2
Enter the elemnets to be added in an array 6
Enter the number:
50
Enter the number:
84
Enter the number:
6
Enter the number:
40
Enter the number:
15
Enter the number:
24
Display: 84 50 15 24 -1 40 6
Enter the element to be deleted:
50
Display: 84 15 -1 24 -1 40 6
Enter the number to be searched:
40
Found element

D:\20BCE7323>java HashTable
```