Lab 3

Problem Statement: Steepest Hill Climbing

NAME:	Harshita Bhagat	ROLLNO:	31
CLASS:	TY - IT A	BATCH:	2

Code:

```
package AI;
import java.util.Scanner;


public class HillClimbing {
   static int goal[][] = new int[3][3];
   static int arr[][] = new int[4][2];


   static int mov;
   static float ans = 100;


   static float calHeuristic(int goal[][], int current[][]) {
       int dist = 0;
       for (int i = 0; i < 3; i++) {
           for (int j = 0; j < 3; j++) {
               dist += Math.pow(goal[i][j] - current[i][j], 2);
           }
       }
       float hval = (float) Math.sqrt(dist);
```

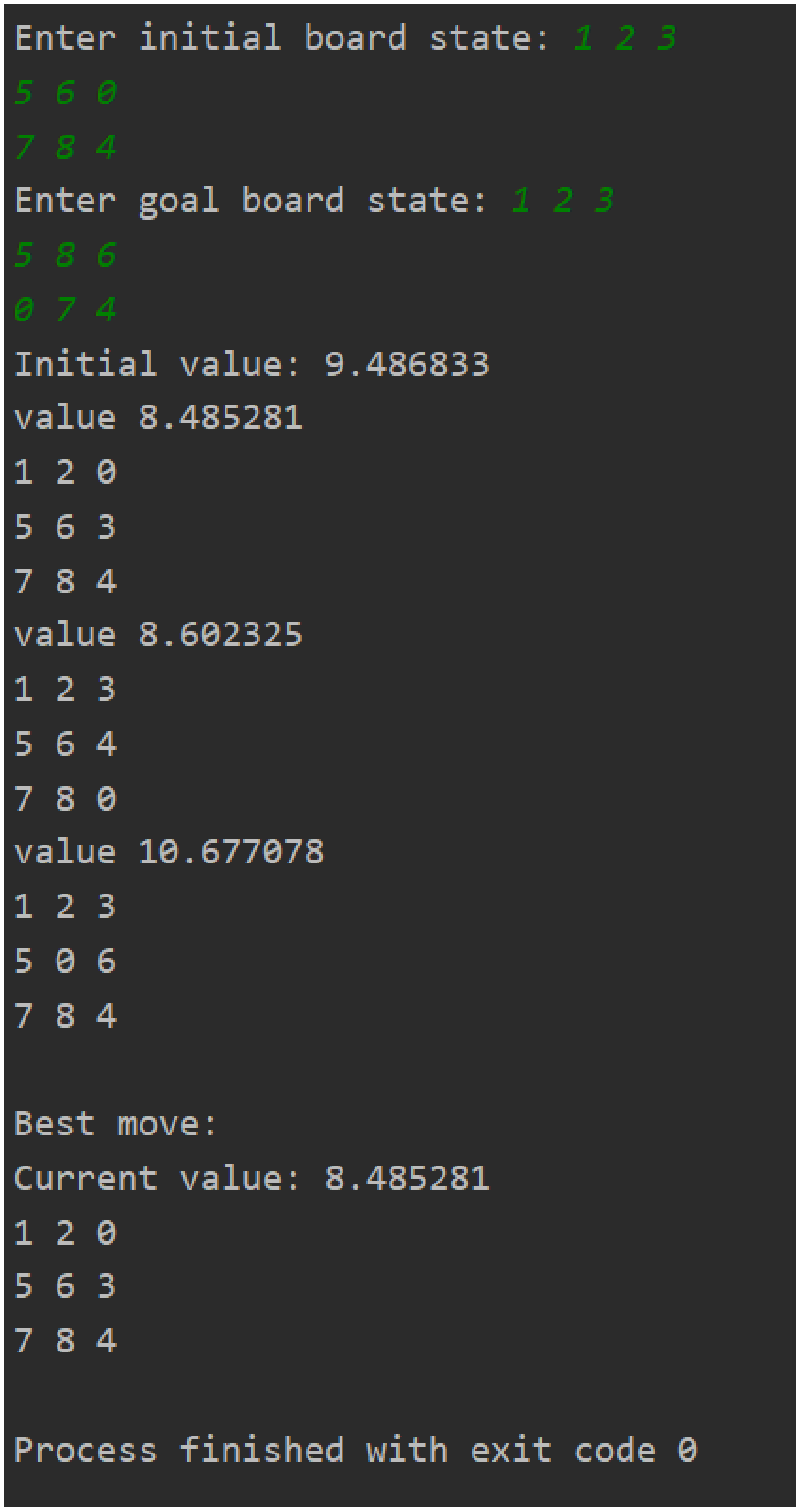
```
        dist = 0;
        return hval;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter initial board state: ");
        int current[][] = new int[3][3];
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                current[i][j] = sc.nextInt();
            }
        }

        System.out.print("Enter goal board state: ");
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                goal[i][j] = sc.nextInt();
            }
        }

        ans = calHeuristic(goal, current);
        System.out.println("Initial value: " + ans);
        if (ans == 0) {
            System.out.println("Initial state is the goal
state.");
        } else {
            int x = -1, y = -1;
            for (int i = 0; i < 3; i++) {
                for (int j = 0; j < 3; j++) {
                    if (current[i][j] == 0) {
                        x = i;
                        y = j;
                    }
                }
            }
            mov = findIndex(x, y, arr);
            float bestHval = ans;
            int[][] SUCC = new int[3][3];
            for (int i = 0; i < mov; i++) {
                int temp[][] = new int[3][3];
                drawMatrix(current, x, y, arr[i][0], arr[i][1],
temp);
                float tans = calHeuristic(goal, temp);
                System.out.println("value " + tans);
                for (int k = 0; k < 3; k++) {
                    for (int j = 0; j < 3; j++) {
                        System.out.print(temp[k][j] + " ");
                    }
                    System.out.println();
```

```java
            }
            if (tans < bestHval) {
                bestHval = tans;
                SUCC = temp;
            }
        }
        if (bestHval >= ans) {
            System.out.println("No operator left");
        }
        current = SUCC;
        ans = bestHval;
        System.out.println("\nBest move: ");
        System.out.println("Current value: " + ans);
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                System.out.print(current[i][j] + " ");
            }
            System.out.println();
        }
        if (ans == 0) {
            System.out.println("Goal state reached.");
        }
    }
}
```

```
   static void drawMatrix(int[][] current, int x, int y, int p,
int q, int[][] temp) {

       for (int i = 0; i < 3; i++) {

           for (int j = 0; j < 3; j++) {

               if (i == x && j == y) {

                   temp[i][j] = current[p][q];

               } else if (i == p && j == q) {

                   temp[i][j] = 0;

               } else {

                   temp[i][j] = current[i][j];

               }

           }

       }

   }


   static int findIndex(int i, int j, int arr[][]) {

       int k = 0, cnt = 0;

       if ((3 > (i - 1) && i - 1 >= 0) && (3 > j && j >= 0)) {

           arr[k][0] = i - 1;

           arr[k][1] = j;

           k++;

           cnt++;

       }

       if (3 > i + 1 && i + 1 >= 0 && 3 > j && j >= 0) {

           arr[k][0] = i + 1;
```

```
            arr[k][1] = j;

            k++;

            cnt++;

        }

        if ((3 > i && i >= 0) && (3 > (j + 1) && j + 1 >= 0)) {

            arr[k][0] = i;

            arr[k][1] = j + 1;

            k++;

            cnt++;

        }

        if ((3 > i && i >= 0) && (3 > j - 1 && j - 1 >= 0)) {

            arr[k][0] = i;

            arr[k][1] = j - 1;

            k++;

            cnt++;

        }

        mov = cnt;

        return mov;

    }

}
```

Output:



```
Enter initial board state: 1 2 3
5 6 0
7 8 4
Enter goal board state: 1 2 3
5 8 6
0 7 4
Initial value: 9.486833
value 8.485281
1 2 0
5 6 3
7 8 4
value 8.602325
1 2 3
5 6 4
7 8 0
value 10.677078
1 2 3
5 0 6
7 8 4

Best move:
Current value: 8.485281
1 2 0
5 6 3
7 8 4

Process finished with exit code 0
```