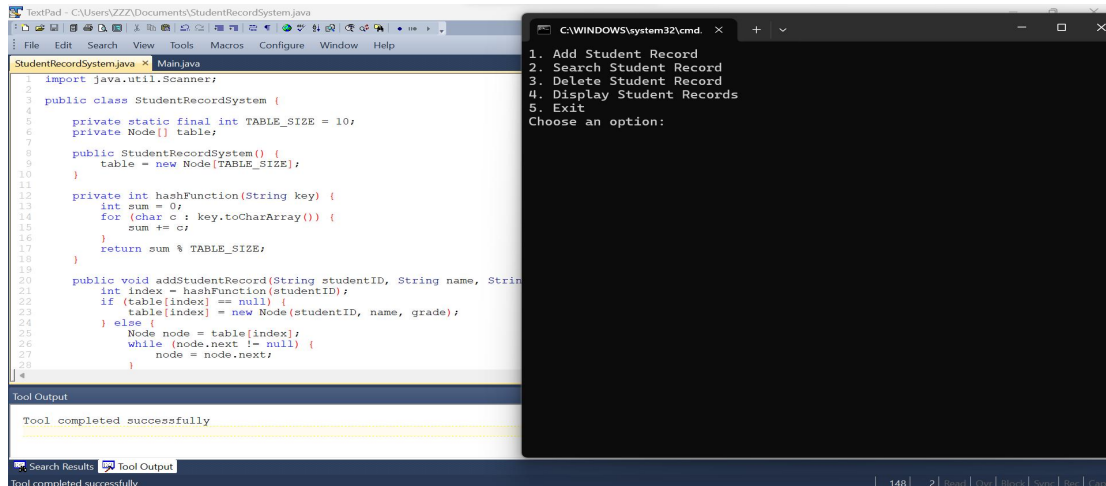


Garen Xade Ponsica  
BSIT-2B

## Student Record System



The screenshot displays a Java IDE (TestPad) on the left and a Windows Command Prompt on the right. The IDE shows the source code for a 'StudentRecordSystem.java' file. The code defines a hash table-based system for managing student records. It includes a hash function, a method to add records, and a menu-driven interface. The command prompt window shows the program's output, which is a menu with five options: 1. Add Student Record, 2. Search Student Record, 3. Delete Student Record, 4. Display Student Records, and 5. Exit. The prompt asks the user to 'Choose an option:'.

```
StudentRecordSystem.java Main.java
1 import java.util.Scanner;
2
3 public class StudentRecordSystem {
4
5     private static final int TABLE_SIZE = 10;
6     private Node[] table;
7
8     public StudentRecordSystem() {
9         table = new Node[TABLE_SIZE];
10    }
11
12    private int hashFunction(String key) {
13        int sum = 0;
14        for (char c : key.toCharArray()) {
15            sum += c;
16        }
17        return sum % TABLE_SIZE;
18    }
19
20    public void addStudentRecord(String studentID, String name, String grade) {
21        int index = hashFunction(studentID);
22        if (table[index] == null) {
23            table[index] = new Node(studentID, name, grade);
24        } else {
25            Node node = table[index];
26            while (node.next != null) {
27                node = node.next;
28            }
29            node.next = new Node(studentID, name, grade);
30        }
31    }
32
33    public void searchStudentRecord(String studentID) {
34        int index = hashFunction(studentID);
35        Node node = table[index];
36        while (node != null) {
37            if (node.studentID.equals(studentID)) {
38                System.out.println("Student Record Found: " + node.name + ", " + node.grade);
39            }
40            node = node.next;
41        }
42    }
43
44    public void deleteStudentRecord(String studentID) {
45        int index = hashFunction(studentID);
46        Node node = table[index];
47        while (node != null) {
48            if (node.studentID.equals(studentID)) {
49                node = node.next;
50            }
51            node = node.next;
52        }
53    }
54
55    public void displayStudentRecords() {
56        for (int i = 0; i < table.length; i++) {
57            Node node = table[i];
58            while (node != null) {
59                System.out.println("Student Record: " + node.studentID + ", " + node.name + ", " + node.grade);
60                node = node.next;
61            }
62        }
63    }
64
65    public static void main(String[] args) {
66        StudentRecordSystem system = new StudentRecordSystem();
67        Scanner scanner = new Scanner(System.in);
68        while (true) {
69            System.out.println("1. Add Student Record\n2. Search Student Record\n3. Delete Student Record\n4. Display Student Records\n5. Exit\nChoose an option:");
70            int option = scanner.nextInt();
71            switch (option) {
72                case 1:
73                    System.out.println("Enter studentID, name, and grade separated by spaces:");
74                    String[] input = scanner.nextLine().split(" ");
75                    system.addStudentRecord(input[0], input[1], input[2]);
76                    break;
77                case 2:
78                    System.out.println("Enter studentID to search:");
79                    system.searchStudentRecord(scanner.nextLine());
80                    break;
81                case 3:
82                    System.out.println("Enter studentID to delete:");
83                    system.deleteStudentRecord(scanner.nextLine());
84                    break;
85                case 4:
86                    system.displayStudentRecords();
87                    break;
88                case 5:
89                    break;
90            }
91        }
92    }
93}
```

Tool Output

Tool completed successfully

I made this application to demonstrate a practical use of hash tables in a real-world scenario. Student record systems are a common use case for hash tables, as they require fast lookup and insertion of student records.

### How This Application Contributes to Me or the Community

This application contributes to me by helping me develop my skills in designing and implementing hash table-based data structures. It also contributes to the community by providing a simple and easy-to-understand example of how hash tables can be used in a student record system.