```java
/*
 * Countries of the World App 1.5
 * CountryIndex.java "CountryIndex "
 * Waleed Gudah
 *
 */
import java.io.*;

public class CountryIndex {

    private TheLog theLog;
    private BufferedReader input;
    private int MAX_N_HOME_LOC = 20;
    private int MAX_OVRFLOW = 15;
    private IndexNode[] indexNodes = new IndexNode[MAX_N_HOME_LOC];
    private IndexNode[] overflow = new IndexNode[MAX_OVRFLOW];
    private int nColl = 0;
    private int nHome = 0;
    private int counter = 0;

    public CountryIndex(TheLog theLog) {

        this.theLog = theLog;

        openFile();

        restoreBackup();
    }

    // ***************************************************************//
    private void insertCodeInIndex(String Code) {

        theLog.toLog("   SORRY, insertCodeInIndex not yet working");

    }

    // ***************************************************************//
    private int hashFunction(String countryCode) {

        int result = 1;

        char[] characters = countryCode.toCharArray();

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

            char currentChar = characters[i];

            int j = (int) currentChar;

            result = result * j;

        }

        return (result % (MAX_N_HOME_LOC));

    }

    // ***************************************************************//
    public int locationfindInIndex(String code) {

        int home = hashFunction(code);

        if (indexNodes[home] != null
                && code.equalsIgnoreCase(indexNodes[home].getKeyValue())) {
```

```
        return home;

    } else {

    }

    return home;
}

// ************************************************************//
public void snapShot() {

    theLog.transProcess(" "
            + String.format(
                    "CODE INDEX> MAX_N_HOME_LOC: %d, nHome: %d, nColl: %d",
                    MAX_N_HOME_LOC, 17, nColl));

    theLog.transProcess("  [SUB] CODE | DRP |LINK |");

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

        if (indexNodes[i] != null) {

            theLog.toLog(String.format("[%03d] %4s | %-3d | %-3d |" + "\n",
                    i, indexNodes[i].getKeyValue(), indexNodes[i].getDRP(),
                    indexNodes[i].getHP()));

        } else {

            theLog.toLog(String.format("[%03d]  EMPTY" + "\n", i));

        }
    }

    for (int r = 0; r < overflow.length; r++) {

        theLog.toLog(String.format("[%03d] %4s | %-3d | %-3d |" + "\n",
                r + 20, overflow[r].getKeyValue(), overflow[r].getDRP(),
                overflow[r].getHP()));

    }

}

// ************************************************************//
public void restoreBackup() {

    try {

        for (int i = 0; i < (MAX_N_HOME_LOC + MAX_OVRFLOW); i++) {

            String record;

            record = input.readLine();

            if (record.equals(",,-1")) {

                indexNodes[i] = null;

            } else if (i > MAX_N_HOME_LOC - 1) {

                nColl++;
```

```
                String[] oneRecord = record.split(",");

                overflow[i - MAX_N_HOME_LOC] = (new IndexNode(oneRecord[0],
                        Integer.parseInt(oneRecord[1]),
                        Integer.parseInt(oneRecord[2])));
            } else {

                String[] oneRecord = record.split(",");

                indexNodes[i] = (new IndexNode(oneRecord[0],
                        Integer.parseInt(oneRecord[1]),
                        Integer.parseInt(oneRecord[2])));

            }
        }

    } catch (IOException e) {

        e.printStackTrace();

    }

}

// ************************************************************//
public void openFile() {

    try {

        input = new BufferedReader(new FileReader("CountryIndex.csv"));

        theLog.statusFile("IndexBackup FILE opened");

    } catch (IOException e) {

        System.out.println("Error, File may have been prematurely closed");

    }

}

// ************************************************************//
public void closeFile() {

    try {

        input.close();

        theLog.statusFile("IndexBackup FILE closed");

    } catch (IOException e) {

        System.out.println("Error, File may have been prematurely closed");

    }

}

// ************************************************************//
public int getnColl() {

    return nColl;

}
```

```
// ************************************************************//
public int getnHome() {

    return nHome;

}

// ************************************************************//
public int getCounter() {

    return counter;

}

// ************************************************************//
public int getMAX_N_HOME_LOC() {
    return MAX_N_HOME_LOC;
}

// ************************************************************//
// ************************************************************//
public class IndexNode {

    private String keyValue;
    private int DRP;
    private int HP;

    public IndexNode(String keyValue, int DRP, int HP) {

        this.keyValue = keyValue;
        this.DRP = DRP;
        this.HP = HP;

    }

    public IndexNode(int HP) {

        this.HP = HP;

    }

    // ************************************************************//
    public String getKeyValue() {

        return keyValue;

    }

    // ************************************************************//
    public int getDRP() {

        return DRP;

    }

    // ************************************************************//
    public int getHP() {

        return HP;

    }

    // ************************************************************//
```

```
        public void setKeyValue(String keyValue) {
            this.keyValue = keyValue;
        }

        // ************************************************************//
        public void setDRP(int dRP) {
            DRP = dRP;
        }

        // ************************************************************//
        public void setHP(int hP) {
            HP = hP;
        }

    }

}
```