Visvesvaraya Technological University
Belagavi-590018, Karnataka

A Mini Project Report on

"INDEXING BASED ON PRIMARY & SECONDARY KEY"

Submitted in partial fulfilment of the requirement for the
FILE STRUCTURES LABORATORY WITH MINI PROJECT [17ISL68]

Bachelor of Engineering
in
Information Science and Engineering

Submitted by
Moulya.A[1JT17IS023]

Department of Information Science and Engineering
Jyothy Institute of Technology
Tataguni, Bengaluru-560082
2019-2020

CERTIFICATE

Certified that the mini project work entitled **“INDEXING BASED ON PRIMARY & SECONDARY KEY** ”carried out by **Moulya.A[1JT17IS023]** bonafide student of Jyothy Institute of Technology, in partial fulfilment for the award of **Bachelor of Engineering** in **Information Science & Engineering** department of the **Visvesvaraya Technological University, Belagavi** during the year **2019-2020**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of Mini Project work prescribed for the said Degree.

Vadiraja.A.
Guide, Assist Professor
Dept. Of ISE

Dr.Harshavardhan Tiwari
Associate Professor and HoD
Dept. Of ISE

External Viva Examiner

Signature with Date

1.
2.

ACKNOWLEDGEMENT

Firstly, I’m very grateful to this esteemed institution **Jyothy Institute of Technology** for providing me an opportunity to complete our project.

I express our sincere thanks to my Principal, **Dr. Gopalakrishna K for** providing me with adequate facilities to undertake this project.

I would like to thank associate Prof **Dr. Harshwardhan Tiwari, Professor and Head** of Information Science and Engineering Department for providing for his valuable support.

I would like to thank my guide **Mr. Vadiraja.A, Asst. Prof.** for his keen interest and guidance in preparing this work.

Finally, I would thank all my friends who have helped me directly or indirectly in this project.

ABSTRACT

In this mini project I have created an application which is easy to access and user friendly. The project title is “Indexing based on Primary & Secondary key”, which is used in the application and for distinguishing the type of file structure to be used. As we all know Index is a table containing a list of topics(keys) & numbers of pages where the topics can be found(reference fields). In this mini project I have implemented indexing technique based on primary and secondary key. This particular code was created using the programming language JAVA with an objective to formulate user-friendly access to build indexes based on the selected keys. Indexing simplify the work and it takes the same time to search the first as well as the last record in the file. Thus, this code is more efficient and broader as it can build indexes based both on primary and secondary keys.

CHAPTER 1
INTRODUCTION

1. INTRODUCTION

1.1 Introduction to File Structures

A file is a collection of data stored on mass storage (e.g., disk or tape).

But there is one important distinction that must be made at the outset when discussing file structures. And that is the difference between the logical and physical organization of the data.

On the whole a file structure will specify the logical structure of the data, that is the relationships that will exist between data items independently of the way in which these relationships may actually be realized within any computer. It is this logical aspect that we will concentrate on. The physical organization is much more concerned with optimizing the use of the storage medium when a particular logical structure is stored on, or in it. Typically for every unit of physical store there will be a number of units of the logical structure (probably records) to be stored in it.

For example, if we were to store a tree structure on a magnetic disk, the physical organization would be concerned with the best way of packing the nodes of the tree on the disk given the access characteristics of the disk.

Like all subjects in computer science the terminology of file structures has evolved higgledy-piggledy without much concern for consistency, ambiguity, or whether it was possible to make the kind of distinctions that were important.

It was only much later that the need for a well-defined, unambiguous language to describe file structures became apparent. In particular, there arose a need to communicate ideas about file structures without getting bogged down by hardware considerations.

1.2 Indexing

An index in a day-to-day life is more generally found in books either at the beginning or at the end, which contains a table consisting of a list of topics(keys) and the numbers of the pages where the topic can be found(reference fields).All indexes are based on the same concept i.e., keys and reference fields. Simple indexes are represented using simple arrays of structures containing of keys and reference fields.

In general, indexing lets us impose order on a file without rearranging the file.

Simple indexing can be useful when the entire index can be held in the memory. Changes like addition and deletions require both the index and the data file to be changed.

If index is held in memory, record addition, deletion and retrieval can be done much more quickly with an indexed, entry sequenced file rather than a sorted file. Index files also allow us to maintain different views of the records in a data file.

Indexing can be concluded as a way of structuring a file in an alternative to sorting because records can be found by key.

Keeping all these points in mind an index can be defined as a tool for finding records in a file. It consists of a key field on which the index is searched and a reference field that tells where to find the data file record associated with a particular key.

CHAPTER 2
REQUIREMENT ANALYSIS & DESIGN

2.REQUIREMENT ANALYSIS AND DESIGN

2.1 Domain understanding

The main object of the project is to index all the key words of the document in a separate file and perform operations like finding & deleting. The outcome of this project is to ease the user for finding and replacing words with a friendly, understandable GUI.

2.2 Classification of Requirements

2.2.1 **User Requirements**

Operating System: Windows/Linux

Java Installed

2.2.2 **Software and Hardware Requirements**

Java run time Environment

Any Linux machine with 4GB RAM

2.3 System Analysis

When the program is executed, it primarily asks the user to operate any of the 8 operations presented, which includes find, add a new record, or even deleting an existing record and to build indexes using either primary or secondary indexes. The operations are performed based on the option selected.

CHAPTER 3
IMPLEMENTATION

3.1 ALGORITHMS

3.1.1 SEARCH:

Binary Search Algorithm:

```
BINARY_SEARCH(A, lower_bound, upper_bound, VAL)
Step 1: [INITIALIZE] SET BEG = lower_bound
        END = upper_bound, POS = - 1
Step 2: Repeat Steps 3 and 4 while BEG <=END
    Step 3: SET MID = (BEG + END)/2
    Step 4: IF A[MID] = VAL
                SET POS = MID
                PRINT POS
                Go to Step 6
            ELSE IF A[MID] > VAL
                SET END = MID - 1
            ELSE
                SET BEG = MID + 1
            [END OF IF]
    [END OF LOOP]
Step 5: IF POS = -1
            PRINT "VALUE IS NOT PRESENT IN THE ARRAY"
        [END OF IF]
Step 6: EXIT
```

3.1.2 Primary Indexing:

Here we consider the file that contains the records required, and use the getfilepointer() which is used to return the current offset in the file, in bytes, at which the next read or write occurs. We use a while condition with the file to read the file line by line with the help of readLine()[it returns the next line of text from the file, or null if end of file is encountered before even one byte is read]. Inside the while condition we come across a if condition which is used to detect the “*” [astrick special character]. If the filepointer encounters the “*” symbol in the file it has to continue with the execution. Then each record entry is split, and a new file is used to store the indexes created. Here I have used the following piece of code to create the index based on primary key: `indexfile.writeBytes(columns[0]+","+pos+"\n");` where:

1.The indexfile refers to the new file created to store the indexes

2.The writeBytes is used to write the string of bytes to a specified file

3.columns[0] specifies that the 1st column should be considered

4.pos gives the position or the field of the key.

Once these specified operations are done we have to close both the files using the close().

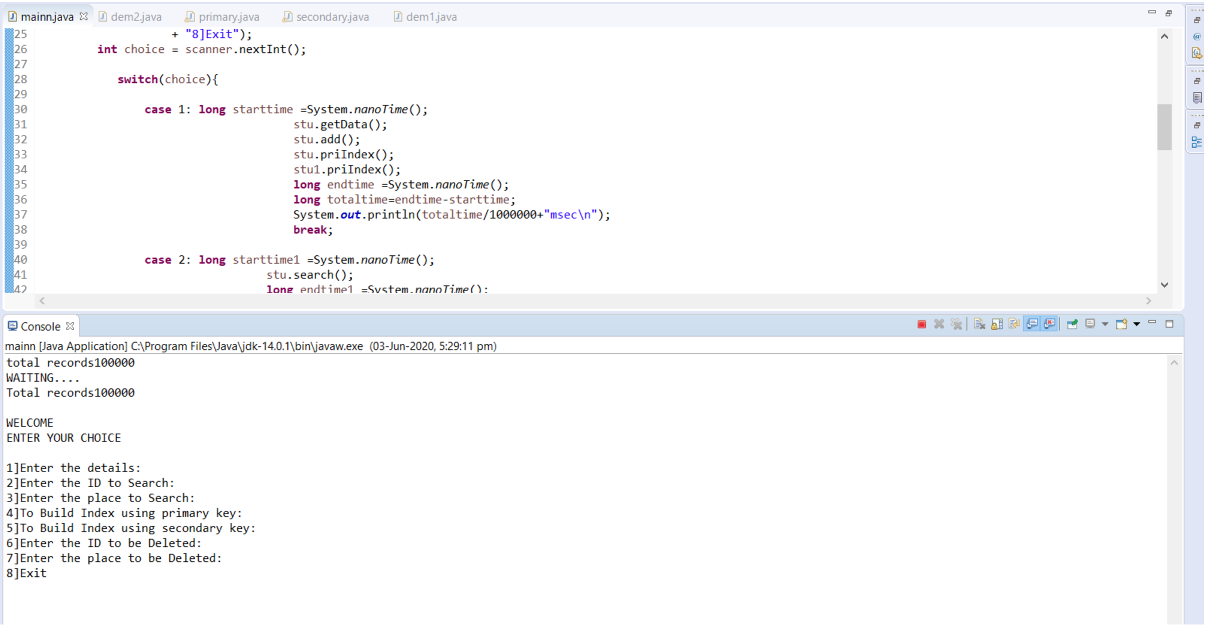
3.1.3 Secondary Indexing:

Creation of index using a secondary key is very much similar to that of the index built on primary key. They differ in the keys used and an another separate file to build the index for the secondary index, i.e., `indexfile.writeBytes(columns[1]+","+pos+"\n"); instead of columns[0] I have used columns[1] which is the secondary key place in my case on consideration.`

CHAPTER 4
RESULTS AND SNAPSHOT

4.1 SNAPSHOTS:

(a)Once the program is executed, following options appear, and the user must select one among them, according to the selected option the operation will be performed.



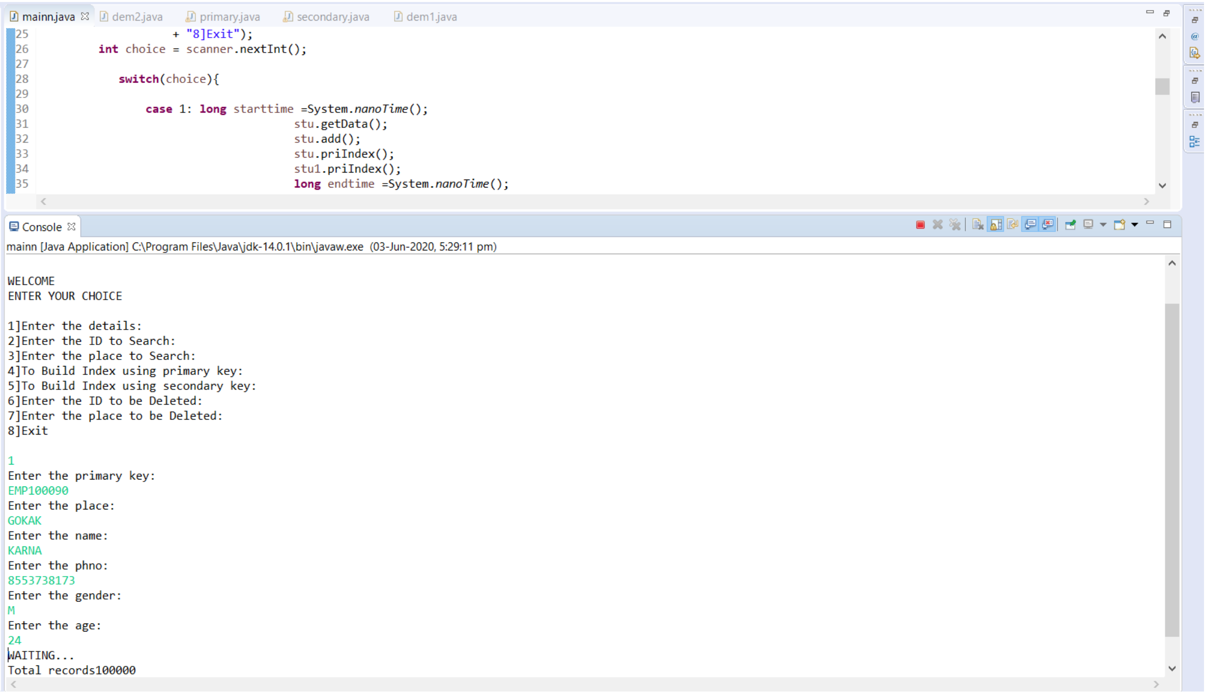
```
                    + "8]Exit");
        int choice = scanner.nextInt();

           switch(choice){

               case 1: long starttime =System.nanoTime();
                                      stu.getData();
                                      stu.add();
                                      stu.priIndex();
                                      stu1.priIndex();
                                      long endtime =System.nanoTime();
                                      long totaltime=endtime-starttime;
                                      System.out.println(totaltime/1000000+"msec\n");
                                      break;

               case 2: long starttime1 =System.nanoTime();
                                  stu.search();
                                  long endtime1 =System.nanoTime();
```

mainn [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (03-Jun-2020, 5:29:11 pm)

```
total records100000
WAITING....
Total records100000

WELCOME
ENTER YOUR CHOICE

1]Enter the details:
2]Enter the ID to Search:
3]Enter the place to Search:
4]To Build Index using primary key:
5]To Build Index using secondary key:
6]Enter the ID to be Deleted:
7]Enter the place to be Deleted:
8]Exit
```

Figure 4.1(a)

(b)When option 1 is selected:



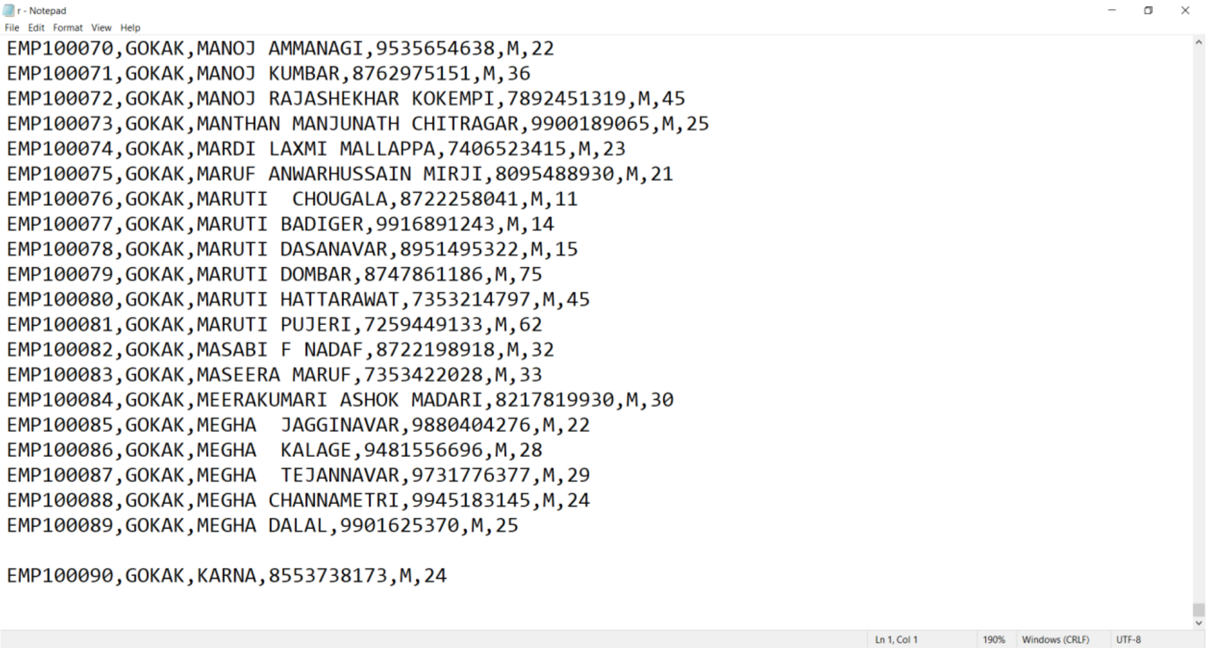
```
                    + "8]Exit");
        int choice = scanner.nextInt();

           switch(choice){

               case 1: long starttime =System.nanoTime();
                                      stu.getData();
                                      stu.add();
                                      stu.priIndex();
                                      stu1.priIndex();
                                      long endtime =System.nanoTime();
```

mainn [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (03-Jun-2020, 5:29:11 pm)

```

WELCOME
ENTER YOUR CHOICE

1]Enter the details:
2]Enter the ID to Search:
3]Enter the place to Search:
4]To Build Index using primary key:
5]To Build Index using secondary key:
6]Enter the ID to be Deleted:
7]Enter the place to be Deleted:
8]Exit

1
Enter the primary key:
EMP100090
Enter the place:
GOKAK
Enter the name:
KARNA
Enter the phno:
8553738173
Enter the gender:
M
Enter the age:
24
WAITING...
Total records100000
```

Figure 4.1(b)

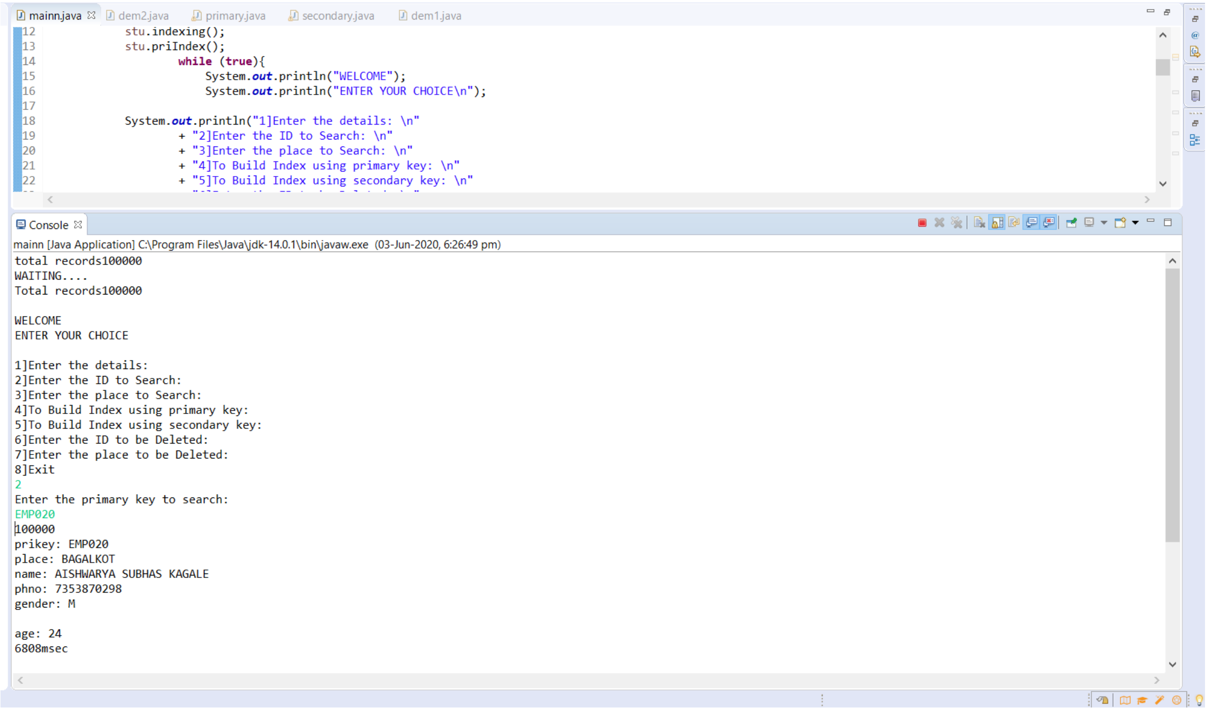
(c)After the entries are made, the new record is added into the source file:



```
EMP100070,GOKAK,MANOJ AMMANAGI,9535654638,M,22
EMP100071,GOKAK,MANOJ KUMBAR,8762975151,M,36
EMP100072,GOKAK,MANOJ RAJASHEKHAR KOKEMPI,7892451319,M,45
EMP100073,GOKAK,MANTHAN MANJUNATH CHITRAGAR,9900189065,M,25
EMP100074,GOKAK,MARDI LAXMI MALLAPPA,7406523415,M,23
EMP100075,GOKAK,MARUF ANWARHUSSAIN MIRJI,8095488930,M,21
EMP100076,GOKAK,MARUTI  CHOUGALA,8722258041,M,11
EMP100077,GOKAK,MARUTI BADIGER,9916891243,M,14
EMP100078,GOKAK,MARUTI DASANAVAR,8951495322,M,15
EMP100079,GOKAK,MARUTI DOMBAR,8747861186,M,75
EMP100080,GOKAK,MARUTI HATTARAWAT,7353214797,M,45
EMP100081,GOKAK,MARUTI PUJERI,7259449133,M,62
EMP100082,GOKAK,MASABI F NADAF,8722198918,M,32
EMP100083,GOKAK,MASEERA MARUF,7353422028,M,33
EMP100084,GOKAK,MEERAKUMARI ASHOK MADARI,8217819930,M,30
EMP100085,GOKAK,MEGHA  JAGGINAVAR,9880404276,M,22
EMP100086,GOKAK,MEGHA  KALAGE,9481556696,M,28
EMP100087,GOKAK,MEGHA  TEJANNAVAR,9731776377,M,29
EMP100088,GOKAK,MEGHA CHANNAMETRI,9945183145,M,24
EMP100089,GOKAK,MEGHA DALAL,9901625370,M,25

EMP100090,GOKAK,KARNA,8553738173,M,24
```

Figure 4.1(c)

(d)When option 2 is selected:



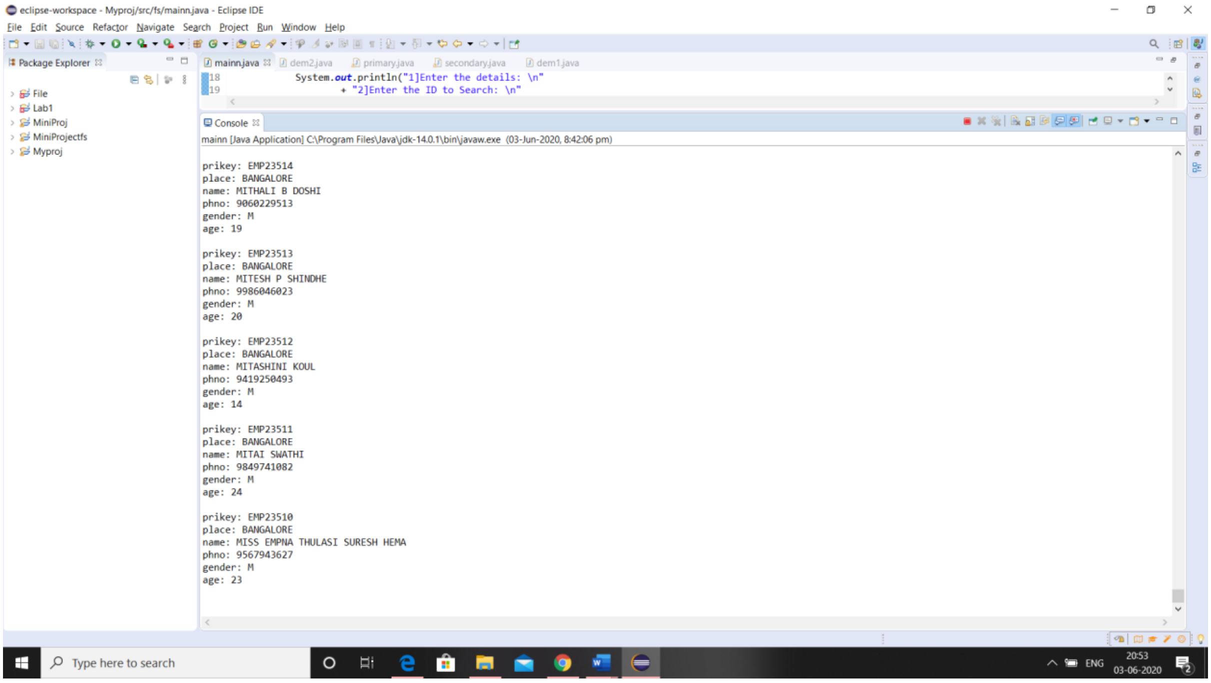
```
            stu.indexing();
            stu.priIndex();
                    while (true){
                        System.out.println("WELCOME");
                        System.out.println("ENTER YOUR CHOICE\n");

            System.out.println("1]Enter the details: \n"
                    + "2]Enter the ID to Search: \n"
                    + "3]Enter the place to Search: \n"
                    + "4]To Build Index using primary key: \n"
                    + "5]To Build Index using secondary key: \n"
```

mainn [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (03-Jun-2020, 6:26:49 pm)

```
total records100000
WAITING....
Total records100000

WELCOME
ENTER YOUR CHOICE

1]Enter the details:
2]Enter the ID to Search:
3]Enter the place to Search:
4]To Build Index using primary key:
5]To Build Index using secondary key:
6]Enter the ID to be Deleted:
7]Enter the place to be Deleted:
8]Exit
2
Enter the primary key to search:
EMP020
100000
prikey: EMP020
place: BAGALKOT
name: AISHWARYA SUBHAS KAGALE
phno: 7353870298
gender: M

age: 24
6808msec
```

Figure 4.1(d)

(e)When option 3 is selected:



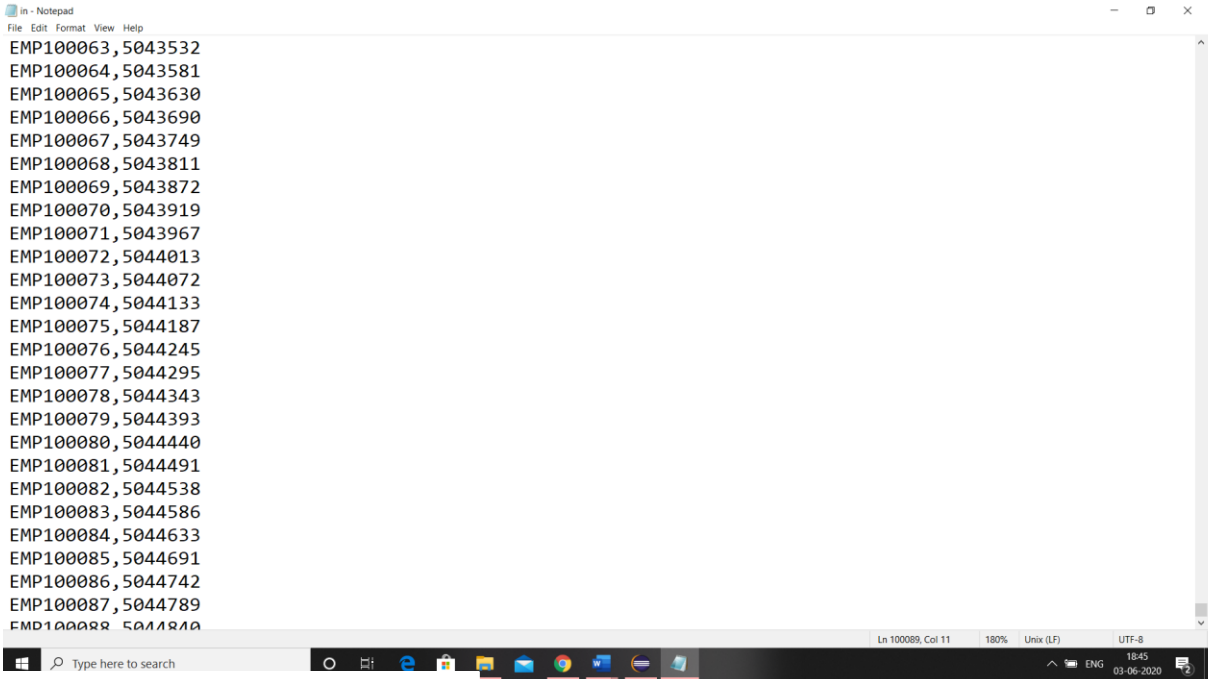
eclipse-workspace - Myproj/src/fs/mainn.java - Eclipse IDE

```
            System.out.println("1]Enter the details: \n"
                    + "2]Enter the ID to Search: \n"
```

```
mainn [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (03-Jun-2020, 8:42:06 pm)

prikey: EMP23514
place: BANGALORE
name: MITHALI B DOSHI
phno: 9060229513
gender: M
age: 19

prikey: EMP23513
place: BANGALORE
name: MITESH P SHINDHE
phno: 9986046023
gender: M
age: 20

prikey: EMP23512
place: BANGALORE
name: MITASHINI KOUL
phno: 9419250493
gender: M
age: 14

prikey: EMP23511
place: BANGALORE
name: MITAI SWATHI
phno: 9849741082
gender: M
age: 24

prikey: EMP23510
place: BANGALORE
name: MISS EMPNA THULASI SURESH HEMA
phno: 9567943627
gender: M
age: 23
```

Figure 4.1(e)

(f)When option 4 is selected:



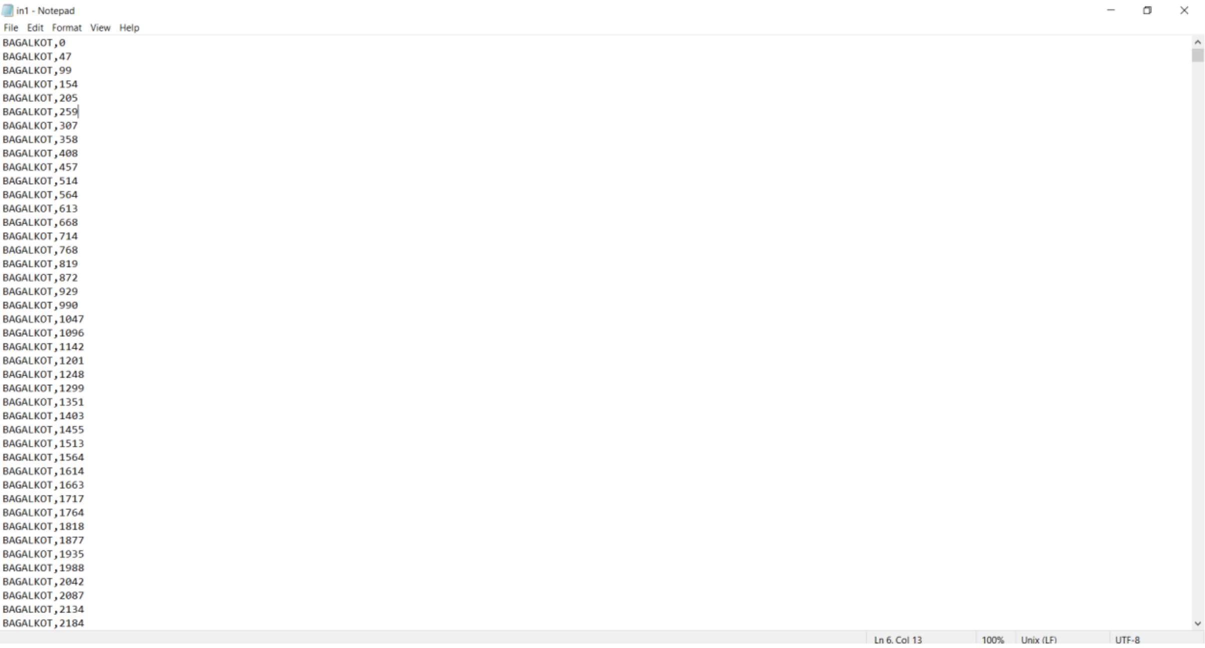
in - Notepad

```
EMP100063,5043532
EMP100064,5043581
EMP100065,5043630
EMP100066,5043690
EMP100067,5043749
EMP100068,5043811
EMP100069,5043872
EMP100070,5043919
EMP100071,5043967
EMP100072,5044013
EMP100073,5044072
EMP100074,5044133
EMP100075,5044187
EMP100076,5044245
EMP100077,5044295
EMP100078,5044343
EMP100079,5044393
EMP100080,5044440
EMP100081,5044491
EMP100082,5044538
EMP100083,5044586
EMP100084,5044633
EMP100085,5044691
EMP100086,5044742
EMP100087,5044789
EMP100088,5044840
```

Ln 100089, Col 11   180%   Unix (LF)   UTF-8

Figure 4.1(f)

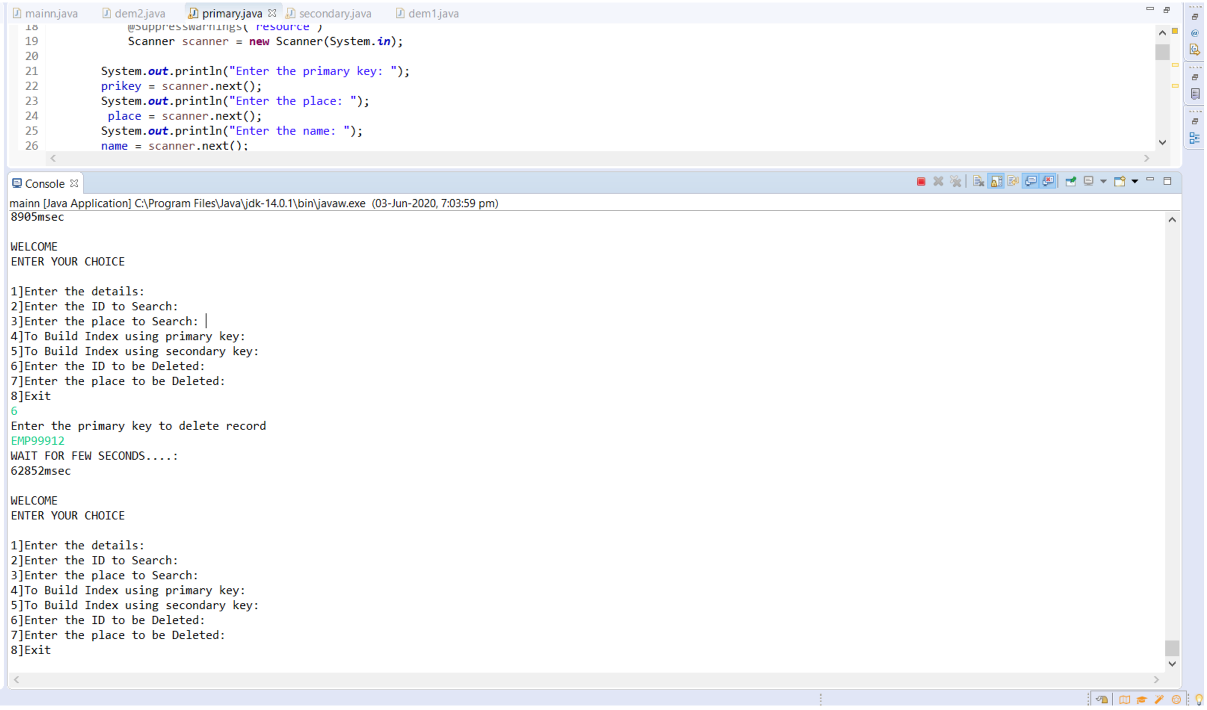
(g)When option 5 is selected:



```
BAGALKOT,0
BAGALKOT,47
BAGALKOT,99
BAGALKOT,154
BAGALKOT,205
BAGALKOT,259
BAGALKOT,307
BAGALKOT,358
BAGALKOT,408
BAGALKOT,457
BAGALKOT,514
BAGALKOT,564
BAGALKOT,613
BAGALKOT,668
BAGALKOT,714
BAGALKOT,768
BAGALKOT,819
BAGALKOT,872
BAGALKOT,929
BAGALKOT,990
BAGALKOT,1047
BAGALKOT,1096
BAGALKOT,1142
BAGALKOT,1201
BAGALKOT,1248
BAGALKOT,1299
BAGALKOT,1351
BAGALKOT,1403
BAGALKOT,1455
BAGALKOT,1513
BAGALKOT,1564
BAGALKOT,1614
BAGALKOT,1663
BAGALKOT,1717
BAGALKOT,1764
BAGALKOT,1818
BAGALKOT,1877
BAGALKOT,1935
BAGALKOT,1988
BAGALKOT,2042
BAGALKOT,2087
BAGALKOT,2134
BAGALKOT,2184
```

Figure 4.1(g)

(h)When option 6 is selected:



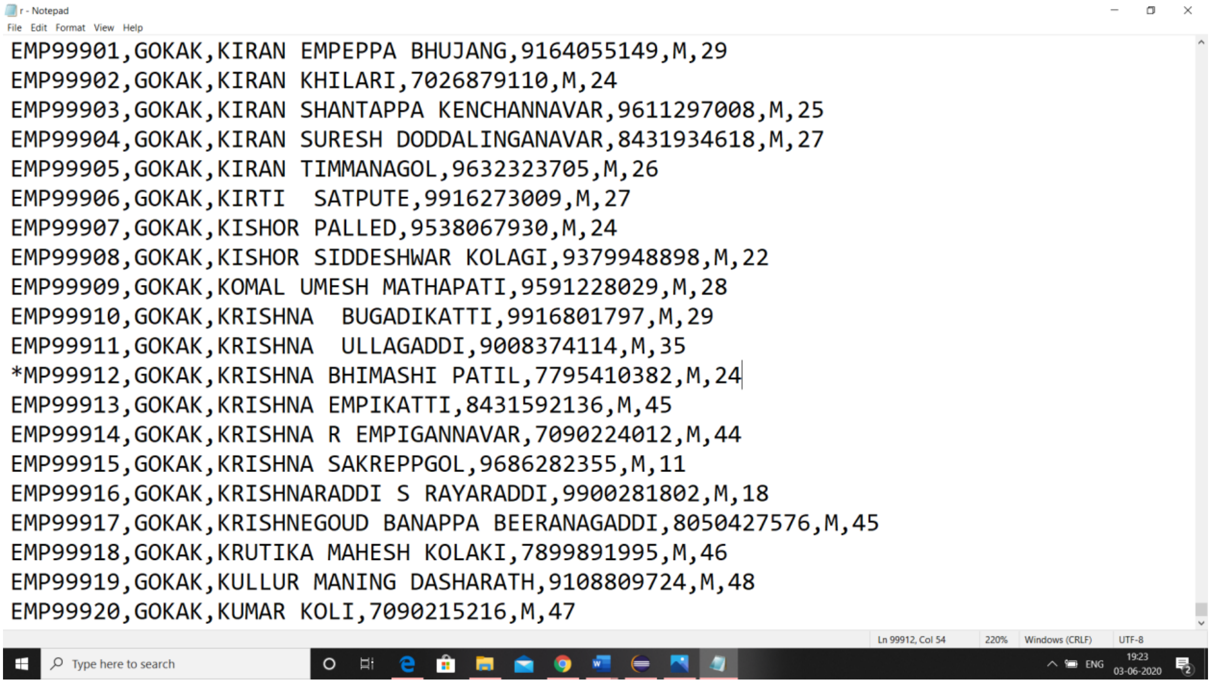
```java
            @SuppressWarnings("resource")
            Scanner scanner = new Scanner(System.in);

        System.out.println("Enter the primary key: ");
        prikey = scanner.next();
        System.out.println("Enter the place: ");
         place = scanner.next();
        System.out.println("Enter the name: ");
        name = scanner.next();
```

mainn [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (03-Jun-2020, 7:03:59 pm)

```
8905msec

WELCOME
ENTER YOUR CHOICE

1]Enter the details:
2]Enter the ID to Search:
3]Enter the place to Search:
4]To Build Index using primary key:
5]To Build Index using secondary key:
6]Enter the ID to be Deleted:
7]Enter the place to be Deleted:
8]Exit
6
Enter the primary key to delete record
EMP99912
WAIT FOR FEW SECONDS....:
62852msec

WELCOME
ENTER YOUR CHOICE

1]Enter the details:
2]Enter the ID to Search:
3]Enter the place to Search:
4]To Build Index using primary key:
5]To Build Index using secondary key:
6]Enter the ID to be Deleted:
7]Enter the place to be Deleted:
8]Exit
```

Figure 4.1(h)

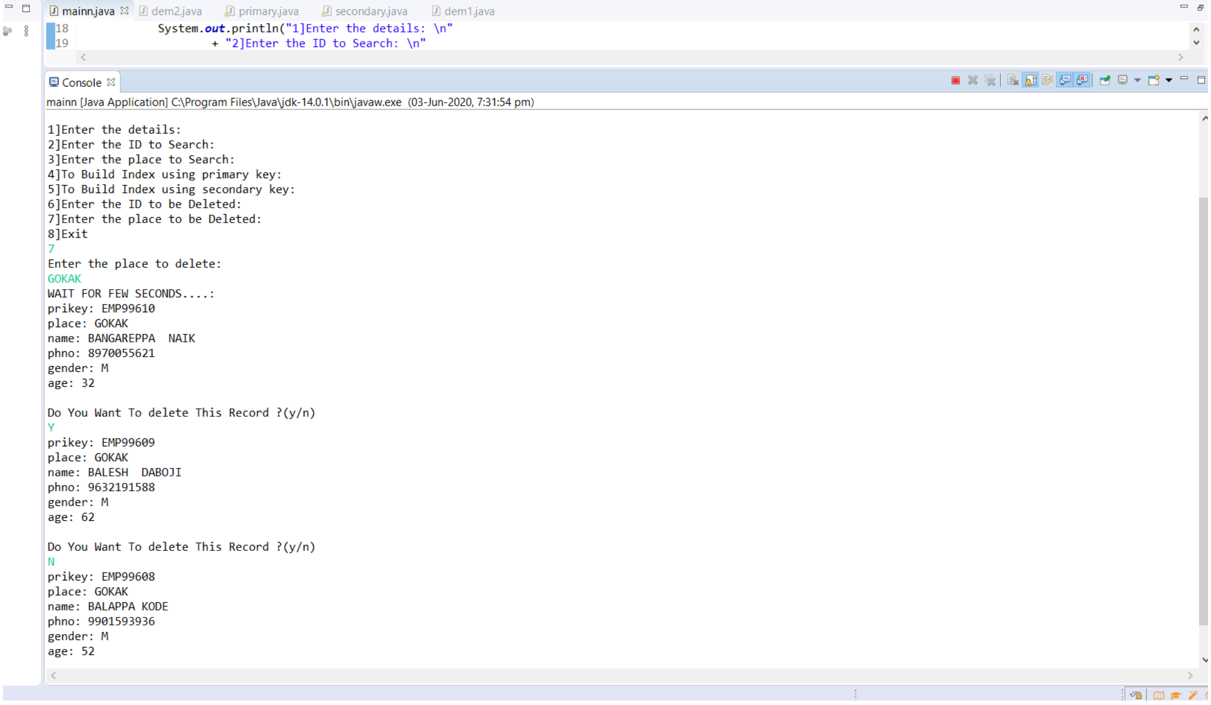
(i)Once the particular record is deleted, it is represented with an "*" at the beginning in the file containing the records:



```
EMP99901,GOKAK,KIRAN EMPEPPA BHUJANG,9164055149,M,29
EMP99902,GOKAK,KIRAN KHILARI,7026879110,M,24
EMP99903,GOKAK,KIRAN SHANTAPPA KENCHANNAVAR,9611297008,M,25
EMP99904,GOKAK,KIRAN SURESH DODDALINGANAVAR,8431934618,M,27
EMP99905,GOKAK,KIRAN TIMMANAGOL,9632323705,M,26
EMP99906,GOKAK,KIRTI  SATPUTE,9916273009,M,27
EMP99907,GOKAK,KISHOR PALLED,9538067930,M,24
EMP99908,GOKAK,KISHOR SIDDESHWAR KOLAGI,9379948898,M,22
EMP99909,GOKAK,KOMAL UMESH MATHAPATI,9591228029,M,28
EMP99910,GOKAK,KRISHNA  BUGADIKATTI,9916801797,M,29
EMP99911,GOKAK,KRISHNA  ULLAGADDI,9008374114,M,35
*MP99912,GOKAK,KRISHNA BHIMASHI PATIL,7795410382,M,24
EMP99913,GOKAK,KRISHNA EMPIKATTI,8431592136,M,45
EMP99914,GOKAK,KRISHNA R EMPIGANNAVAR,7090224012,M,44
EMP99915,GOKAK,KRISHNA SAKREPPGOL,9686282355,M,11
EMP99916,GOKAK,KRISHNARADDI S RAYARADDI,9900281802,M,18
EMP99917,GOKAK,KRISHNEGOUD BANAPPA BEERANAGADDI,8050427576,M,45
EMP99918,GOKAK,KRUTIKA MAHESH KOLAKI,7899891995,M,46
EMP99919,GOKAK,KULLUR MANING DASHARATH,9108809724,M,48
EMP99920,GOKAK,KUMAR KOLI,7090215216,M,47
```

Figure 4.1(i)

(j)When option 7 is selected:

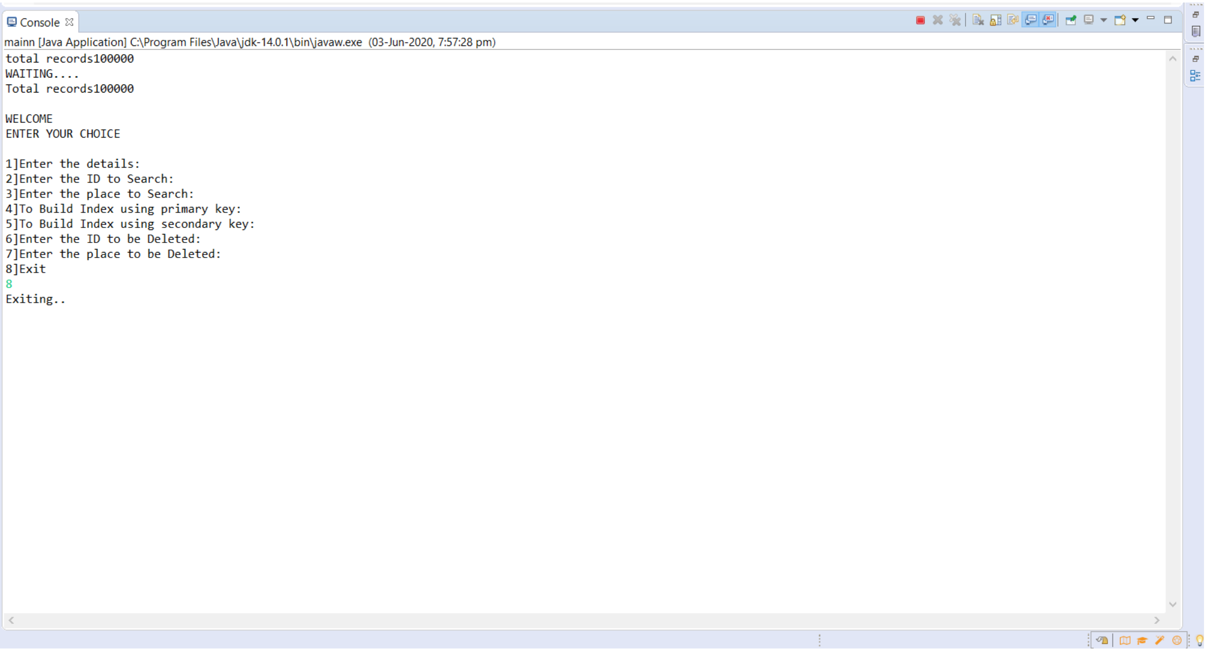

mainn.java
dem2.java
primary.java
secondary.java
dem1.java
Console
mainn [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (03-Jun-2020, 7:31:54 pm)

```
System.out.println("1]Enter the details: \n"
        + "2]Enter the ID to Search: \n"
```

```
1]Enter the details:
2]Enter the ID to Search:
3]Enter the place to Search:
4]To Build Index using primary key:
5]To Build Index using secondary key:
6]Enter the ID to be Deleted:
7]Enter the place to be Deleted:
8]Exit
7
Enter the place to delete:
GOKAK
WAIT FOR FEW SECONDS....:
prikey: EMP99610
place: GOKAK
name: BANGAREPPA  NAIK
phno: 8970055621
gender: M
age: 32

Do You Want To delete This Record ?(y/n)
Y
prikey: EMP99609
place: GOKAK
name: BALESH  DABOJI
phno: 9632191588
gender: M
age: 62

Do You Want To delete This Record ?(y/n)
N
prikey: EMP99608
place: GOKAK
name: BALAPPA KODE
phno: 9901593936
gender: M
age: 52
```

Figure 4.2(j)

(k)When option 8 is selected the execution ends:



mainn [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (03-Jun-2020, 7:57:28 pm)

```
total records100000
WAITING....
Total records100000

WELCOME
ENTER YOUR CHOICE

1]Enter the details:
2]Enter the ID to Search:
3]Enter the place to Search:
4]To Build Index using primary key:
5]To Build Index using secondary key:
6]Enter the ID to be Deleted:
7]Enter the place to be Deleted:
8]Exit
8
Exiting..
```

Figure 4.1(k)

4.2 RESULT:

GRAPH 1

This following graph shows the time taken to build an index using primary key. A linear increase in time is seen as the number of records increase. The time here is measured in milliseconds(msec).

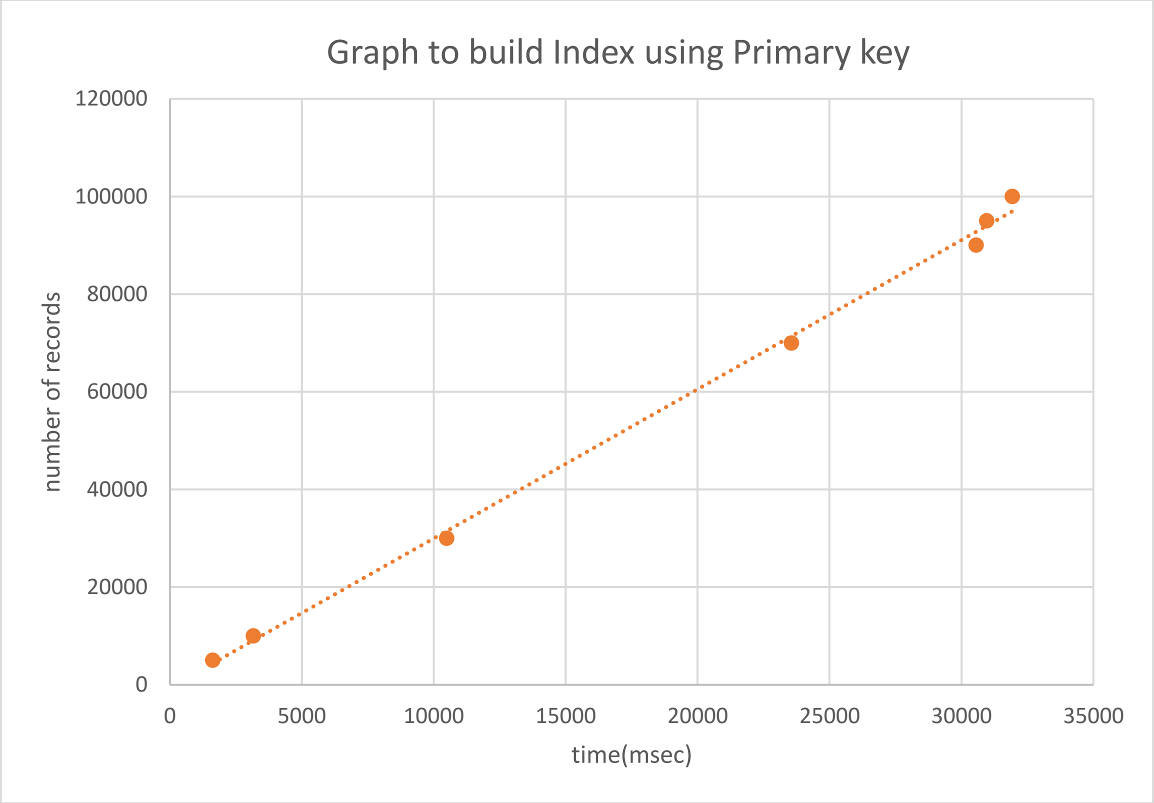


Figure 4.2(a)

GRAPH 2

This following graph shows the time taken to build an index using secondary key. A linear increase in time is seen as the number of records increase. The time here is measured in milliseconds(msec).

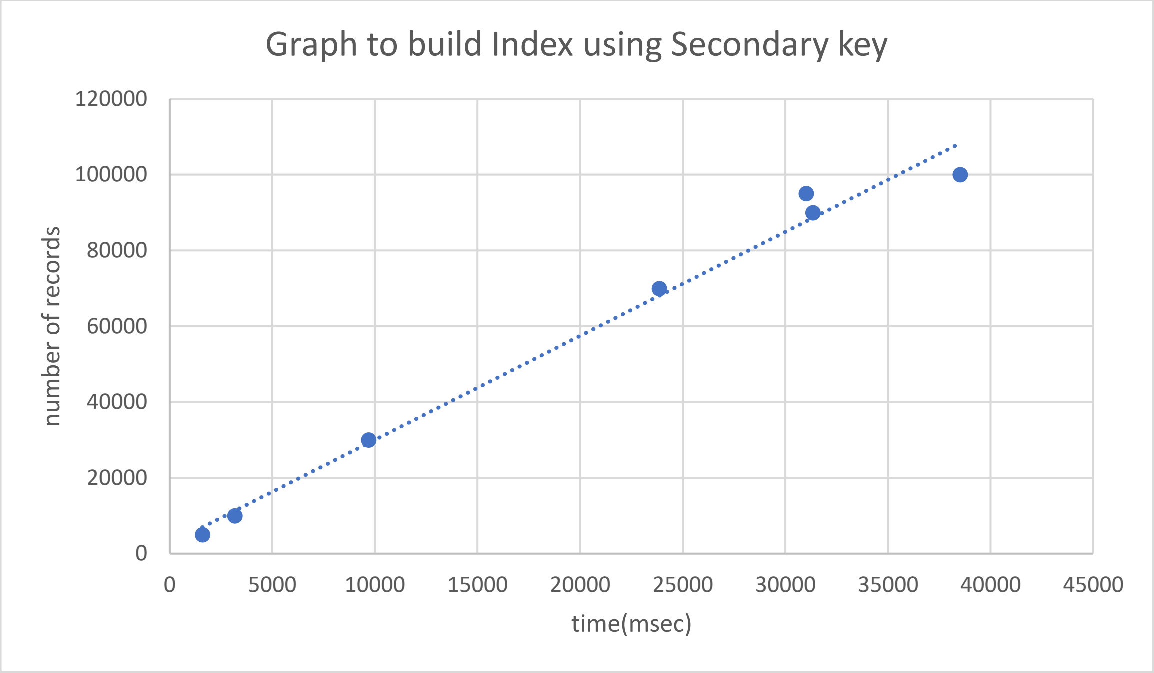


Figure 4.2(b)

CONCLUSION

INDEXING BASED ON PRIMARY & SECONDARY KEY has been successful implemented which helps in building indexes. Along with indexing other options like search, add, delete, are also provided.

Features:

1. Clean separation of indexes based on primary and secondary keys i.e., two separate files to facilitate easy understanding.
2. Quick and easy access to different data pieces with the indexes built.
3. Deleted record are easy to identify.

REFERENCES

1.Michael J. Folk, Bill Zoellick, Greg Riccardi: File Structures-An Object Oriented Approach with C++, 3rd Edition, Pearson Education, 1998.

2.www.geeksforgeeks.org/B-Tree-set-1-introduction-2/

3.en.wikipedia.org/wiki/B-Tree

4.searchsqlserver.techtarget.com/definition/B-Tree

5. www.javatpoint.com

6.btechsmartclass.com/DS/U5_T3.html