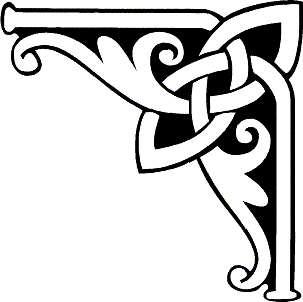
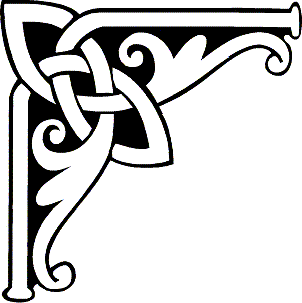
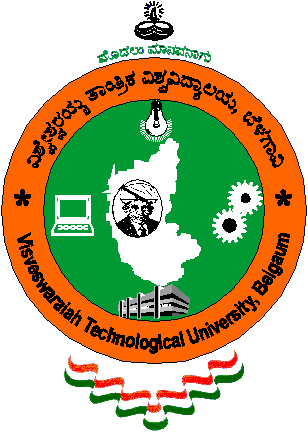
**Visvesvaraya Technological University**

**Belagavi, Karnataka**



bb



**A FILE STRUCTURE MINI- PROJECT REPORT**

**ON**

***" HOSPITAL RECORD MANAGEMENT USING PRIMARY INDEXING "***

Submitted to Visvesvaraya Technological University in partial fulfillment of the requirement for the award of Bachelor of Engineering degree in Information Science and Engineering.

**Submitted by**

**1. PUNEETH K S 4JN18IS065**

**2. SUBRAMANYA K S 4JN18IS100**

**Under the guidance of**

**Ms.Samara.Mubeena B.E, M. Tech**

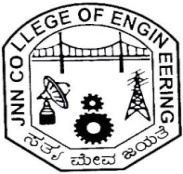
Asst. Professor, Dept. of IS&E

JNNCE, Shivamogga.

Department of Information Science & Engineering

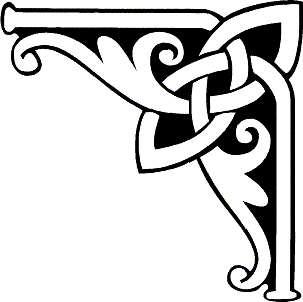
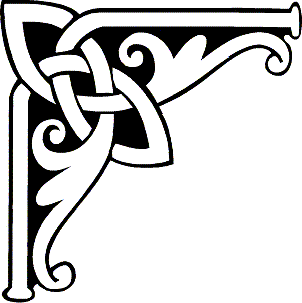
Jawaharlal Nehru National College of Engineering

Shivamogga - 577 204



**National Educational Society ®**

****



**J N N COLLEGE OF ENGINEERING**

**SHIVAMOGGA-577204.**

**DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING**

**CERTIFICATE**

This is to certify that the FILE STRUCTURES - MINI project entitled

***“HOSPITAL RECORD MANAGEMENT USING PRIMARY INDEXING*”**

**Submitted by**

**1. PUNEETH K S 4JN18IS065**

**2. SUBRAMANYA KS 4JN18IS100**

Students of 6th semester B.E. ISE, in partial fulfillment of the requirement for the award of degree of Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belagavi during the year 2020-21

**Signature of guide Signature of HOD**

**Ms.Samara.Mubeena B.E, M.Tech Dr. Sanjeev Kunte M. Tech., Ph. D**

**Asst. Professor, Prof. & Head,**

**Dept. of IS&E Dept. of IS&E**

**JNNCE, Shivamogga.**   **JNNCE, Shivamogga.**

**Name Signature**

**Examiner 1:**

**Examiner 2:**

**ACKNOWLEDGEMENT**

On presenting the File structure Mini – Project report on "**Hospital Management using Primary Indexing**", We feel great to express our humble feelings of thanks to all those who have helped us directly or indirectly in the successful completion of the project work.

We would like to thank our respected guide **Ms.Samra.Mubeena B.E ., M. Tech.**, Dept. of IS & E, who has helped us a lot in completing this task, for his continuous encouragement and guidance throughout the project work.

We would like to thank **Dr. Sanjeev Kunte M.Tech.,Ph.D,** Professor and Head, Dept of IS & E, JNNCE, Shivamogga and **Dr. P Manjunath**, The Principal, JNNCE, Shivamogga for all their support and encouragement.

We are grateful to **Department of Information Science and Engineering** and our institution **Jawaharlal Nehru National College of Engineering** and for imparting us the knowledge with which we can do our best.

Finally, we would like to thank the whole teaching and non teaching staff of Information Science and Engineering Department.

Thanking you all,

**PUNEETH K S (4JN18IS065)**

**SUBRAMANYA K S (4JN18IS100)**

**ABSTRACT**

The “Hospital Records Management using primary indexing” is an application for maintaining the records of hospital staff, and patient. This project is mainly concerned with developing a system where an Admin can add the Doctor , Nurse , Receptionist and Ward details. Also admin can view and Delete the records of the Staff. Doctor and Nurse can view the Registerd Paient details . Receptionist is going to register the patient.The main aim of this application is to computerize the process of Hospital Records management. This application is designed to provide a user-friendly access for the Admin to add , view or delete the records. This application saves time and also reduces the cost which is costly if carried out manually.

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ACKNOWLEDGEMENT** | **i** |
|  | **ABSTRACT** | **ii** |
|  | **CONTENTS** | **iii** |
|  |  |  |
|  |  |  |
| **Chapter 1** | **INTRODUCTION** | **01** |
|  | 1.1 Introduction to File Structures |  |
|  | 1.2 Primary Indexing |  |
|  |  |  |
| **Chapter 2** | **ALGORITHM** | **04** |
|  |  |  |
| **Chapter 3** | **IMPLEMENTAATION** | **06** |
|  | 3.1 Problem Statement |  |
|  | 3.2 Objectives of the project |  |
|  | 3.3 About the programming language |  |
| **Chapter 4** | **SNAPSHOTS** | **07** |
|  | 4.1 Front page |  |
|  | 4.2 Admin operation after Login |  |
|  | 4.3 Patient registration by Receptionist |  |
|  |  |  |
| **Chapter 5** | **CONCLUSION** | **14** |
|  |  |  |
| **Chapter 6** | **REFERENCES** | **15** |

**CHAPTER 1**

**INTRODUCTION**

**1.1 Introduction to File Structures**

File Structures is the Organization of Data in Secondary Storage Device in

Such a way that minimizes the access time and the storage space.

**1.1.1 File Structure: It is a combination of**

* Representation for data in files and
* The operations for accessing the data

It allows applications to read write and modify data search .An improvement in file Structure design make a application hundred times faster .To access the data faster From the storage disk we use file structure .Disks are slow which are used to pack Thousands of megabytes of data.

**1.1.2 Records and its types:**

Are collection of fields, possibly of different Data types, typically in fixed number of sequence.

The fields of are also be called members. For example, a date could be sorted as a record containing a numeric year field, a Month field represented as a string, and a numeric day of month field. A record are distinguished from arrays by the fact that the number of fields Is typically fixed, each field has a name, and that each field may have a different types.

A record type is a data type that describes such values and variables. The Definition includes specifying the data type of each field and an identifiers by which it can be accessed.

**1.1.3 Why we need File Structure??**

As we know without proper structure of organizing is ordinary system, it will generate some types of problems. So avoiding this kind of problems we go for File Structure.

The File structure means its tell about how the system will stored and access the record from memory and also its tell how the disk are performed its tasks and also how to speed up the execution of transferring data.

Data processing from a computer science prospective:

-storage of data

-organization of data

-access to data

This will be built on your on your knowledge of Data Structures. Some of the important concepts used

**Field Structures:**

(1) Fixed Length Fields: The method to organize fields is by limiting the maximum size of each field. The advantage in this method is that since the size of each field is fixed, the entire field can be read at once.

(2) Length Indicator Fields: The length of each field is specified as a prefix to actual data (3) Delimited Fields: Any special character which is not a part of actual data can be used as separator.

**Record Structures:**

(1) Fixed Length Record Structure: Each record is stored in fixed size. The size can be determined by adding the maximum space occupied by each field and some space reserved for the header data.

(2) Fixed Field Count: The number of the fields in each record is fixed.

(3) Index Structure for Records: An index is a collection of key field and reference field.

(4) Key Field: Key Field is a member of record which can uniquely identify the record.

(5) Reference Field: Reference Field contains the value that points to the address of the corresponding record in the file.

**1.2 Primary Indexing:**

A primary index consists of all prime-key attributes of a table and a pointer to physical memory address of the record of data file. To retrieve a record on the basis of all primary key attributes, primary index is used for fast searching. Binary search is done on index table and then directly retrieve that record from physical memory. It may be sparse.

**Advantages of Primary index**

1. Search operation is very fast.
2. Index table record is usually smaller.
3. A Primary index is guaranteed not to duplicate.

**Disadvantages of Primary index**

1. There is only one primary index of a table. To search a record on less than all prime-key attributes, linear search is performed on index table.
2. To create a primary index of an existing table, record should be in some sequential order otherwise database is required to be adjusted.

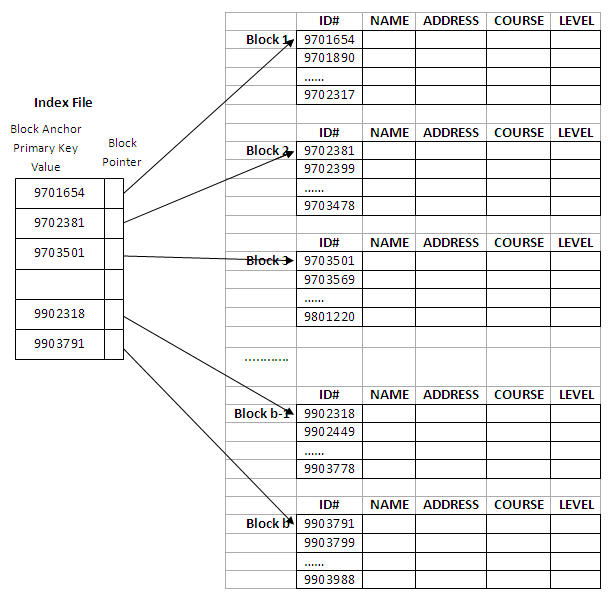


Fig: An Example of Primary Indexing.

**CHAPTER 2**

**ALGORITHM**

**Createindex function.**

1. Open the txt file in input mode using fstream.

2. While not the end of file.

1. Get the position.

2. Erase the buffer content.

3. Extract the characters from file and append it to buffer.

4. If the buffer first position is not equal to \*

1. Check for buffer empty if so then break.

2. Call extracted function and assign to id.

3. assign id to id list.

4. Assign pos to addlist.

3. End while.

4. call sort function.

5. Erase the Buffer.

6. End of function.

**Insert function.**

1. Read the details with id.

2. Append the details with id to buffer.

3. Open the txt file in input mode.

4. Append the buffer elements to file and close the file.

5. Assign id to id list.

6. Assign pos to addlist.

7. call sort index function.

8. End of function.

**ExtractId function.**

1. Get buffer as parameter.
2. Until the buffer has first delimiter symbol
3. Assign buffer element to id.

3. Return the id.

**SortIndex function.**

1. Declare variables.
2. For i in range of 0 to count.
3. For j in range of i+1 to count.
4. If id in Idlist at ith position is greater than id in Idlist at jth position, then sort the id in idlist
5. End if.

2. end for.

3. End for.

4. End.

**Search function.**

1. Get the key as parameter.
2. Declare Variables.
3. Erase the buffer element.
4. Call searchIndex function with parameter key to get the position
5. If postion equals to -1 then print record not found.
6. Else if position is greater than or equal to 0.
7. Open the txt file.
8. Get the position from addlist.
9. Seek the file
10. Extract the characters from file and append it to buffer.
11. Print the message and details.Close the file.

7. End.

**SearchIndex function.**

1. Assign the low to zero and high to count.
2. While low is less than mid.
3. Mid element is equal to sum of low and hight and divide the sum by 2.
4. If mid element of Idlist is equal to key. Then return mid.
5. Else if mid element of Idlist is less than key, Then low equal to mid+1.return mid
6. Else if mid element of Idlist is greater than key, Then high equals to mid-1. Return mid
7. Else return -1;
8. End.

**Remove function.**

1. Declare variable.
2. Call searchIndex function with key as parameter and assign to position.
3. If position is greater than 0.then.
4. Open txt file .
5. Get the position from addlist.
6. Seek the file
7. Put \* at the beginning of deleted file.
8. Move the next file id to deleted file location id in idlist and same in addlist.
9. Decrement the count.
10. Else
11. Print the record not found.
12. Display updated file.

**CHAPTER 3**

**IMPLEMENTATION**

**3.1 Problem statement**

This project is mainly concerned with developing a system where an receptionist can add patients details, search the details of patients using primary indexing, remove the details. The admin can add doctor nurse and get the details of them.The hospital record management system is developed using Primary Indexing**.**

**3.2 Objectives of the project**

* The project Hospital Record Management system has been made to automate system.
* To allow only authorized user to access various functions and process available in the system.
* This system will help receptionist to add patients, remove patients and view the details.
* This system will help admin to add doctors, nurse, ward, receptionist, remove them and search them using their id’s.

**3.3 About the Programming Language C++**

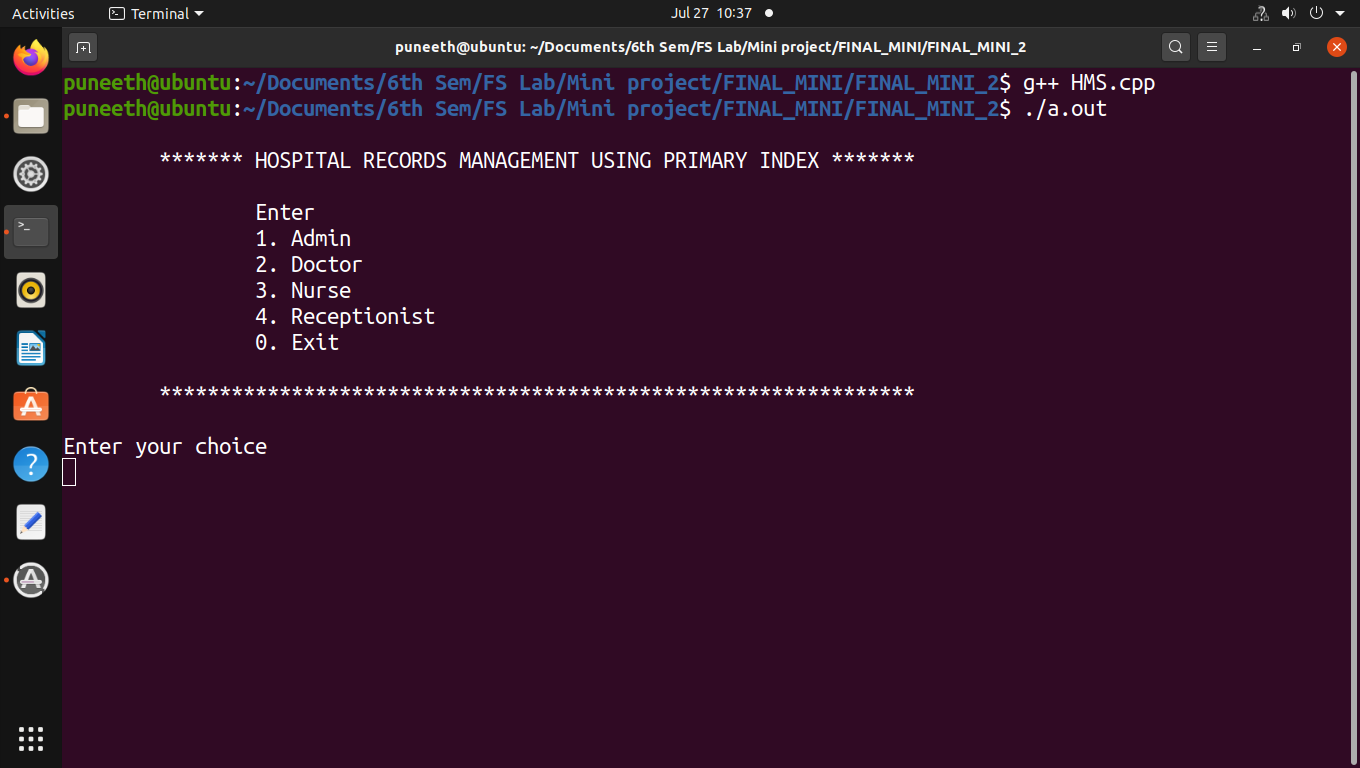
In the program “Time Complexity of Search and Traverse of a Key in primary indexing”, we have used C++ as a medium to compile and run the program because of its versatile uses in the industry.

It was designed with a bias toward system programming and embedded, resource-constrained and large systems, with performance, efficiency and flexibility of use as its design highlights. C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications, including desktop applications, servers (e.g. e-commerce, web search or SQL servers), and performance-critical applications. C++ is a compiled language, with implementations of it available on many platforms. Many vendors provide C++ compilers, including the Free Software Foundation, Microsoft, Intel, and IBM.

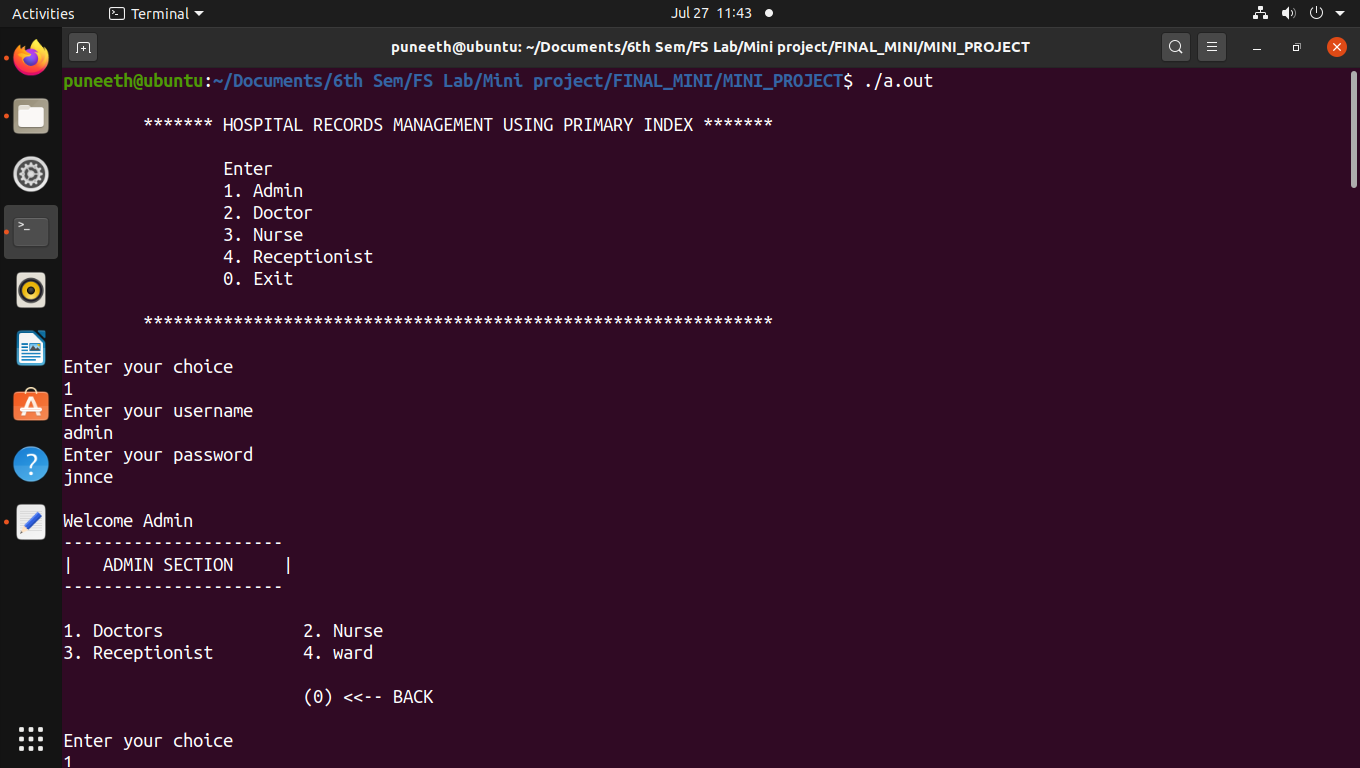
**CHAPTER 4**

**SNAPSHOTS**

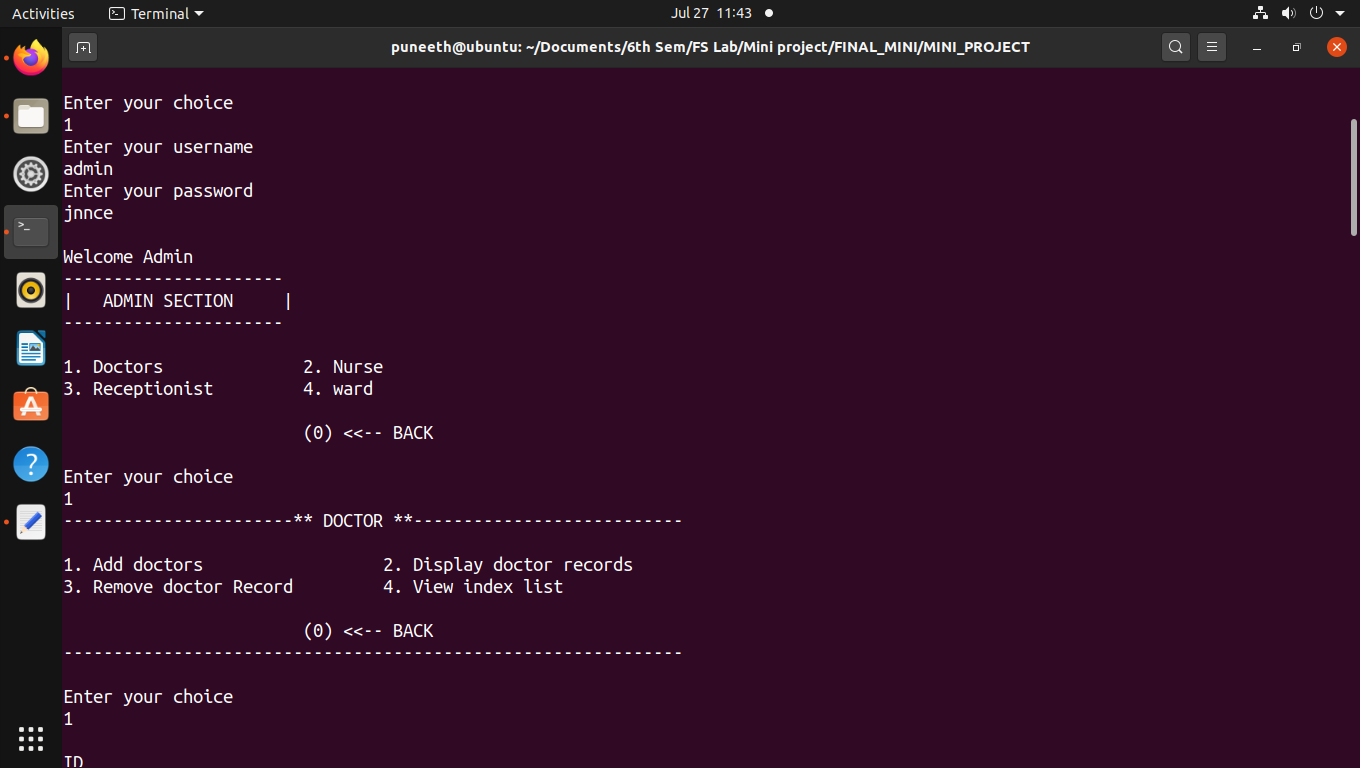
**4.1 Front page**

****

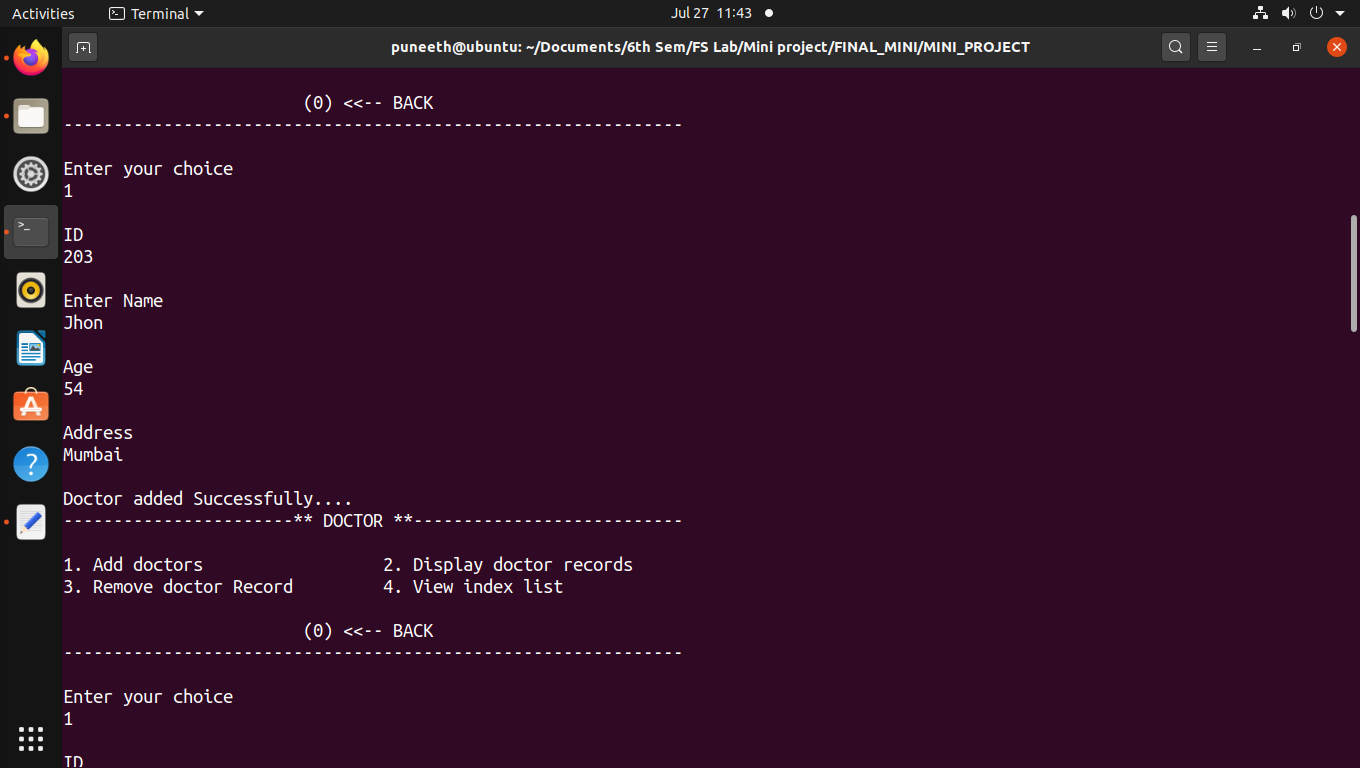
**4.2 Admin operations after login**

****

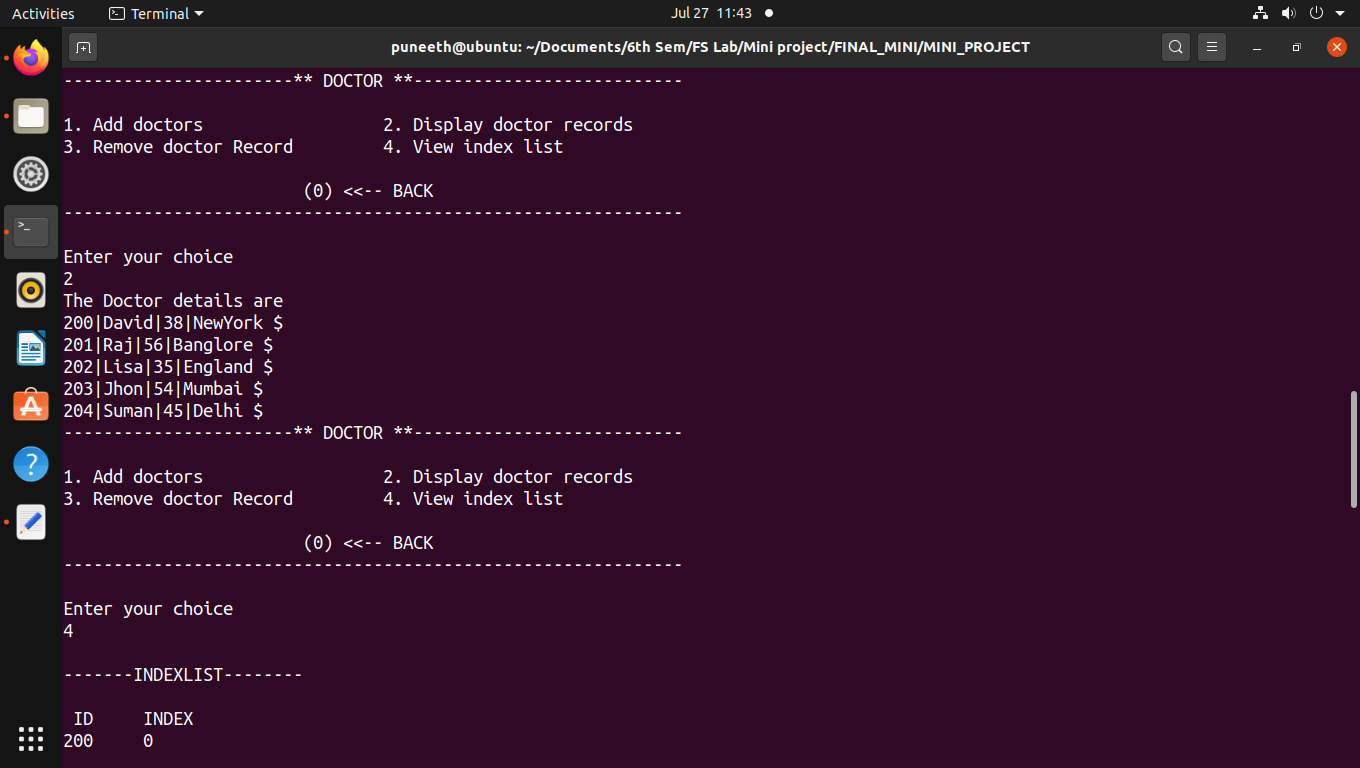
**4.2.1 Adding Doctor details to the file**

****

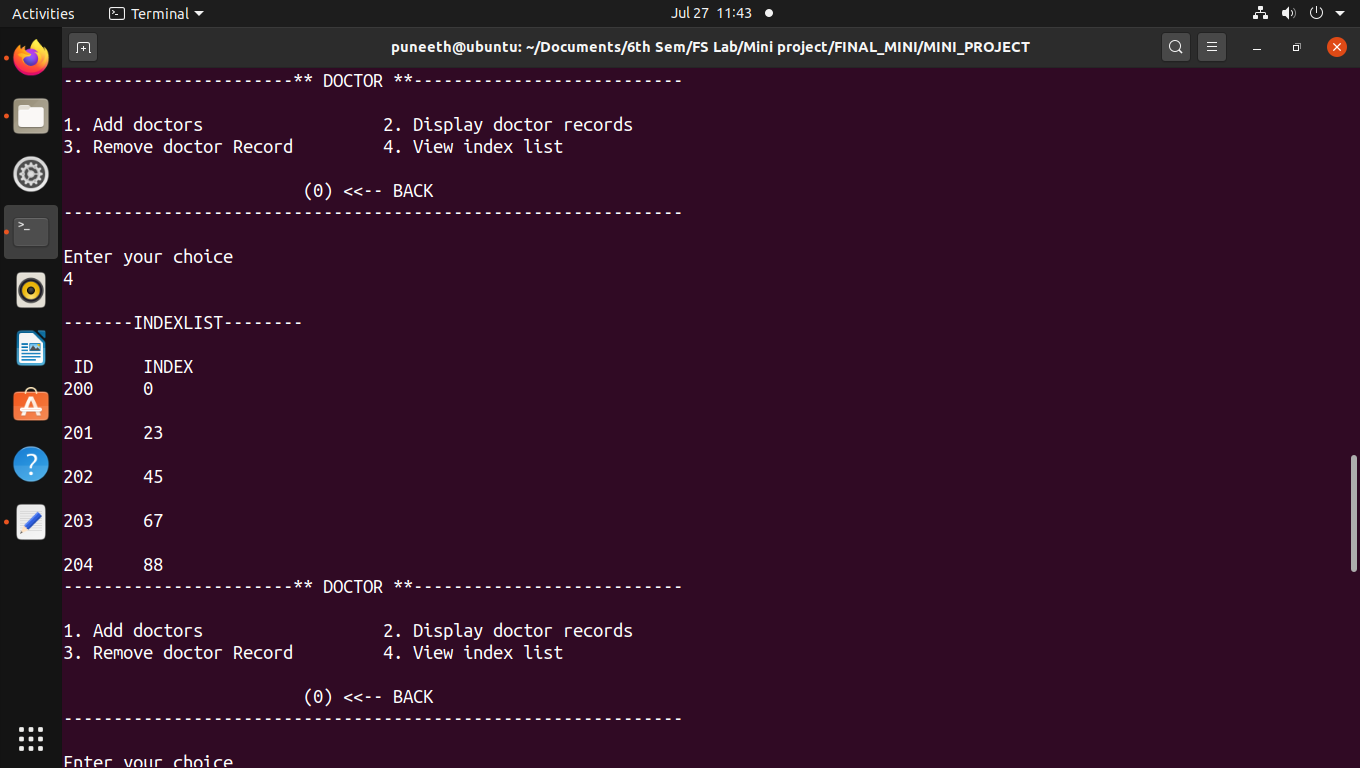
**4.2.2 Adding Doctor details to the file**

****

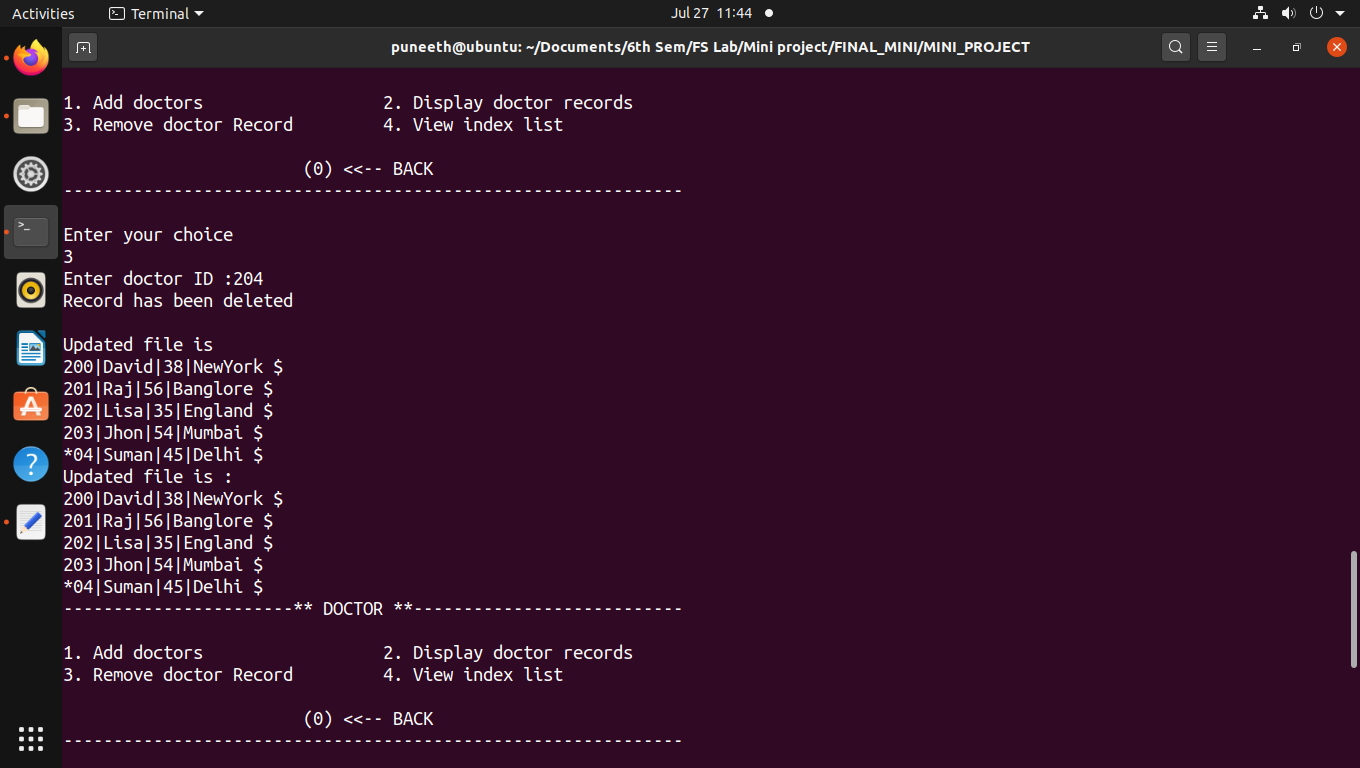
**4.2.3 Display the Doctor deatils**

****

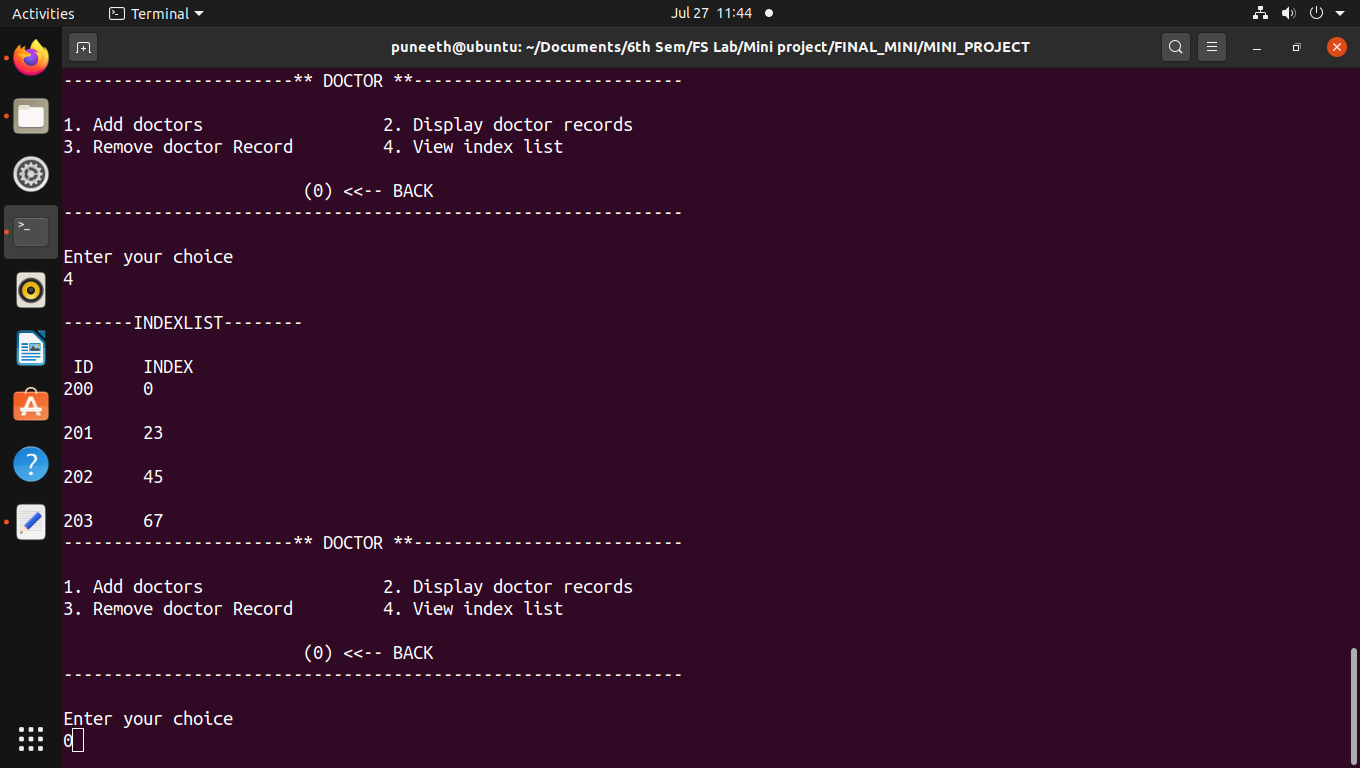
**4.2.4 Display the index List of Doctors**

****

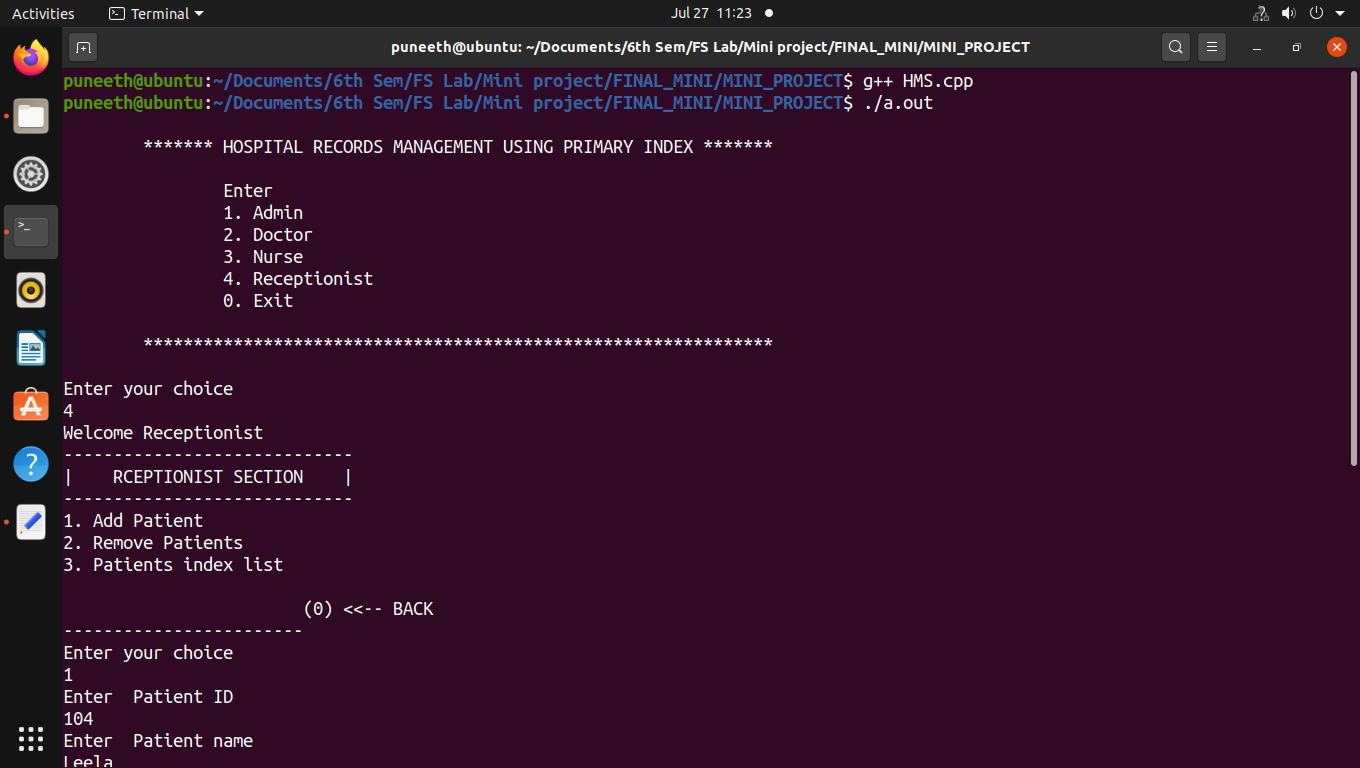
**4.2.5 Remove the doctor record from the file**

****

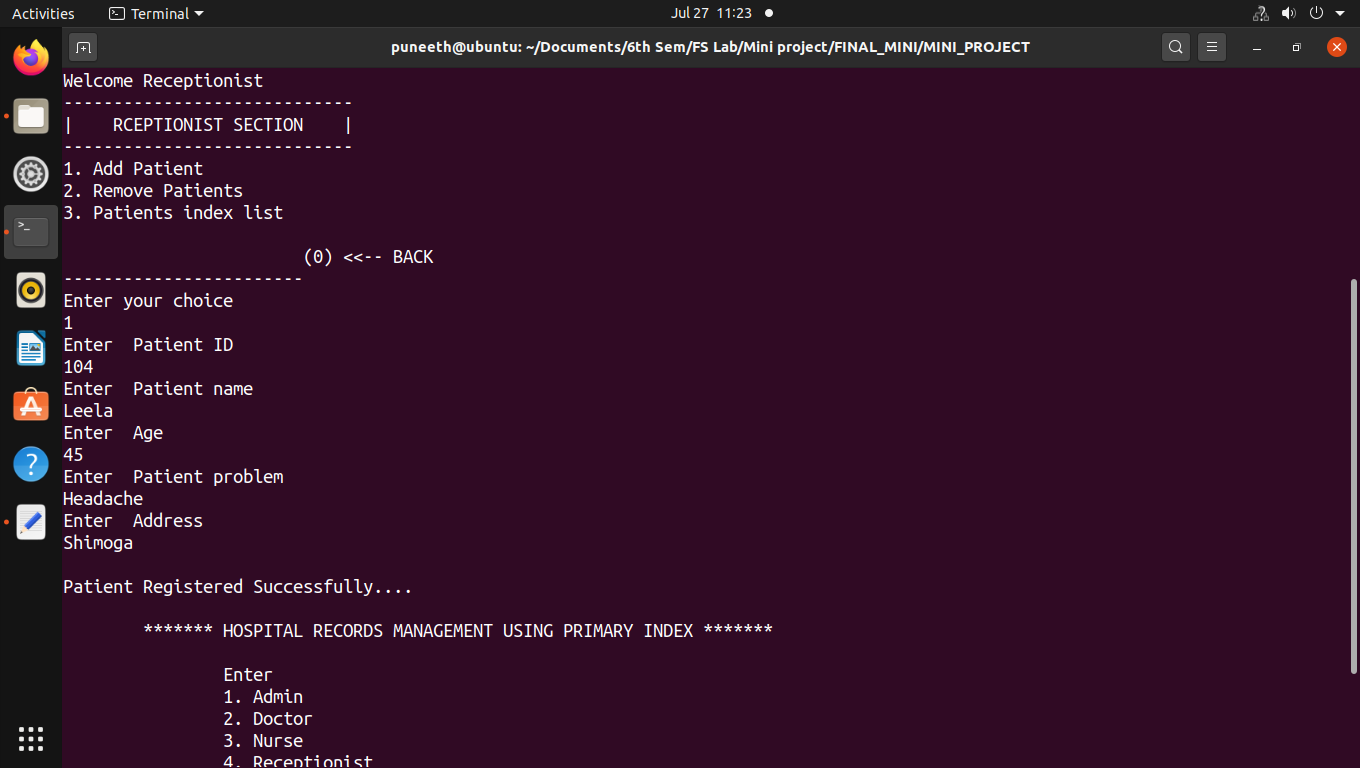
**4.2.6 Updated Index list after removing the doctor record from file**

****

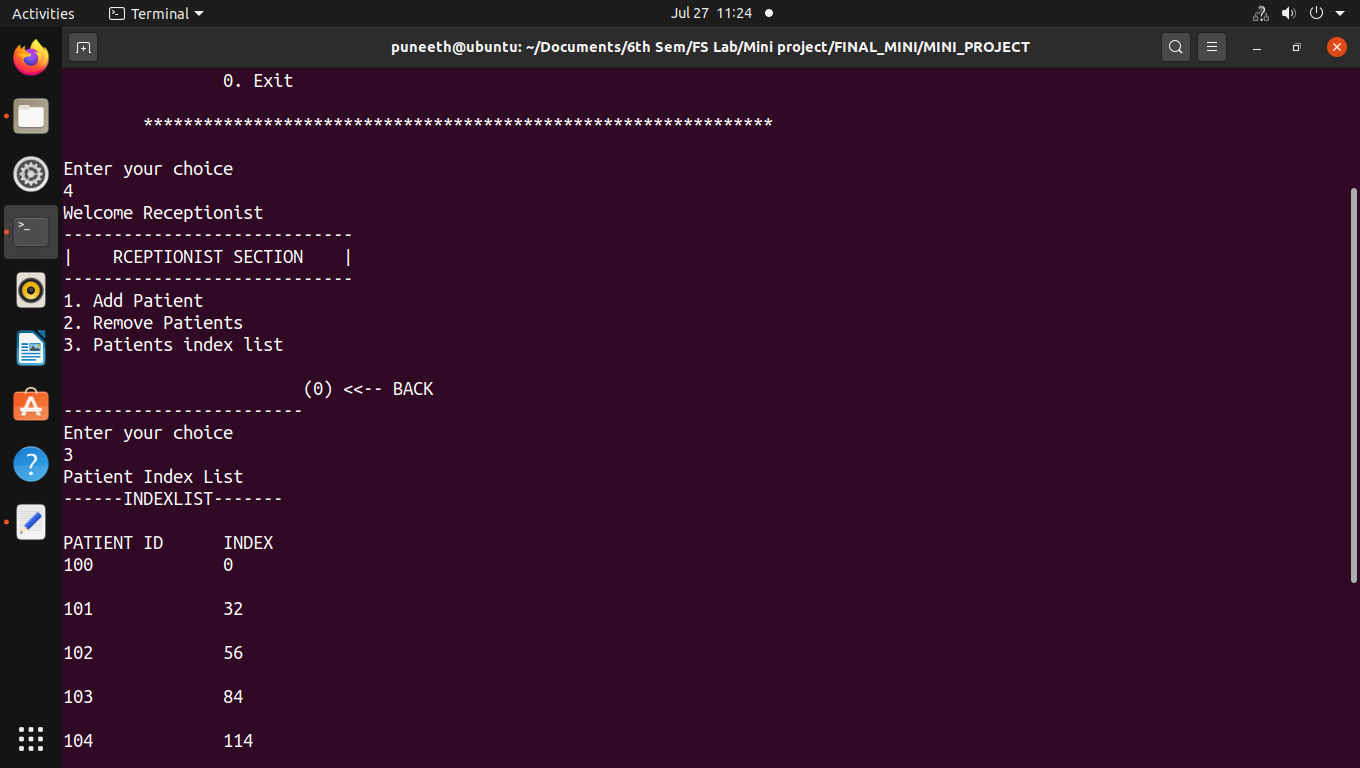
**4.3 Patient Registration by Receptionist**

****

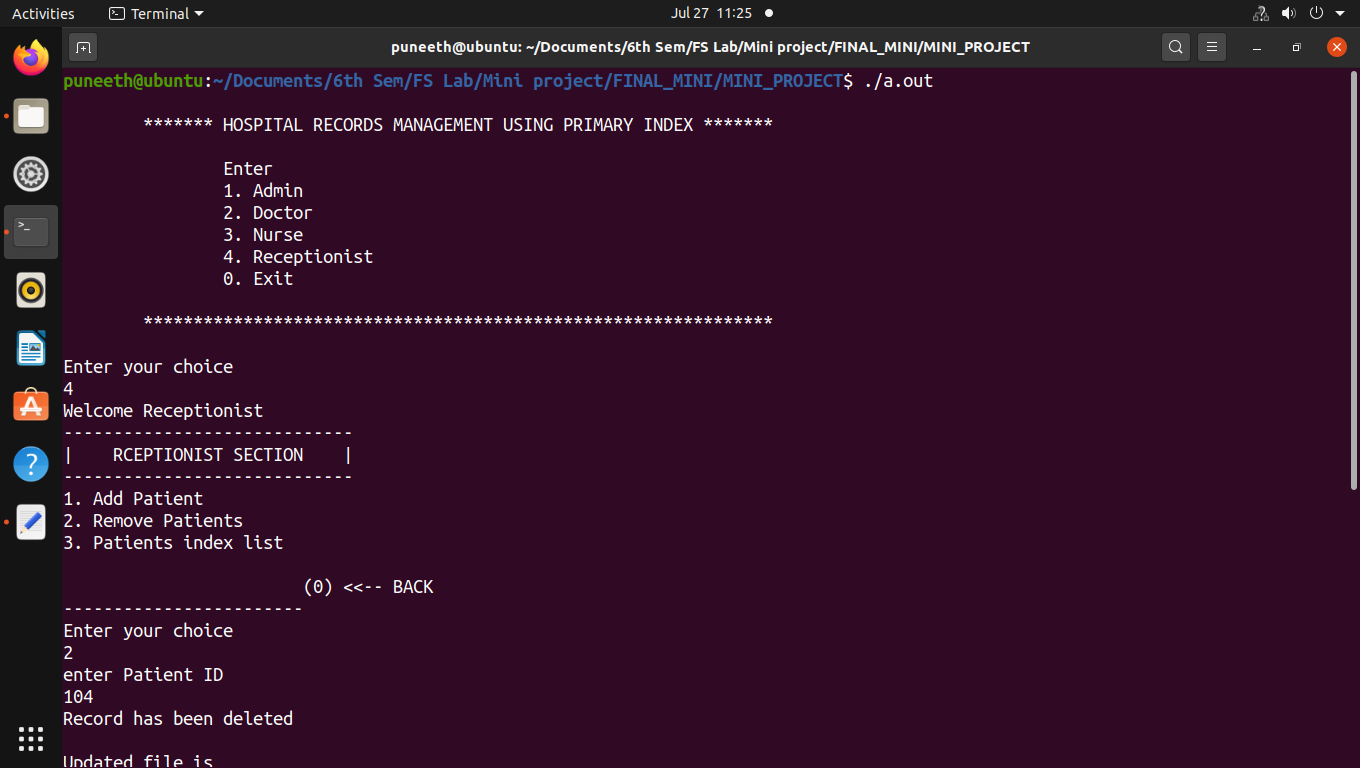
**4.3.1 Patient Registration by Receptionist**

****

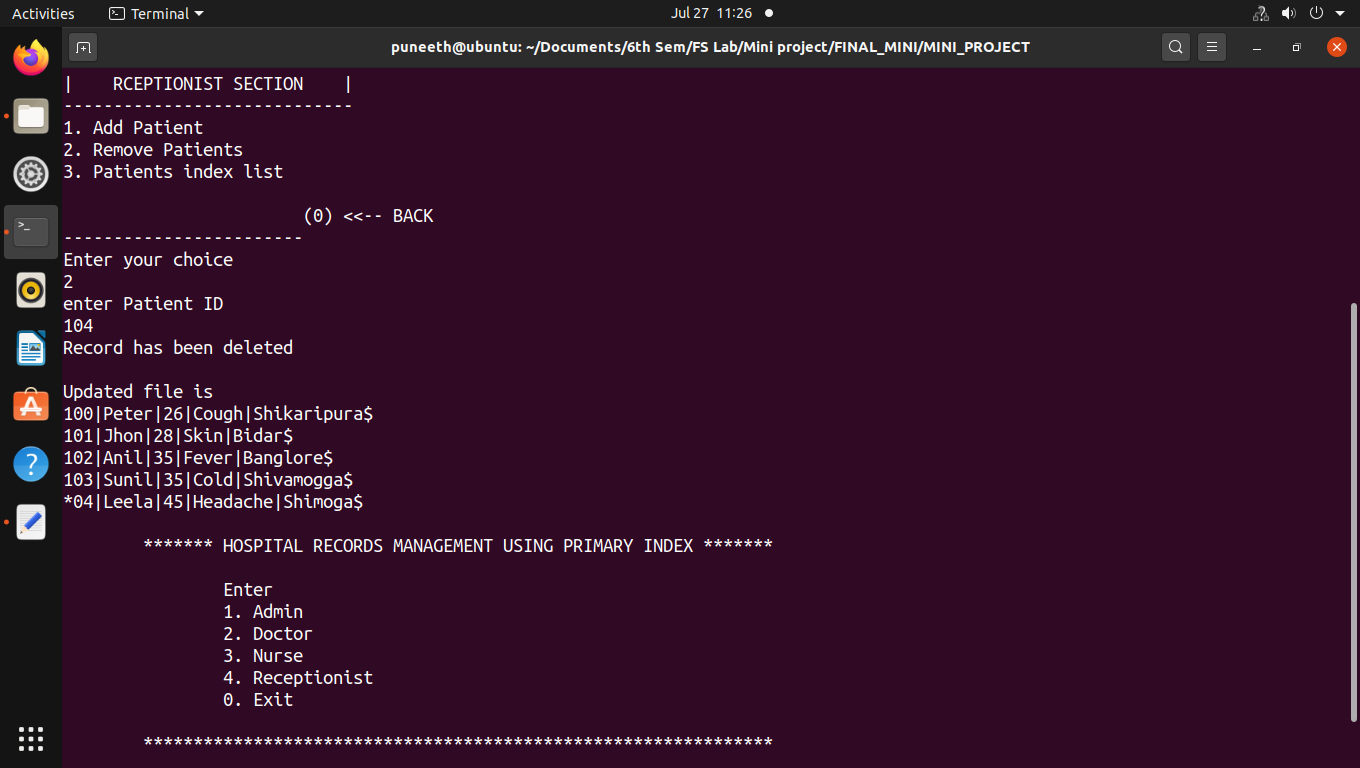
**4.3.2 Index List of Patient ID**

****

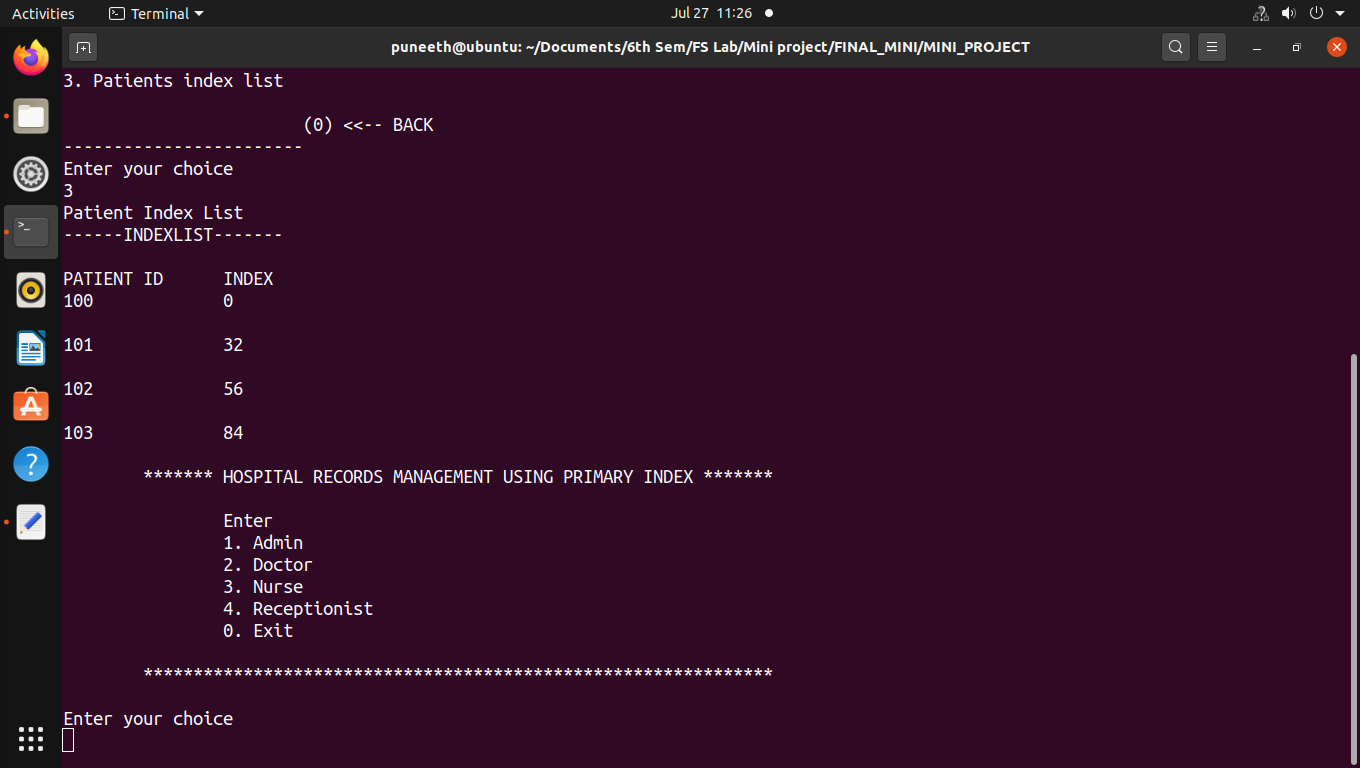
**4.3.3 Remove patient record from the file**

****

**4.3.4 Remove patient record from the file**

****

**4.3.5 Index List after removing the patient record from the file**

****

**CHAPTER 5**

**CONCLUSION**

The “Hospital Record Management System using primary indexing” executed well without any exceptions . The total time complexity of the program could be improved with better searching techniques. The index of the record was stored in sorted order.The records can be viewed by the user explicitly using display() functions.

In this program, we have used variable length records and Primary index on primary key, so that the user gets the advantage of using these type of records. The program runs well but there is always a scope for improvement. Better search techniques can be implemented and searching can be done based on other aspects too.

**CHAPTER 6**

**REFERENCES**

**Books:**

1. File Structures with C++ by Michael J Folk.
2. File Structures using C++ by K R Venugopal, K G Srinivasa, P M Krishnaraj,Tata McGraw-Hill ,2009, Referred Page No-4,39-40.