Hostel Room Allotment System

Team Members

Asutosh Mohapatra (17BEC0141)

Parshuram Kumar Gupta (17BEC0240)

Report Submitted For The Final Project

Review Of Data Structures And Algorithms

Course Code-: CSE 2003

Slot-: B1

Under The Guidance Of

Professor Dr. VISHNU SRINIVA MURTHY

ABSTRACT

This project is a room allotment system that can actually replace the existing system. After attending VIT room counseling we came to an idea to implement the room counseling processing in a better way such that students should not feel any difficulty in finding their roommate or room in a particular Hostel Block. It is fully functional and provides the same features provided by the current system used by VIT for hostel room allotment, plus some extra features such as finding the occupants of a room or finding a student in the room matrix. We will be using the C++ programing language.

ACKNOWLEDGMENT

We would like to thank our Prof. VISHNU SRINIVA MURTHY for his constant support and guidance that enabled us to make a wonderful project on “HOSTEL ROOM ALLOTMENT SYSTEM”.

We have learned a lot during the project and it would not have been possible without the support system provided by VIT authorities. It was due to the mutual understanding, support, interest, cooperation, and hard work of all the group members that enabled us to complete this project.

INTRODUCTION

In this project, we have created a tree of hostel blocks and then each block has sub-nodes bed count wise and each bed count has rooms linked room number wise. The hostel information was provided from a TXT file. We have also implemented a priority queue that has stored the student information. Priority will be decided as per the CGPA of students. To make the process easier we have created a linked list that stores every student's information in descending order of CGPA. The student information was provided in an excel TXT file. We will call students with the highest priority wise. Then he will be asked to select a room and his roommates. As per the data provided, we will be performing all the operations.

DATA STRUCTURES USED-:

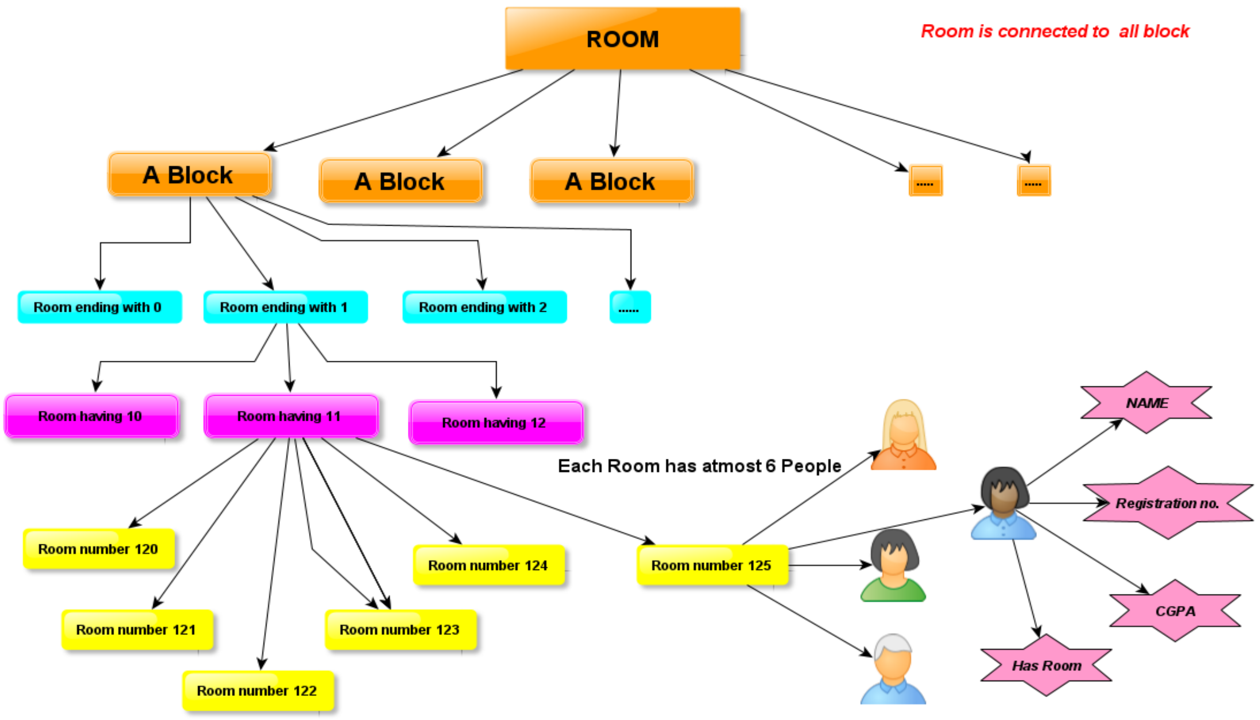
1) TRIE -:

A TRIE (also called radix tree or prefix tree) is a data structure that stores data (generally strings) in the fashion of a tree. The words with common prefixes originate from the same branch. In our project, we modified this widely used Data Structure such that the nodes of the trie actually contain a room in them.

2) PRIORITY QUEUE-:

A priority queue is any data structure that has some type of priority deciding item in each node. In this project, the priority queue is a queue in which students are arranged in the decreasing order of their CGPA. An element 'has room' is present in every student struct, if the element is 0 only then a traveling pointer stops at that node. We have used the Binary Search Tree to create the priority queue.

VISUAL OF PROPOSED MODEL



CODE ARCHITECTURE ANALYSIS-:

1) CREATING A DATABASE-:

Created a student database. In the database, we have Registration ID, Name, CGPA of every student. Created a hostel database. In this database, we have Room Numbers and corresponding Block Name and Bed Count of that room.

2) READING STUDENT DATA AND MAKING OF A PRIORITY QUEUE

Created "readStudent" Function to read student names, ids, and their respective CGPA from the studentlist.txt file. Created a Binary Search Tree using CGPA as the key (i.e. comparing CGPA values). Functions used for this operation -: "insert" and "new node". Created a Linked List of students keeping their CGPA's in descending order. Function used-: "inorder".

3) CREATED A TRIE TREE OF HOSTELS AND ROOMS AVAILABLE IN DATABASE

Created "readHostel" function to read available hostels and rooms in those blocks. Used function "createHostel" to get the address of the head node of the TRIE tree and pass the rest information to "newBlock" function. Used function "newBlock" to create new blocks if not already present in the tree. Used function "newRoom" to assign new rooms in those respective blocks.

4) PERFORMING ROOM BOOKING OPERATIONS

i) ROOM BOOKING

- First we go to "roomBooking" function. Then the first student name and id in the Linked List is displayed and we ask the student how many bedded rooms he wants.
- Then we display room numbers available for those many beds and ask the student to choose a room from the above options.
- After the student enters the block name and room number.
- Then we traverse through all the blocks until the desired block is found.
- The in the block according to the bed count he chooses we travel through all vacant rooms of that much bed count.
- We assign the first person in the room to the group leader.
- Then if the bed count is greater than 1 then we ask him to give registration numbers of the students he wants as his roommates one by one.
- We check if the student has allotted any room previously or not. If not then put him in that room.
- If the student has already allotted any room or not present in the database then we display the same and ask to re-enter id correctly.
- After the booking is done we show the status that booking is successful and corresponding occupant details.

ii) Room Searching

- In this option, we ask the user to enter the block name and room number.
- Then we traverse through all the blocks until the desired block is found.
- Then inside the block, we traverse through all the rooms irrespective of bed count.

- If that room is found we show all the occupants in the room and their corresponding details.
- If any room with that number is not found in the block we display that the room number entered is invalid. Enter the room number again.

iii) HOSTEL BLOCK STATUS

- In this option, we display the total number of rooms available according to their bed count.

iv) CHECKING IF A STUDENT HAS ALREADY ALLOTTED ANY ROOM OR NOT

- In this option, we ask the user to enter a student id.
- If a student is not present in the database we display student information not available.
- Otherwise, we display the room allotment status of that student.

v) THEN TO KEEP A TRACK OF NEXT STUDENT IN QUEUE WE HAVE THIS OPTION

- In this option, we display the name and registration number of the next student who has to come for room counseling.

vi) CHECKING ALL STUDENT DETAILS

- This option displays all the students sorted according to their CGPA and their room allotment status and name and id respectively.

CODE-:

```
#include<iostream>
#include<fstream>
#include<string>
#include<string.h>
#include <bits/stdc++.h>
using namespace std;

struct student
{
    string name; //to store name of student
    char currentBlock; //
    string id;//to store registration id
    float cgpa;//to store his cgpa
    int roomNo;//to store his room number
    bool isBooked;// to store details if he has alloted any room
or not.
    struct student *right,*left,*nextll,*nextRoommate; //right
and left used in the BST, nextll is used in sorted linked list,

//nextRoommate is to store address of his next Room-mate.
};

struct room
{
    int room_number; //to store the room number
    int status;//to store 1 if room is booked else 0 if room not
booked
    struct room *nextRoom; //to store next room address of same
having same bed count (4 types of rooms are there)
```

```
    struct student *s; // to store the address of group leader
student if someone books the room

};


struct block

{

    char block_name; //To store the block name

    int  avail_one,avail_two,avail_three,avail_four;//to  store
the number of rooms available as per the bed count

    struct block *nextBlock; //to store the address of a new
block.

    struct room *one,*two,*three,*four; // to store the address
of one bedded two beded ... room heads


};


struct hostel

{

    int total_one,total_two,total_three,total_four; //to store
the total number of rooms available as per bed count

    struct block *firstBlock; //to store the address of first
block

};


struct room* newRoom(struct room* baseRoom,int roomNo);

struct block *newBlock(struct block *fb,char block_name,int
num_beds,int roomNo);

struct hostel *createHostel(struct hostel *base,char
block_name,int num_beds,int roomNo);

struct hostel *readHostel(struct hostel *base);

void hostelStatus(struct hostel *base);

void trieTraversal(struct hostel *base,int numBeds);

void roomBooking(struct hostel* base,struct student*
llroot,struct student* root);
```

```
void trieSearch(struct hostel* base);
struct student *newnode(string id,string name,float cgpa);
struct student* inseart(struct student* node,string id,string
name,float cgpa);
void inorder (struct student* root,struct student* llroot,struct
student* temp);
bool ifPresent(struct student* root,string id);
struct student *readStudent(struct student* root);
struct student* funcIfPresent(struct student* root);
struct student* nextStudent(struct student* llroot);
void studentDetails(struct student* root);






main()
{
    cout<<"***********WELCOME TO HOSTEL ROOM ALLOTMENT SYSTEM
By-: ASUTOSH MOHAPATRA & PARSHURAM KUMAR GUPTA**********"<<endl;
    struct student* root = NULL;
    struct student* llroot =new (nothrow) (struct student);
    struct student* temp = new (nothrow) (struct student);
    root = readStudent(root);

    struct hostel* base = NULL;
    base = readHostel(base);

    inorder(root,llroot,temp);
    llroot = llroot->nextll;

    struct student* currentNode =NULL;
    int operation=-1;


    while(operation!=0)
    {
        cout<<"\n"<<"TO BOOK ROOM ENTER 1"<<endl;
        cout<<"TO SEARCH FOR A ROOM ENTER 2"<<endl;
        cout<<"TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3"<<endl;
        cout<<"TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM
OR NOT ENTER 4"<<endl;
        cout<<"TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR
COUNCELLING ENTER 5"<<endl;
        cout<<"TO CHECK ALL SUDENT DETAILS ENTER 6"<<endl;


        cout<<"TO TERMINATE PROCESS ENTER 0"<<endl<<endl;
        cout<<"ENTER: ";
        cin>>operation;
        if(operation==1) roomBooking(base,llroot,root);
        else if(operation==2) trieSearch(base);
        else if(operation==3)  hostelStatus(base);
        else if(operation==4) funcIfPresent(root);
        else if(operation==5)
        {
            llroot = nextStudent(llroot);
            cout<<"Next Student In Queue is : "<<llroot->name<<"
With Registration Id: "<<llroot->id<<endl;
        }
        else if(operation==6) studentDetails(root);
        else if(operation==0) break;
    }
    cout<<"TASK TERMINATED"<<endl;
}
```

```
struct room* newRoom(struct room *baseRoom,int roomNo)
{
    struct room* temp=baseRoom;
    if(baseRoom==NULL)
    {
        temp = new (nothrow) (struct room);
        temp->room_number = roomNo;
        temp->status = 0;
        temp->nextRoom = NULL;
        temp->s = NULL;
        return temp;
    }
    while(baseRoom->nextRoom!=NULL)
        baseRoom= baseRoom->nextRoom;
    baseRoom->nextRoom = newRoom(baseRoom->nextRoom,roomNo);
    return temp;

}
struct block *newBlock(struct block *fb,char block_name,int
num_beds,int roomNo)
{
    if (fb==NULL)
    {
        struct block *temp =new (nothrow) (struct block);
        temp->block_name = block_name;
        temp->avail_one = 0;
        temp->avail_two = 0;
        temp->avail_three = 0;
        temp->avail_four = 0;
        temp->nextBlock = NULL;
        temp->one=temp->two=temp->three=temp->four=NULL;
        fb = temp;

    }

    if (fb->block_name!=block_name && fb->nextBlock!=NULL)

        newBlock(fb->nextBlock,block_name,num_beds,roomNo);

    else if(fb->block_name!=block_name && fb->nextBlock==NULL)

        fb->nextBlock              =              newBlock(fb-
>nextBlock,block_name,num_beds,roomNo);

    else if(fb->block_name==block_name)

    {

        if(num_beds==1)

        {

            fb->avail_one+=1;

            fb->one = newRoom(fb->one,roomNo);

        }

        else if(num_beds==2)

        {

            fb->avail_two+=1;

            fb->two = newRoom(fb->two,roomNo);

        }

        else if(num_beds==3)

        {

            fb->avail_three+=1;

            fb->three = newRoom(fb->three,roomNo);

        }

        else if(num_beds==4)

        {

            fb->avail_four+=1;

            fb->four = newRoom(fb->four,roomNo);

        }

    }

    return fb;
```

```
}

struct    hostel    *createHostel(struct    hostel    *base,char
block_name,int num_beds,int roomNo)
{
    if(base==NULL)
    {
        struct hostel *temp =new (nothrow) (struct hostel);
        temp-> total_one = 0;
        temp-> total_two = 0;
        temp-> total_three = 0;
        temp-> total_four = 0;
        temp->firstBlock = NULL;
        temp->firstBlock            =            newBlock(temp-
>firstBlock,block_name,num_beds,roomNo);
        base = temp;
    }
    else
        newBlock(base->firstBlock,block_name,num_beds,roomNo);

    if(num_beds==1)
        base-> total_one += 1;
    else if(num_beds==2)
        base-> total_two += 1;
    else if(num_beds==3)
        base-> total_three += 1;
    else if(num_beds==4)
        base-> total_four += 1;

    return base;
```

```
}
struct hostel *readHostel(struct hostel *base)
{
    ifstream infile;
    infile.open("hostel_room.txt");
    int i=0;
    string s;
    while(!infile.eof()) // To get you all the lines.
        {
             getline(infile,s); // Saves the line in STRING.
             if(s.empty())
                continue;
             vector <string> tokens;
             stringstream check1(s);
            string intermediate;
            while(getline(check1, intermediate, ' '))
                tokens.push_back(intermediate);
            if (i!=0)
            {
                char block_name = tokens[0][0];
                int num_beds = std::stoi(tokens[1]);
                float cgpa = std::stoi(tokens[2]);
                if(base==NULL)
                {
                    base                                     =
createHostel(base,block_name,num_beds,cgpa);
                }
                else

createHostel(base,block_name,num_beds,cgpa);
            }
            i++;
        }
    infile.close();
    return base;
}

void hostelStatus(struct hostel *base)
{
    cout<<"Total    One    Beded    Rooms    Avail:    "<<base-
>total_one<<endl;
    cout<<"Total    Two    Beded    Rooms    Avail:    "<<base-
>total_two<<endl;
    cout<<"Total    Three    Beded    Rooms    Avail:    "<<base-
>total_three<<endl;
    cout<<"Total    Four    Beded    Rooms    Avail:    "<<base-
>total_four<<endl;
}
void trieTraversal(struct hostel *base,int numBeds)
{

    struct block *tempBlock = base->firstBlock;
    while(tempBlock!=NULL)
    {

        struct room *tempRoom=NULL;
        if(numBeds==1 && tempBlock->avail_one>0)
        {
            cout<<"In Block    "<<tempBlock->block_name<<endl;
            cout<<"Available One Beded Rooms: ";
            tempRoom = tempBlock->one;
            while(tempRoom->nextRoom!=NULL)
            {
                if(tempRoom->status==0)
                    cout<<tempRoom->room_number<<"  ";
                tempRoom=tempRoom->nextRoom;
            }
            cout<<endl;
        }

        if(numBeds==2 && tempBlock->avail_two>0)
        {
            cout<<"In Block    "<<tempBlock->block_name<<endl;
            cout<<"Available two Beded Rooms: ";
            tempRoom = tempBlock->two;
            while(tempRoom->nextRoom!=NULL)
            {
                if(tempRoom->status==0)
                    cout<<tempRoom->room_number<<"  ";
                tempRoom=tempRoom->nextRoom;
            }
            cout<<endl;
        }

        if(numBeds==3 && tempBlock->three>0)
        {
            cout<<"In Block    "<<tempBlock->block_name<<endl;
            cout<<"Available Three Beded Rooms: ";
            tempRoom = tempBlock->three;
            while(tempRoom->nextRoom!=NULL)
            {
                if(tempRoom->status==0)
                    cout<<tempRoom->room_number<<"  ";
                tempRoom=tempRoom->nextRoom;
            }
            cout<<endl;
        }

        if(numBeds==4 && tempBlock->four>0)
        {
            cout<<"In Block    "<<tempBlock->block_name<<endl;
            cout<<"Available Four Beded Rooms: ";
            tempRoom = tempBlock->four;
            while(tempRoom->nextRoom!=NULL)
            {
                if(tempRoom->status==0)
                    cout<<tempRoom->room_number<<"  ";
                tempRoom=tempRoom->nextRoom;
            }
            cout<<endl;
        }
        tempBlock=tempBlock->nextBlock;
    }
}
void roomBooking(struct hostel* base,struct student* llroot,struct student* root)
{
    int numBeds,roomNo;
    char blockName;
    llroot = nextStudent(llroot);
    string id = llroot->id;
    llroot->isBooked=true;
    hostelStatus(base);
    cout<<"\n"<< llroot-> name <<"  "<<llroot->id<<"\n"<<"  How many bedded rooms you want?:  ";
```

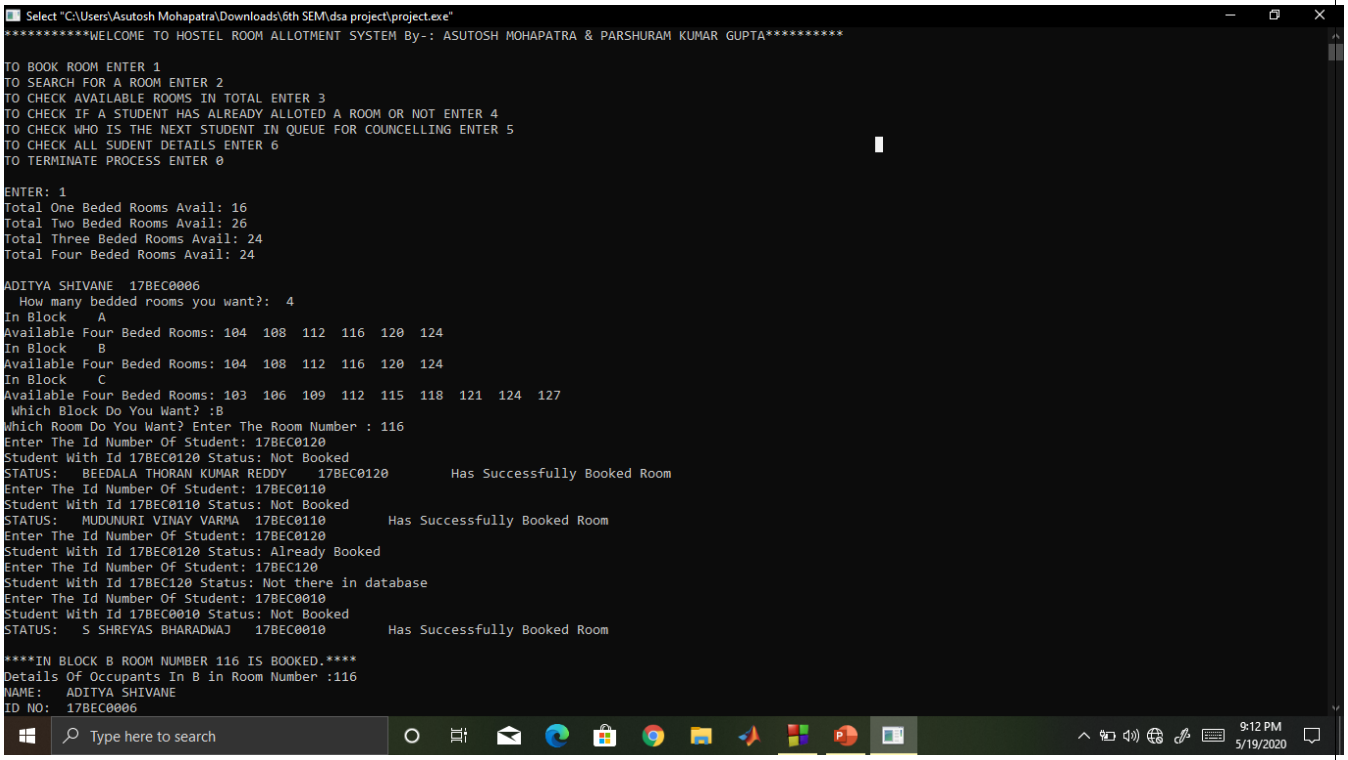
```
cin>>numBeds;
trieTraversal(base,numBeds);
cout<<" Which Block Do You Want? : Enter The Block name: ";
cin >> blockName;
cout<<"Which Room Do You Want? Enter The Room Number : ";
cin>>roomNo;
struct block *tempBlock = base->firstBlock;
struct room *tempRoom=NULL;
while(tempBlock->block_name!=blockName)
    tempBlock=tempBlock->nextBlock;

if(numBeds==1)
{
    base->total_one=base->total_one-1;
    tempRoom = tempBlock->one;
    tempBlock->avail_one = tempBlock->avail_one-1;
    while(tempRoom->room_number!=roomNo)
        tempRoom=tempRoom->nextRoom;
}
else if(numBeds==2)
{
    base->total_two=base->total_two-1;
    tempRoom = tempBlock->two;
    tempBlock->avail_two = tempBlock->avail_two-1;
    while(tempRoom->room_number!=roomNo)
        tempRoom=tempRoom->nextRoom;
}

else if(numBeds==3)
{
    base->total_three=base->total_four-1;
```

```
        tempRoom = tempBlock->three;
        tempBlock->avail_three = tempBlock->avail_three-1;
        while(tempRoom->room_number!=roomNo)
            tempRoom=tempRoom->nextRoom;
    }

    else if(numBeds==4)
    {
        base->total_four=base->total_four-1;
        tempRoom = tempBlock->four;
        tempBlock->avail_four = tempBlock->avail_four-1;
        while(tempRoom->room_number!=roomNo)
            tempRoom=tempRoom->nextRoom;
    }
    tempRoom->status=1;
    tempRoom->s = llroot;
    llroot->currentBlock = blockName;
    llroot->roomNo = roomNo;
    int i=numBeds-1;
    while(i>0)
    {
        struct student* currentNode =NULL;
        currentNode = funcIfPresent(root);
        if(currentNode!=NULL and currentNode->isBooked==false)
        {
            currentNode->isBooked=true;
            llroot->nextRoommate = currentNode;
            llroot = currentNode;
            llroot->currentBlock = blockName;
            llroot->roomNo = roomNo;
            cout<<"STATUS:      "<<currentNode->name<<  "\t"<<
currentNode->id <<"\t Has Successfully Booked Room"<< endl;

            i=i-1;

        }

    }

    cout<<"\n"<<"****IN  BLOCK  >>"<<tempBlock->block_name<<"<<
ROOM   NUMBER   >>"<<   tempRoom->room_number<<"<<   IS   BOOKED
SUCCESSFULLY.****"<<endl;

    struct student *studentDetails = tempRoom->s;

    cout<<"Details  Of  Occupants  In  "<<blockName<<"  in  Room
Number :"<<roomNo<<endl;

    while(studentDetails->nextRoommate!=NULL)

    {

        cout<<"NAME:"<<"\t"<<studentDetails->name<<endl;

        cout<<"ID NO:"<<"\t"<<studentDetails->id<<endl;

        cout<<"CGPA:"<<"\t"<<studentDetails->cgpa<<endl;

        studentDetails=studentDetails->nextRoommate;

    }

    cout<<"NAME:"<<"\t"<<studentDetails->name<<endl;

    cout<<"ID NO:"<<"\t"<<studentDetails->id<<endl;

    cout<<"CGPA:"<<"\t"<<studentDetails->cgpa<<endl<<endl;

}

void trieSearch(struct hostel* base)

{

    char blockName;

    int roomNo,bedCount;

    struct block *tempBlock = base->firstBlock;

    struct room *tempRoom=NULL;

    while(true)

    {

        tempBlock = base->firstBlock;
        cout<<" ENTER THE BLOCK NAME: ";
        cin>>blockName;
        while(           tempBlock!=NULL           &&tempBlock-
    >block_name!=blockName )
            tempBlock=tempBlock->nextBlock;
        if(tempBlock==NULL)
            cout<<"Invalid Choice"<<endl;
        else break;
    }
    while(true)
    {
        cout<<" ENTER THE ROOM NUMBER :";
        cin>>roomNo;
        bool loopbreak=false;
        tempRoom = tempBlock->one;
        while(tempRoom->nextRoom!=NULL)
        {
            if(tempRoom->room_number==roomNo) break;
            tempRoom=tempRoom->nextRoom;
        }
        if(tempRoom->room_number==roomNo) break;
        tempRoom = tempBlock->two;
        while(tempRoom->nextRoom!=NULL)
        {
            if(tempRoom->room_number==roomNo) break;
            tempRoom=tempRoom->nextRoom;
        }
        if(tempRoom->room_number==roomNo) break;

        tempRoom = tempBlock->three;
        while(tempRoom->nextRoom!=NULL)
        {
            if(tempRoom->room_number==roomNo) break;
            tempRoom=tempRoom->nextRoom;
        }
        if(tempRoom->room_number==roomNo) break;
        tempRoom = tempBlock->four;
        while(tempRoom->nextRoom!=NULL)
        {
            if(tempRoom->room_number==roomNo) break;
            tempRoom=tempRoom->nextRoom;
        }
        if(tempRoom->room_number==roomNo) break;
        cout<<"Entered Invalid RoomNo "<<endl;
    }
    if(tempRoom->status==0)
        cout<<"ROOM NOT BOOKED."<<endl<<endl;

    else
    {
        struct student *studentDetails = tempRoom->s;
        cout<<"Details Of Occupants In "<<blockName<<" in Room
Number :"<<roomNo<<endl;
        while(studentDetails->nextRoommate!=NULL)
        {
            cout<<"NAME:"<<"\t"<<studentDetails->name<<endl;
            cout<<"ID NO:"<<"\t"<<studentDetails->id<<endl;
            cout<<"CGPA:"<<"\t"<<studentDetails->cgpa<<endl;
            studentDetails=studentDetails->nextRoommate;
        }
        cout<<"NAME:"<<"\t"<<studentDetails->name<<endl;
        cout<<"ID NO:"<<"\t"<<studentDetails->id<<endl;
        cout<<"CGPA:"<<"\t"<<studentDetails->cgpa<<endl<<endl;
    }
}
struct student *newnode(string id,string name,float cgpa)
{
    struct student *temp =new (nothrow) (struct student);
    temp->id = id;
    temp->cgpa =cgpa;
    temp->name= name;
    temp->right=temp->left =temp->nextll=NULL;
    temp->isBooked=false;
    temp->roomNo = 0;
    temp->currentBlock = 'n';
    //cout<<temp->id<<"\t"<<temp->name<<"\t"<<temp-
>cgpa<<"\n";
    return temp;
}
struct student* inseart(struct student* node,string id,string
name,float cgpa)
{
    if(node==NULL) return newnode(id,name,cgpa);
    if( cgpa <= node->cgpa)
        node->left = inseart(node->left,id,name,cgpa);
    else if(cgpa>node->cgpa)
        node->right =inseart(node->right,id,name,cgpa);
    return node;
}

void inorder (struct student* root,struct student* llroot,struct
student* temp)
{
    if(root!=NULL)
    {
        inorder(root->right,llroot,temp);
        if(temp->nextll!=NULL)
        {
            llroot = temp->nextll;
            temp->nextll = NULL;
        }
        llroot->nextll = root;
        llroot  =  llroot->nextll;
        inorder(root->left,llroot,temp);
        if(temp->nextll==NULL)
            temp->nextll=llroot;
    }
}

bool ifPresent(struct student* node,string id,struct student*
currentNode)
{
    if (node == NULL)
        return false;

    if (node->id== id)
    {
        currentNode->nextll = node;
        return true;
    }

    bool res1 = ifPresent(node->left,id,currentNode);

    if(res1) return true;
    bool res2 = ifPresent(node->right,id,currentNode);

    return res2;
}

struct student *readStudent(struct student* root)
{
    ifstream infile;

    infile.open("student_list.txt");

    int i=0;
    string s;
    while(!infile.eof()) // To get you all the lines.
        {
             getline(infile,s); // Saves the line in STRING.
             if(s.empty())
                continue;
             vector <string> tokens;
             stringstream check1(s);
            string intermediate;
            while(getline(check1, intermediate, ','))
                tokens.push_back(intermediate);



            if (i!=0)
            {
                string id = tokens[0];
                string name = tokens[2];
                float cgpa = std::stof(tokens[1]);
                if(root==NULL)
```

```
                    root = inseart(root,id,name,cgpa);
                else
                    inseart(root,id,name,cgpa);
            }
            i++;
        }
    infile.close();
    return root;

}
struct student* funcIfPresent(struct student* root)
{
    string id;
    cout<<"Enter The Id Number Of Student: ";
    cin>>id;
    struct student* currentNode=new (nothrow) (struct student);
    currentNode->nextll = NULL;
    bool status = ifPresent(root,id,currentNode);

    cout<<"Student With Id "<<id ;
    if(status==true)
    {
        if(currentNode->nextll->isBooked==false)
        {
            cout<<" Status: Not Booked"<<endl;
        }
        else
        {
            cout<<" Status: Already Booked"<<endl;
        }
    }
    else
    {
        cout<<" Status: Not there in database"<<endl;
    }
    return currentNode->nextll;
}
struct student* nextStudent(struct student* llroot)
{
    while(llroot->isBooked==true)
        llroot = llroot->nextll;
    return llroot;
}

void studentDetails(struct student* root)
{
    if(root!=NULL)
    {
        studentDetails(root->right);
        cout<<  "NAME:  \t"<<root->name<<"\t  ID:  \t"<<  root-
>id<<"\t CGPA: \t"<<root->cgpa;
        if(root->isBooked)
            cout<<"\t BLOCK NAME: \t"<<root->currentBlock <<"\t
ROOM NUMBER: \t"<<root->roomNo<< endl;
        else
            cout<<"\t ROOM BOOKING YET TO BE COMPLETED "<<endl;
        studentDetails(root->left);
    }
}
```

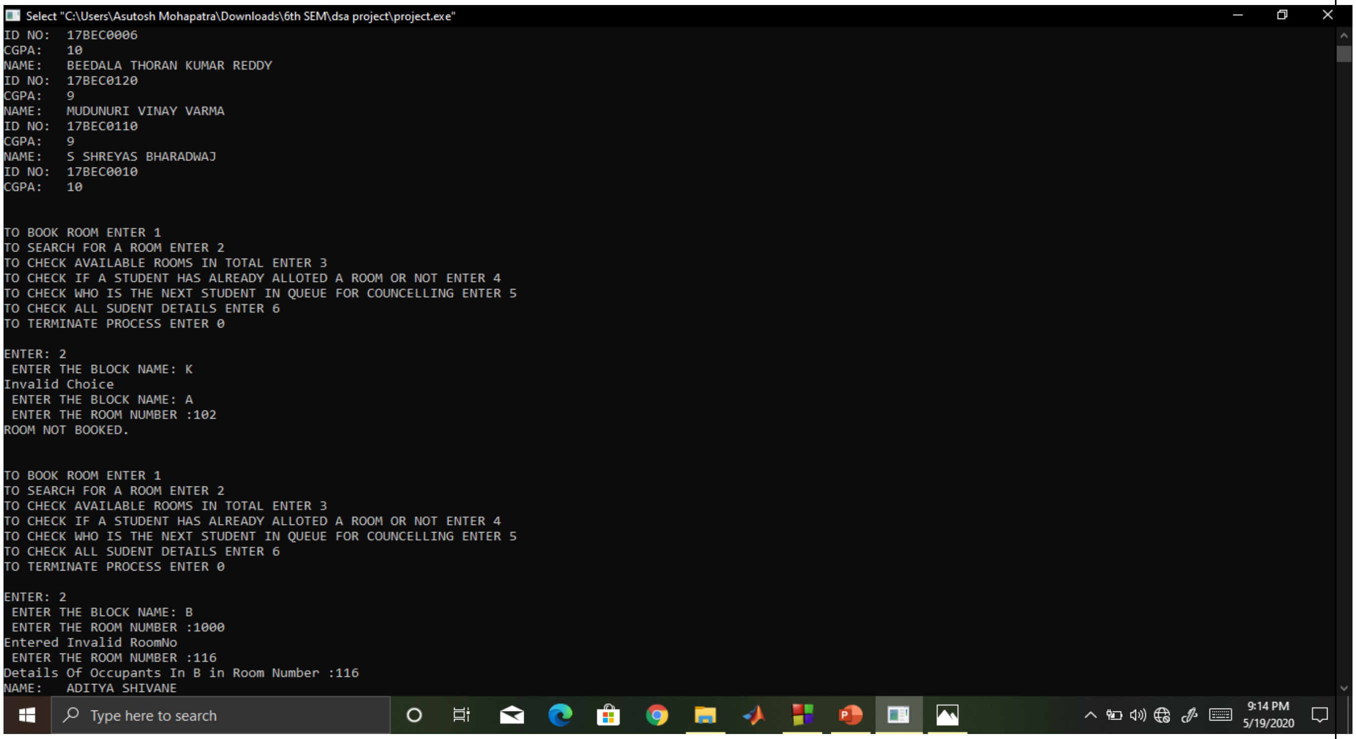
PROGRAM OUTPUT AND OPERATION-:

1) ROOM BOOKING


Select "C:\Users\Asutosh Mohapatra\Downloads\6th SEM\dsa project\project.exe"
Type here to search
9:12 PM
5/19/2020

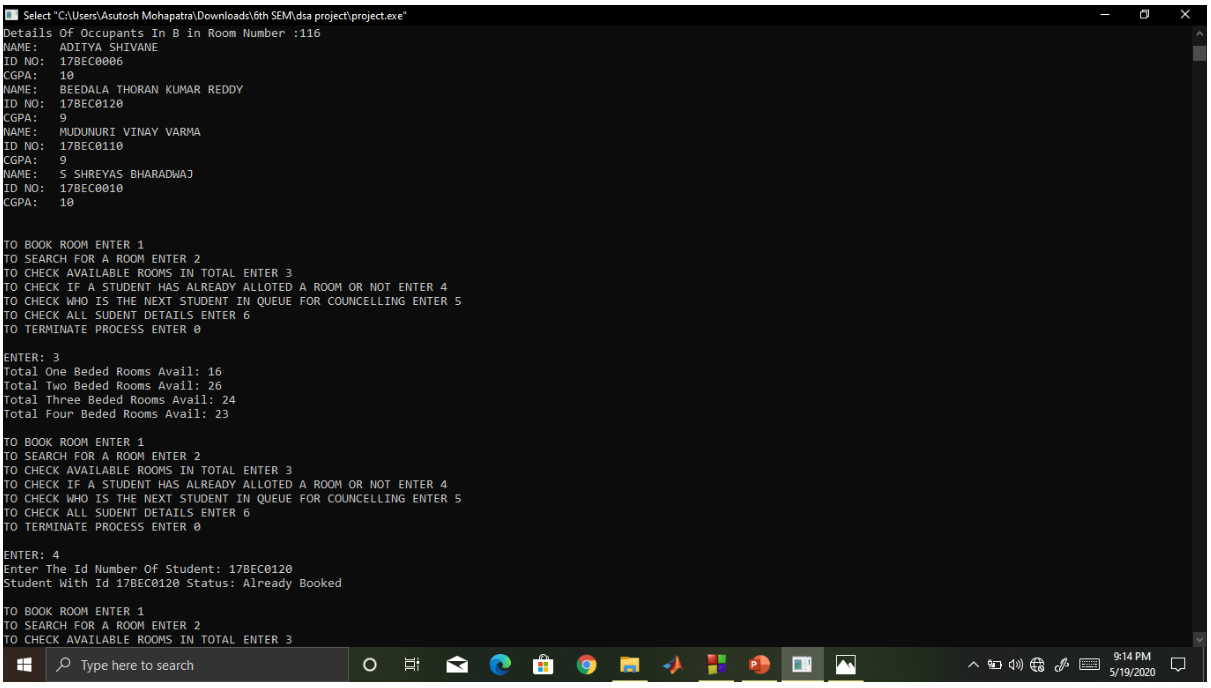
```
***********WELCOME TO HOSTEL ROOM ALLOTMENT SYSTEM By-: ASUTOSH MOHAPATRA & PARSHURAM KUMAR GUPTA**********

TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 1
Total One Beded Rooms Avail: 16
Total Two Beded Rooms Avail: 26
Total Three Beded Rooms Avail: 24
Total Four Beded Rooms Avail: 24

ADITYA SHIVANE  17BEC0006
  How many bedded rooms you want?:  4
In Block    A
Available Four Beded Rooms: 104  108  112  116  120  124
In Block    B
Available Four Beded Rooms: 104  108  112  116  120  124
In Block    C
Available Four Beded Rooms: 103  106  109  112  115  118  121  124  127
 Which Block Do You Want? :B
Which Room Do You Want? Enter The Room Number : 116
Enter The Id Number Of Student: 17BEC0120
Student With Id 17BEC0120 Status: Not Booked
STATUS:   BEEDALA THORAN KUMAR REDDY    17BEC0120        Has Successfully Booked Room
Enter The Id Number Of Student: 17BEC0110
Student With Id 17BEC0110 Status: Not Booked
STATUS:   MUDUNURI VINAY VARMA  17BEC0110        Has Successfully Booked Room
Enter The Id Number Of Student: 17BEC0120
Student With Id 17BEC0120 Status: Already Booked
Enter The Id Number Of Student: 17BEC120
Student With Id 17BEC120 Status: Not there in database
Enter The Id Number Of Student: 17BEC0010
Student With Id 17BEC0010 Status: Not Booked
STATUS:   S SHREYAS BHARADWAJ   17BEC0010        Has Successfully Booked Room

****IN BLOCK B ROOM NUMBER 116 IS BOOKED.****
Details Of Occupants In B in Room Number :116
NAME:   ADITYA SHIVANE
ID NO:  17BEC0006
```

2) SEARCHING A BLOCK


Select "C:\Users\Asutosh Mohapatra\Downloads\6th SEM\dsa project\project.exe"
Type here to search
9:14 PM
5/19/2020

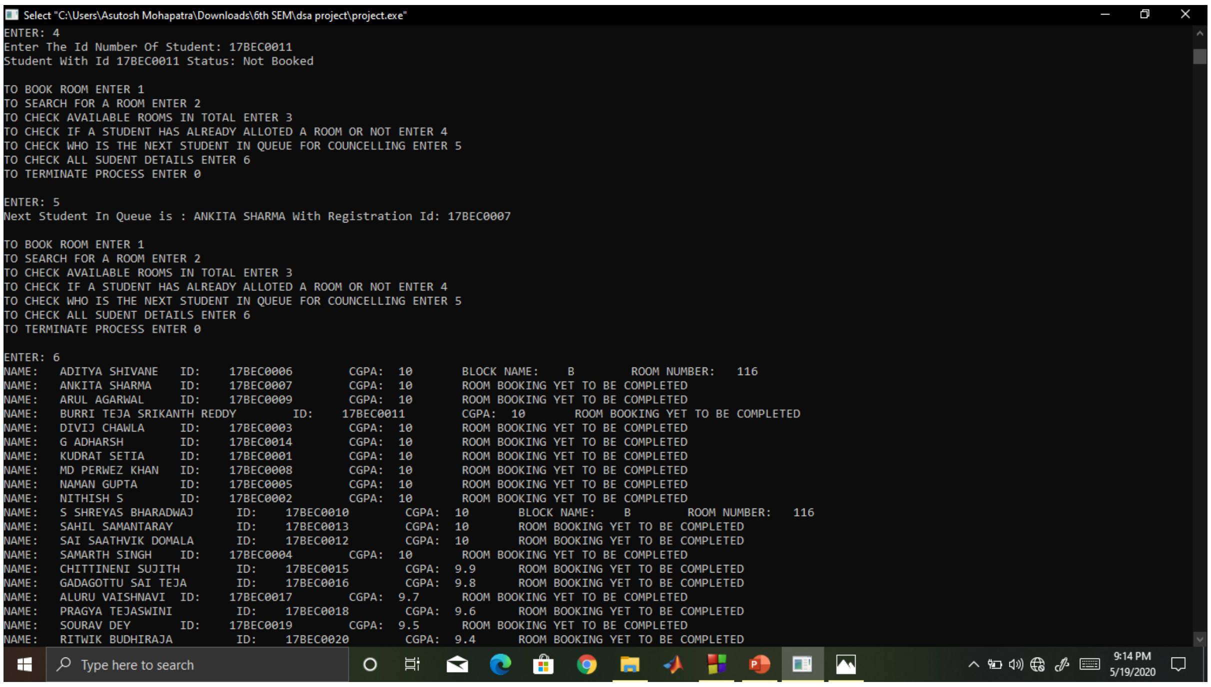
```
ID NO:  17BEC0006
CGPA:   10
NAME:   BEEDALA THORAN KUMAR REDDY
ID NO:  17BEC0120
CGPA:   9
NAME:   MUDUNURI VINAY VARMA
ID NO:  17BEC0110
CGPA:   9
NAME:   S SHREYAS BHARADWAJ
ID NO:  17BEC0010
CGPA:   10


TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 2
 ENTER THE BLOCK NAME: K
Invalid Choice
 ENTER THE BLOCK NAME: A
 ENTER THE ROOM NUMBER :102
ROOM NOT BOOKED.


TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 2
 ENTER THE BLOCK NAME: B
 ENTER THE ROOM NUMBER :1000
Entered Invalid RoomNo
 ENTER THE ROOM NUMBER :116
Details Of Occupants In B in Room Number :116
NAME:   ADITYA SHIVANE
```

3) TOTAL ROOMS AVAILABLE AS PER BEDCOUNT

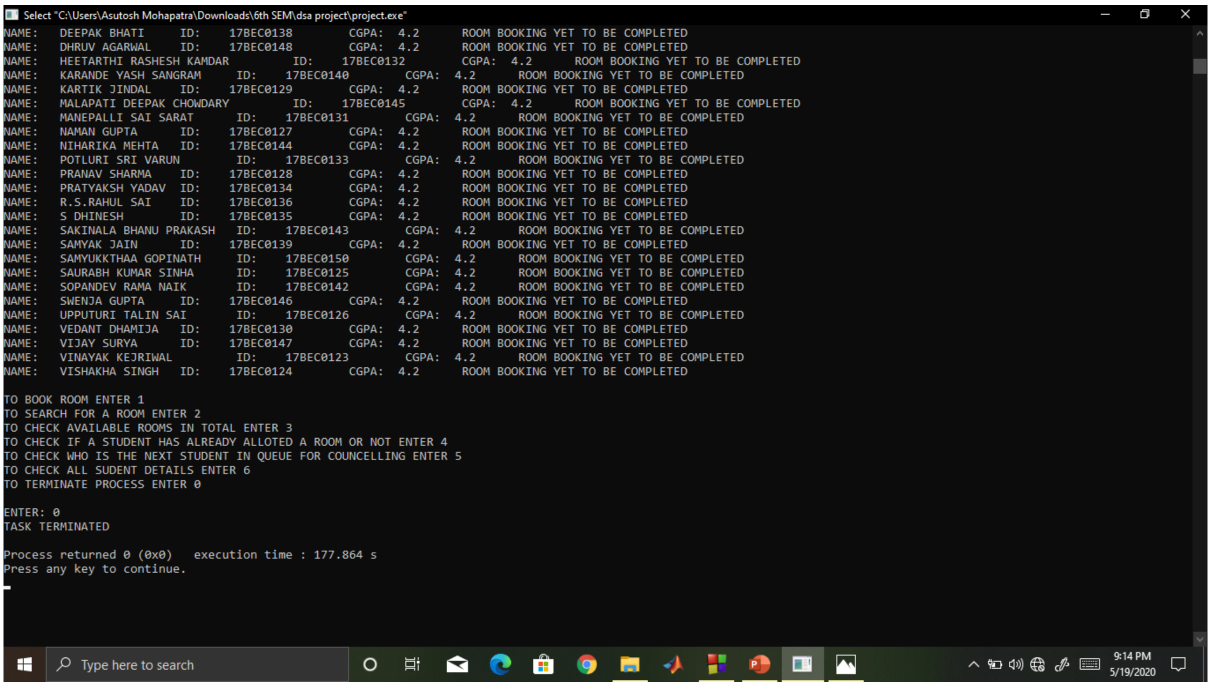


```
Select "C:\Users\Asutosh Mohapatra\Downloads\6th SEM\dsa project\project.exe"
Details Of Occupants In B in Room Number :116
NAME:   ADITYA SHIVANE
ID NO:  17BEC0006
CGPA:   10
NAME:   BEEDALA THORAN KUMAR REDDY
ID NO:  17BEC0120
CGPA:   9
NAME:   MUDUNURI VINAY VARMA
ID NO:  17BEC0110
CGPA:   9
NAME:   S SHREYAS BHARADWAJ
ID NO:  17BEC0010
CGPA:   10


TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 3
Total One Beded Rooms Avail: 16
Total Two Beded Rooms Avail: 26
Total Three Beded Rooms Avail: 24
Total Four Beded Rooms Avail: 23

TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 4
Enter The Id Number Of Student: 17BEC0120
Student With Id 17BEC0120 Status: Already Booked

TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
```

4) NEXT STUDENT FOR ROOM COUNSELLING AND ALL STUDENT DETAILS



```
Select "C:\Users\Asutosh Mohapatra\Downloads\6th SEM\dsa project\project.exe"
ENTER: 4
Enter The Id Number Of Student: 17BEC0011
Student With Id 17BEC0011 Status: Not Booked

TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 5
Next Student In Queue is : ANKITA SHARMA With Registration Id: 17BEC0007

TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 6
NAME:   ADITYA SHIVANE   ID:    17BEC0006        CGPA:  10       BLOCK NAME:    B        ROOM NUMBER:   116
NAME:   ANKITA SHARMA    ID:    17BEC0007        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   ARUL AGARWAL     ID:    17BEC0009        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   BURRI TEJA SRIKANTH REDDY        ID:    17BEC0011        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   DIVIJ CHAWLA     ID:    17BEC0003        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   G ADHARSH        ID:    17BEC0014        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   KUDRAT SETIA     ID:    17BEC0001        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   MD PERWEZ KHAN   ID:    17BEC0008        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   NAMAN GUPTA      ID:    17BEC0005        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   NITHISH S        ID:    17BEC0002        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   S SHREYAS BHARADWAJ      ID:    17BEC0010        CGPA:  10       BLOCK NAME:    B        ROOM NUMBER:   116
NAME:   SAHIL SAMANTARAY         ID:    17BEC0013        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   SAI SAATHVIK DOMALA      ID:    17BEC0012        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   SAMARTH SINGH    ID:    17BEC0004        CGPA:  10       ROOM BOOKING YET TO BE COMPLETED
NAME:   CHITTINENI SUJITH        ID:    17BEC0015        CGPA:  9.9      ROOM BOOKING YET TO BE COMPLETED
NAME:   GADAGOTTU SAI TEJA       ID:    17BEC0016        CGPA:  9.8      ROOM BOOKING YET TO BE COMPLETED
NAME:   ALURU VAISHNAVI  ID:    17BEC0017        CGPA:  9.7      ROOM BOOKING YET TO BE COMPLETED
NAME:   PRAGYA TEJASWINI         ID:    17BEC0018        CGPA:  9.6      ROOM BOOKING YET TO BE COMPLETED
NAME:   SOURAV DEY       ID:    17BEC0019        CGPA:  9.5      ROOM BOOKING YET TO BE COMPLETED
NAME:   RITWIK BUDHIRAJA         ID:    17BEC0020        CGPA:  9.4      ROOM BOOKING YET TO BE COMPLETED
```



```
NAME:   DEEPAK BHATI     ID:    17BEC0138        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   DHRUV AGARWAL    ID:    17BEC0148        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   HEETARTHI RASHESH KAMDAR         ID:    17BEC0132        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   KARANDE YASH SANGRAM     ID:    17BEC0140        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   KARTIK JINDAL    ID:    17BEC0129        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   MALAPATI DEEPAK CHOWDARY         ID:    17BEC0145        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   MANEPALLI SAI SARAT      ID:    17BEC0131        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   NAMAN GUPTA      ID:    17BEC0127        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   NIHARIKA MEHTA   ID:    17BEC0144        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   POTLURI SRI VARUN        ID:    17BEC0133        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   PRANAV SHARMA    ID:    17BEC0128        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   PRATYAKSH YADAV  ID:    17BEC0134        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   R.S.RAHUL SAI    ID:    17BEC0136        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   S DHINESH        ID:    17BEC0135        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   SAKINALA BHANU PRAKASH   ID:    17BEC0143        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   SAMYAK JAIN      ID:    17BEC0139        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   SAMYUKKTHAA GOPINATH     ID:    17BEC0150        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   SAURABH KUMAR SINHA      ID:    17BEC0125        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   SOPANDEV RAMA NAIK       ID:    17BEC0142        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   SWENJA GUPTA     ID:    17BEC0146        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   UPPUTURI TALIN SAI       ID:    17BEC0126        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   VEDANT DHAMIJA   ID:    17BEC0130        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   VIJAY SURYA      ID:    17BEC0147        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   VINAYAK KEJRIWAL         ID:    17BEC0123        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED
NAME:   VISHAKHA SINGH   ID:    17BEC0124        CGPA:  4.2      ROOM BOOKING YET TO BE COMPLETED

TO BOOK ROOM ENTER 1
TO SEARCH FOR A ROOM ENTER 2
TO CHECK AVAILABLE ROOMS IN TOTAL ENTER 3
TO CHECK IF A STUDENT HAS ALREADY ALLOTED A ROOM OR NOT ENTER 4
TO CHECK WHO IS THE NEXT STUDENT IN QUEUE FOR COUNCELLING ENTER 5
TO CHECK ALL SUDENT DETAILS ENTER 6
TO TERMINATE PROCESS ENTER 0

ENTER: 0
TASK TERMINATED

Process returned 0 (0x0)   execution time : 177.864 s
Press any key to continue.
```

Select "C:\Users\Asutosh Mohapatra\Downloads\6th SEM\dsa project\project.exe"
Type here to search
9:14 PM
5/19/2020

CONCLUSION-:

This project had the motive of creating a very efficient and dynamic hostel room booking system that is achieved. The program written is very time efficient and storage efficient too. We have not only provided a system to book a room but we have included so many functions. We have provided both student searching and hostel room searching.

In the future, we can extend this project and include machine learning to it to help students to find a better room. By including machine learning in this project will be more advanced and room booking and searching and finding roommates can be done more efficiently.

REFERENCES-:

1) https://www.geeksforgeeks.org/trie-insert-and-search/
2) https://www.geeksforgeeks.org/priority-queue-set-1-introduction/
3) http://www.fredosaurus.com/notes-cpp/io/readtextfile.html
4) https://www.geeksforgeeks.org/cpp-program-read-file-word-word/
5) Narasimha Karumanchi- Data structures and algorithms made easy.