Hospital Management System

Abstract :

Hospital management requires a lot of decision making which is highly difficult if there is no strong management system in place. Since you need precise and accurate implementation at every stage, the automation system in the hospital has to be self-sufficient. Today, it is not possible to imagine a super-specialty hospital without it. A reliable, cost-effective, and efficient system becomes the backbone of the success of a medical center. There are several benefits of installing full-fledged software.

When your hospital wants to be among the top-preferred and high-rated hospitals by insurance companies, there is no escape from hospital management system. Medicare companies and insurance companies rely on electronic data up to a major extent that is possible only when there is an automated system in place.

It is one of the critical quality indicators. A hospital that is capable of sending and receiving patient information and medical reports electronically always gets a higher preference over others. A good hospital management system stands out your medical center, nursing home, or hospital distinctly from competitors. It adds a value to your reputation in the market.

A good quality management system makes sure that operational and clinical decision-making process is fast, accurate, and efficient. With an easy, single view availability of data points, doctors, and medical support staff gets facilitated.

Data Structures Used :

1) Stacks
2) Queues
3) Linked Lists
4) Trees
5) Linear Search
6) Binary Search
7) Sorting

Why the above Data Structures were being Used :

1) Stack :

A stack is a linear data structure.The elements in a stack are added and removed only from one end, which is called the TOP . Hence, a stack is called a LIFO (Last-In-First-Out) data structure.Every time an element is added, it goes on the top of the stack, the only element that can be removed is the element that was at the top of the stack, just like a pile of objects.

2) Queue :

A queue is a FIFO (First-In, First-Out) data structure in which the element that is inserted first is the first one to be taken out. The elements in a queue are added at one end called the REAR and

removed from the other end called the FRONT.A queue is open at both its ends. One end is always used to insert data (enqueue) and the other is used to remove data (dequeue).

3) Linked List :

A linked list, in simple terms, is a linear collection of data elements. These data elements are called nodes. Linked list is a data structure which in turn can be used to implement other data.A linked list can be perceived as a train or a sequence of nodes in which each node contains one or more data fields and a pointer to the next node.Both arrays and linked lists are a linear collection of data elements. But unlike an array, a linked list does not store its nodes in consecutive memory locations.Another advantage of a linked list over an array is that we can add any number of elements in the list.

4) Tree :

A tree is a data structure which is mainly used to store hierarchical data. A tree is recursively defined as collection of one or more nodes where one node is designated as the root of the tree and the remaining nodes can be partitioned into non-empty sets each of which is a sub-tree of the root.In a binary tree, every node has zero, one, or at the most two successors. A node that has no successors is called a leaf node or a terminal node. Every node other than the root node has a parent.A binary search tree, also known as an ordered binary tree, is a variant of binary tree in which all the nodes in the left sub-tree have a value less than that of the root node and all the nodes in the right sub-tree have a value either equal to or greater than the root node.The average running time of a search operation is O(log 2 n).

5) Linear Search :

Linear search, also called as sequential search, is a very simple method used for searching an array for a particular value. It works by comparing the value to be searched with every element of the array one by one in a sequence until a match is found. Linear search is mostly used to search an unordered list of elements (array in which data elements are not sorted).

6) Binary Search :

Binary search is a searching algorithm that works efficiently with a sorted list.

7) Sorting :

Sorting means arranging the elements of an array so that they are placed in some relevant order which may be either ascending or descending.A sorting algorithm is defined as an algorithm that puts the elements of a list in a certain order, which can be either numerical order, lexicographical order, or any user-defined order. Efficient sorting algorithms are widely used to optimize the use of other algorithms like search and merge algorithms which require sorted lists to work correctly.

Functions Performed :

1) Appointment Registration : It is an application which stores the information of casual pateints.It is implemented by using Queue data structure.It stores the name,full name,age,blood group,gender,phone number,disease of the pateint.It performs enqueue,dequeue,peek,display operations on the Queue.

2) Pateint List : It is an application which stores the in the information of serious disease pateints , who need to be kept in hospital for treatment.It is implemented by using Linked List.It stores the pateint number,name,full name,age,blood group,gender,phone number,disease,allocated doctor's name,date of admission in hospital,treatment type,medicines of the pateint.It performs insertion , deletion , display operations on the linked list.

3) Doctor Availability List : It is an application which stores the in the information of doctors according to their expreince.It is implemented by using Stack data structure.It performs push , pop,peek,display operation on the list.Apart from these functions it also performs the functions linear search,sorting : bubble sort and binary search on the list.

4) Staff Members Details List : It is an application which stores the in the information of Staff Members present in the hospital.It is implemented by using Tree data sructure.It stores the name ,full name,age,gender,phone number,designation,salary,lab number of the staff.It performs the functions insertion,deletion,searching of staff members and also finding the total number of staff members in the hospital.

Program :

```
#include<stdio.h>
#include<stdlib.h>
#include<malloc.h>
#include<string.h>
#define max 10

struct queue
{
        char name[40];
        char fname[40];
        int age;
        char bg[10];
        char gender[10];
        char ph[20];
        char disease[60];
        struct queue *next;
};

struct queue *front = NULL;
struct queue *rear = NULL;

void insert();
void delete();
void peek1();
void display1();

struct node
{
        int pat_no;
        char name[40];
        char fname[40];
```

```
        int age;
        char bg[10];
        char gender[10];
        char ph[10];
        char disease[60];
        char doc_name[40];
        char date[10];
        char treatment[40];
        char med[40];
        struct node *next;
};

struct node *start=NULL;
struct node *create_ll(struct node *);
struct node *display2(struct node *);
struct node *insert_beg(struct node *);
struct node *insert_end(struct node *);
struct node *insert_before(struct node *);
struct node *insert_after(struct node *);
struct node *delete_beg(struct node *);
struct node *delete_end(struct node *);
struct node *delete_node(struct node *);
struct node *delete_after(struct node *);
struct node *sort_list(struct node *);
struct node *delete_list(struct node *);
char st[max][50];
int top=-1;

void push(char st[][50]);
void pop(char st[][50]);
void peek3(char st[][50]);
void display3(char st[][50]);
void bubble_sort(char st[][50],int top);
void linear_search(char st[][50],int top);
void binary_search(char st[][50],int top);

struct imptree
{
        char name[40];
        char fname[40];
        int age;
        char gender[10];
        char ph[50];
        char desig[60];
        char sal[40];
        char lab_no[40];
        struct imptree *left;
        struct imptree *right;
};

struct imptree *tree=NULL;
struct imptree *insert_element(struct imptree *);
```

```
struct imptree *search_element(struct imptree *);
int totalNodes(struct imptree *);
struct imptree *deleteTree(struct imptree *);

void main()
{
	int choice,value;
	int choice1,value1;
	int choice2,value2;
	int choice3,value3;
	int choice4,value4;
	choice=0;
	system("setterm -bold on");
	printf("\n\n\n\t\t\t\t\t\t\t ***** Welcome To Jupiter Hospital *****\n\n\n\n\n");
	system("setterm -bold off");

	while(choice!=5)
	{
		printf("*****Main Menu*****\n");
		printf("1. Appointment Registration\n");
		printf("2. Pateint List\n");
		printf("3. Doctor Availability List\n");
		printf("4. Staff Members Details List\n");
		printf("5. Exit\n");
		printf("Enter your choice: ");
		scanf("%d",&choice);

		switch(choice)
		{

			case 1:

				choice1=0;
				printf("\n\n");
				printf("*****Doctor's Appointment Registration Form*****\n");
				printf("\n\n");

				while(choice1!=5)
				{

					printf("******Main Menu******\n");
					printf("1. Add Pateint in the Queue\n");
					printf("2. Remove Pateint from the Queue\n");
					printf("3. Next Pateint Information\n");
					printf("4. Display all Pateints in the Queue\n");
					printf("5. Exit\n");
					printf("Enter your choice: ");
					scanf("%d",&choice1);

					switch(choice1)
					{
						case 1:
```

```
                        insert();
                        break;
                case 2:
                        delete();
                        break;
                case 3:
                        peek1();
                        break;
                case 4:
                        display1();
                        break;
                case 5:
                        printf("Exiting Application\n");
                        printf("\n");
                        break;
                default:
                        printf("Invalid Choice\n");
                        break;
        }
    }
    break;

case 2:

    choice2=0;
    printf("\n\n");
    printf("*****Pateint List*****\n");
    printf("\n\n");

    while(choice2!=12)
    {
        printf("*****Main Menu*****\n");
        printf("1. Create a Pateint's list\n");
        printf("2. Display the Pateint's list\n");
        printf("3. Add a Pateint at the beginning\n");
        printf("4. Add a Pateint at the end\n");
        printf("5. Add a Pateint before a given Pateint\n");
        printf("6. Add a Pateint after a given Pateint\n");
        printf("7. Delete a Pateint at the beginning\n");
        printf("8. Delete a Pateint at the end\n");
        printf("9. Delete a given Pateint\n");
        printf("10. Delete a Pateint after a given Pateint\n");
        printf("11. Delete the entire Pateint's list\n");
        printf("12. Exit\n");
        printf("Enter your choice ");
        scanf("%d",&choice2);

        switch(choice2)
        {

                case 1:
                        start=create_ll(start);
```

```
                        printf("Linked List Created \n");
                        break;
                case 2:
                        start=display2(start);
                        break;
                case 3:
                        start=insert_beg(start);
                        break;
                case 4:
                        start=insert_end(start);
                        break;
                case 5:
                        start=insert_before(start);
                        break;
                case 6:
                        start=insert_after(start);
                        break;
                case 7:
                        start=delete_beg(start);
                        break;
                case 8:
                        start=delete_end(start);
                        break;
                case 9:
                        start=delete_node(start);
                        break;
                case 10:
                        start=delete_after(start);
                        break;
                case 11:
                        start=delete_list(start);
                        break;
                case 12:
                        printf("Exiting Application\n");
                        printf("\n");
                        break;
                default:
                        printf("Invalid Choice\n");
                        break;
            }
        }
        break;

case 3:

        choice3=0;
        printf("\n\n");
        printf("*****Doctor Availablity List*****\n");
        printf("\n\n");

        while(choice3!=8)
        {
```

```
                printf("*****Main Menu*****\n");
                printf("1. Enter a Doctor in Doctor Availablity List\n");
                printf("2. Appoint a Doctor from Doctor Avalibility List \n");
                printf("3. Next Doctor available for appointment\n");
                printf("4. Display all Doctors available\n");
                printf("5. Sort the Doctor Availability List\n");
                printf("6. Search the Doctor's name using Linear Search\n");
                printf("7. Search the Doctor's name using Binary Search\n");
                printf("8. Exit\n");
                printf("Enter your option ");
                scanf("%d",&choice3);

                switch(choice3)
                {
                        case 1:
                                push(st);
                                break;
                        case 2:
                                pop(st);
                                break;
                        case 3:
                                peek3(st);
                                break;
                        case 4:
                                display3(st);
                                break;
                        case 5:
                                bubble_sort(st,top);
                                break;
                        case 6:
                                linear_search(st,top);
                                break;
                        case 7:
                                binary_search(st,top);
                                break;
                        case 8:
                                printf("The program is terminated\n");
                                printf("\n");
                                break;
                        default :
                                printf("You have entered an invalid option\n");
                                break;
                }
        }
        break;

case 4:

        choice4=0;
        printf("\n\n");
        printf("*****Staff Member Management Portal*****\n");
        printf("\n\n");
```

```
                        while(choice4!=5)
                        {
                                printf("*****Main Menu*****\n");
                                printf("1. Insert Staff Member\n");
                                printf("2. Search Staff Member\n");
                                printf("3. Find Total Staff Members in the Hospital\n");
                                printf("4. Delete all Staff Member's Records\n");
                                printf("5. Exit\n");
                                printf("Enter your choice ");
                                scanf("%d",&choice4);
                                switch(choice4)
                                {
                                        case 1:
                                                tree=insert_element(tree);
                                                break;
                                        case 2:
                                                tree=search_element(tree);
                                                break;
                                        case 3:
                                                printf("Total Staff Members in the Hospital is
%d\n",totalNodes(tree));
                                                break;
                                        case 4:
                                                tree=deleteTree(tree);
                                                break;
                                        case 5:
                                                printf("Exiting Application\n");
                                                printf("\n");
                                                break;
                                        default:
                                                printf("Invalid Choice\n");
                                                break;
                                }
                        }
                        break;

                case 5:

                        printf("Exiting Application\n");
                        break;

                default:

                        printf("Invalid Choice\n");
                        break;
        }
    }
}

void insert(int value)
{
    struct queue *new_node;
```

```
	new_node = (struct queue*)malloc(sizeof(struct queue));
	printf("\n");
	while(getchar() != '\n');
	printf("Enter the data of the pateint :\n");
	printf("Enter the Pateint's Name ");
	gets(new_node->name);
	printf("Enter the Pateint's Full Name ");
	gets(new_node->fname);
	printf("Enter the Pateint's Age ");
	scanf("%d",&new_node->age);
	while(getchar() != '\n');
	printf("Enter the Pateint's Blood Group ");
	gets(new_node->bg);
	printf("Enter the Pateint's Gender ");
	gets(new_node->gender);
	printf("Enter the Pateint's Phone Number ");
	gets(new_node->ph);
	printf("Enter the Pateint's Disease ");
	gets(new_node->disease);
	new_node->next = NULL;
	if(front == NULL)
	{
		front=new_node;
		rear=new_node;
	}
	else
	{
		rear->next=new_node;
		rear=new_node;
	}
	printf("\n");
	printf("Success!!! Your Appointment is fixed with the Doctor.\n");
	printf("You will soon receive a message regarding the same conveying the time of
appoinment.\n");
	printf("Thanks for choosing Jupiter Hospital.\n");
	printf("\n");
}

void delete()
{
	printf("\n");
	if(front==NULL)
	{
		printf("Pateint Queue is Empty!!!\n");
	}
	else
	{
		struct queue *temp=front;
		front = front->next;
		printf("The Next Pateint to be examined is\n");
		printf("\n");
		printf("Pateint's Name : %s\n",temp->name);
```

```
                printf("Pateint's Full Name : %s\n",temp->fname);
                printf("Pateint's Age : %d\n",temp->age);
                printf("Pateint's Blood Group : %s\n",temp->bg);
                printf("Pateint's Gender : %s\n",temp->gender);
                printf("Pateint's Phone Number : %s\n",temp->ph);
                printf("Pateint's Disease : %s\n",temp->disease);
                free(temp);
        }
        printf("\n");
}

void peek1()
{
        printf("\n");
        struct queue *temp=front;
        if(front==NULL)
        {
                printf("Pateint Queue is empty\n");
        }
        else
        {
                printf("The Next Pateint in the Queue is\n");
                printf("\n");
                printf("Pateint's Name : %s\n",temp->name);
                printf("Pateint's Full Name : %s\n",temp->fname);
                printf("Pateint's Age : %d\n",temp->age);
                printf("Pateint's Blood Group : %s\n",temp->bg);
                printf("Pateint's Gender : %s\n",temp->gender);
                printf("Pateint's Phone Number : %s\n",temp->ph);
                printf("Pateint's Disease : %s\n",temp->disease);
        }
        printf("\n");
}

void display1()
{
        printf("\n");
        if(front == NULL)
        {
                printf("\nPateint Queue is Empty!!!");
        }
        else
        {
                struct queue *temp=front;
                while(temp!= NULL)
                {
                        printf("Pateint's Name : %s\n",temp->name);
                        printf("Pateint's Full Name : %s\n",temp->fname);
                        printf("Pateint's Age : %d\n",temp->age);
                        printf("Pateint's Blood Group : %s\n",temp->bg);
                        printf("Pateint's Gender : %s\n",temp->gender);
                        printf("Pateint's Phone Number : %s\n",temp->ph);
```

```
                printf("Pateint's Disease : %s\n",temp->disease);
                printf("\n");
                temp = temp->next;
            }
        }
        printf("\n");
}

struct node *create_ll(struct node *start)
{
        struct node *new_node,*ptr;
        int num;
        char a[100],value;
        printf("Enter -1 to end\n");
        printf("Enter the Pateint Number : ");
        scanf("%d",&num);
        while(num!=-1)
        {
            new_node=(struct node *)malloc(sizeof(struct node));
            new_node->pat_no=num;
            while(getchar() != '\n');
            printf("Enter the data of the pateint :\n");
            printf("Enter the Pateint's Name ");
            gets(new_node->name);
            printf("Enter the Pateint's Full Name ");
            gets(new_node->fname);
            printf("Enter the Pateint's Age ");
            scanf("%d",&new_node->age);
            while(getchar() != '\n');
            printf("Enter the Pateint's Blood Group ");
            gets(new_node->bg);
            printf("Enter the Pateint's Gender ");
            gets(new_node->gender);
            printf("Enter the Pateint's Phone Number ");
            gets(new_node->ph);
            printf("Enter the Pateint's Disease ");
            gets(new_node->disease);
            printf("Enter the Pateint's allocated Doctor's Name ");
            gets(new_node->doc_name);
            printf("Enter the Pateint's date of admission ");
            gets(new_node->date);
            printf("Enter the Pateint's treatment type ");
            gets(new_node->treatment);
            printf("Enter the Pateint's medicines ");
            gets(new_node->med);
            if(start==NULL)
            {
                new_node->next=NULL;
                start=new_node;
            }
            else
            {
```

```
                ptr=start;
                while(ptr->next!=NULL)
                {
                        ptr=ptr->next;
                }
            ptr->next=new_node;
            new_node->next=NULL;
            }

            printf("Enter the Pateint Number : ");
            scanf("%d",&num);
        }
        return start;
}

struct node *display2(struct node *start)
{
        struct node *ptr;
        ptr=start;
        while(ptr!=NULL)
        {
            printf("Pateint's Number : %d\n",ptr->pat_no);
            printf("Pateint's Name : %s\n",ptr->name);
            printf("Pateint's Full Name : %s\n",ptr->fname);
            printf("Pateint's Age : %d\n",ptr->age);
            printf("Pateint's Blood Group : %s\n",ptr->bg);
            printf("Pateint's Gender : %s\n",ptr->gender);
            printf("Pateint's Phone Number : %s\n",ptr->ph);
            printf("Pateint's Disease : %s\n",ptr->disease);
            printf("Pateint's allocated Doctor's Name : %s\n",ptr->doc_name);
            printf("Pateint's date of admission : %s\n",ptr->date);
            printf("Pateint's treatment type : %s\n",ptr->treatment);
            printf("Pateint's medicines : %s\n",ptr->med);
            printf("\n");
            ptr=ptr->next;
        }
        return start;
}

struct node *insert_beg(struct node *start)
{
        struct node *new_node;
        int num;
        printf("Enter the Pateint Number : ");
        scanf("%d",&num);
        new_node=(struct node *)malloc(sizeof(struct node));
        new_node->pat_no=num;
        while(getchar() != '\n');
        printf("Enter the data of the pateint :\n");
        printf("Enter the Pateint's Name ");
        gets(new_node->name);
        printf("Enter the Pateint's Full Name ");
```

```
        gets(new_node->fname);
        printf("Enter the Pateint's Age ");
        scanf("%d",&new_node->age);
        while(getchar() != '\n');
        printf("Enter the Pateint's Blood Group ");
        gets(new_node->bg);
        printf("Enter the Pateint's Gender ");
        gets(new_node->gender);
        printf("Enter the Pateint's Phone Number ");
        gets(new_node->ph);
        printf("Enter the Pateint's Disease ");
        gets(new_node->disease);
        printf("Enter the Pateint's allocated Doctor's Name ");
        gets(new_node->doc_name);
        printf("Enter the Pateint's date of admission ");
        gets(new_node->date);
        printf("Enter the Pateint's treatment type ");
        gets(new_node->treatment);
        printf("Enter the Pateint's medicines ");
        gets(new_node->med);
        new_node->next=start;
        start=new_node;
        return start;
}

struct node *insert_end(struct node *start)
{
        struct node *ptr,*new_node;
        int num;
        printf("Enter the Pateint Number : ");
        scanf("%d",&num);
        new_node=(struct node *)malloc(sizeof(struct node));
        new_node->pat_no=num;
        while(getchar() != '\n');
        printf("Enter the data of the pateint :\n");
        printf("Enter the Pateint's Name ");
        gets(new_node->name);
        printf("Enter the Pateint's Full Name ");
        gets(new_node->fname);
        printf("Enter the Pateint's Age ");
        scanf("%d",&new_node->age);
        while(getchar() != '\n');
        printf("Enter the Pateint's Blood Group ");
        gets(new_node->bg);
        printf("Enter the Pateint's Gender ");
        gets(new_node->gender);
        printf("Enter the Pateint's Phone Number ");
        gets(new_node->ph);
        printf("Enter the Pateint's Disease ");
        gets(new_node->disease);
        printf("Enter the Pateint's allocated Doctor's Name ");
        gets(new_node->doc_name);
```

```
        printf("Enter the Pateint's date of admission ");
        gets(new_node->date);
        printf("Enter the Pateint's treatment type ");
        gets(new_node->treatment);
        printf("Enter the Pateint's medicines ");
        gets(new_node->med);
        new_node->next=NULL;
        ptr=start;
        while(ptr->next!=NULL)
        {
                ptr=ptr->next;
        }
        ptr->next=new_node;
        return start;
}

struct node *insert_before(struct node *start)
{
        struct node *new_node,*ptr,*preptr;
        int num,val;
        printf("Enter the Pateint Number : ");
        scanf("%d",&num);
        printf("Enter the Pateint Number before which the new Pateint has to be added : ");
        scanf("%d",&val);
        new_node=(struct node *)malloc(sizeof(struct node));
        new_node->pat_no=num;
        while(getchar() != '\n');
        printf("Enter the data of the pateint :\n");
        printf("Enter the Pateint's Name ");
        gets(new_node->name);
        printf("Enter the Pateint's Full Name ");
        gets(new_node->fname);
        printf("Enter the Pateint's Age ");
        scanf("%d",&new_node->age);
        while(getchar() != '\n');
        printf("Enter the Pateint's Blood Group ");
        gets(new_node->bg);
        printf("Enter the Pateint's Gender ");
        gets(new_node->gender);
        printf("Enter the Pateint's Phone Number ");
        gets(new_node->ph);
        printf("Enter the Pateint's Disease ");
        gets(new_node->disease);
        printf("Enter the Pateint's allocated Doctor's Name ");
        gets(new_node->doc_name);
        printf("Enter the Pateint's date of admission ");
        gets(new_node->date);
        printf("Enter the Pateint's treatment type ");
        gets(new_node->treatment);
        printf("Enter the Pateint's medicines ");
        ptr=start;
        while(ptr->pat_no!=val)
```

```
	{
		preptr=ptr;
		ptr=ptr->next;
	}
	preptr->next=new_node;
	new_node->next=ptr;
	return start;
}

struct node *insert_after(struct node *start)
{
	struct node *new_node,*ptr,*preptr;
	int num,value;
	printf("Enter the Pateint Number : ");
	scanf("%d",&num);
	printf("Enter the Pateint Number after which the new Pateint has to be inserted : ");
	scanf("%d",&value);
	new_node=(struct node *)malloc(sizeof(struct node));
	new_node->pat_no=num;
	while(getchar() != '\n');
	printf("Enter the data of the pateint :\n");
	printf("Enter the Pateint's Name ");
	gets(new_node->name);
	printf("Enter the Pateint's Full Name ");
	gets(new_node->fname);
	printf("Enter the Pateint's Age ");
	scanf("%d",&new_node->age);
	while(getchar() != '\n');
	printf("Enter the Pateint's Blood Group ");
	gets(new_node->bg);
	printf("Enter the Pateint's Gender ");
	gets(new_node->gender);
	printf("Enter the Pateint's Phone Number ");
	gets(new_node->ph);
	printf("Enter the Pateint's Disease ");
	gets(new_node->disease);
	printf("Enter the Pateint's allocated Doctor's Name ");
	gets(new_node->doc_name);
	printf("Enter the Pateint's date of admission ");
	gets(new_node->date);
	printf("Enter the Pateint's treatment type ");
	gets(new_node->treatment);
	printf("Enter the Pateint's medicines ");
	gets(new_node->med);
	ptr=start;
	preptr=ptr;
	while(preptr->pat_no!=value)
	{
		preptr=ptr;
		ptr=ptr->next;
	}
	preptr->next=new_node;
```

```
	new_node->next=ptr;
	return start;
}

struct node *delete_beg(struct node *start)
{
	struct node *ptr;
	ptr=start;
	start=start->next;
	free(ptr);
	return start;
}

struct node *delete_end(struct node *start)
{
	struct node *ptr,*preptr;
	ptr=start;
	while(ptr->next!=NULL)
	{
		preptr=ptr;
		ptr=ptr->next;
	}
	preptr->next=NULL;
	free(ptr);
	return start;
}

struct node *delete_node(struct node *start)
{
	struct node *ptr,*preptr;
	int value;
	printf("Enter the Pateint Number which has to be deleted ");
	scanf("%d",&value);
	ptr=start;
	if(ptr->pat_no==value)
	{
		start=delete_beg(start);
		return start;
	}
	else
	{
		while(ptr->pat_no!=value)
		{
			preptr=ptr;
			ptr=ptr->next;
		}
		preptr->next=ptr->next;
		free(ptr);
		return start;
	}
}
```

```
struct node *delete_after(struct node *start)
{
	struct node *ptr,*preptr;
	int value;
	printf("Enter the Pateint Number after which the Pateint has to be deleted : ");
	scanf("%d",&value);
	ptr=start;
	preptr=ptr;
	while(preptr->pat_no!=value)
	{
		preptr=ptr;
		ptr=ptr->next;
	}
	preptr->next=ptr->next;
	free(ptr);
	return start;
}

struct node *delete_list(struct node *start)
{
	struct node *ptr;
	if(start!=NULL)
	{
		ptr=start;
		while(ptr!=NULL)
		{
			printf("%s is the Pateint Name to be deleted next\n",ptr->name);
			start=delete_beg(ptr);
			ptr=start;
		}
	}
	return start;
}

void push(char st[][50])
{
	if(top==max-1)
	{
		printf("Doctor Availablity List Stack Overflow\n");
		printf("The Doctor's Name is not stored\n");
	}
	else
	{
		top=top+1;
		printf("Enter the Name of Available Doctor : ");
		scanf("%s",st[top]);
	}
}

void pop(char st[][50])
{
	if(top==-1)
```

```
	{
		printf("Doctor Availablity List Stack Underflow\n");
	}
	else
	{
		printf("The Doctor appointed is : %s\n",st[top]);
		top=top-1;
	}
}

void peek3(char st[][50])
{
	if(top==-1)
	{
		printf("Doctor Availablity List Stack is empty\n");
	}
	else
	{
		printf("The next Doctor to be appointed is : %s\n",st[top]);
	}
}

void display3(char st[][50])
{
	int i;
	if(top==-1)
	{
		printf("Doctor Availablity List Stack is empty\n");
	}
	else
	{
		printf("The available Doctors are :\n");
		for(i=top;i>=0;i--)
		{
			printf("%s\n",st[i]);
		}
	}
}

void bubble_sort(char st[][50],int top)
{
	int i,j,n;
	char sorted[50][50];
	char temp[50][50];
	n=top;
	for(i=0;i<=n;i++)
	{
		strcpy(sorted[i],st[i]);
	}
	for(i=0;i<=n;i++)
	{
		for(j=0;j<=n-1-i;j++)
```

```
		{
			if(strcmp(sorted[j],sorted[j+1])>0)
			{
				strcpy(temp[0],sorted[j]);
				strcpy(sorted[j],sorted[j+1]);
				strcpy(sorted[j+1],temp[0]);
			}
		}
	}
	printf("The Doctor Availability List in sorted order is\n");
	for(i=0;i<=n;i++)
	{
		printf("%s\n",sorted[i]);
	}
}

void linear_search(char st[][50],int top)
{
	int n,found,i;
	char target[50][50];
	n=top;
	found=0;
	printf("Enter the name of the Doctor to be searched ");
	scanf("%s",target[0]);
	for(i=0;i<=n;i++)
	{
		if(strcmp(st[i],target[0])==0)
		{
			found=1;
			break;
		}
	}
	if(found==1)
	{
		printf("The Doctor %s's name is found at location a[%d]\n",target[0],i);
	}
	else
	{
		printf("The Doctor %s's name is not found in the list\n",target[0]);
	}
}

void binary_search(char st[][50],int top)
{
	int i,j,n,beg,end,mid,found;
	found=0;
	beg=0;
	end=top;
	char target[50][50];
	char sorted[50][50];
	char temp[50][50];
	n=top;
```

```
	for(i=0;i<=n;i++)
	{
		strcpy(sorted[i],st[i]);
	}
	for(i=0;i<=n;i++)
	{
		for(j=0;j<=n-1-i;j++)
		{
			if(strcmp(sorted[j],sorted[j+1])>0)
			{
				strcpy(temp[0],sorted[j]);
				strcpy(sorted[j],sorted[j+1]);
				strcpy(sorted[j+1],temp[0]);
			}
		}
	}
	printf("Enter the name of the Doctor to be searched ");
	scanf("%s",target[0]);
	while(beg<=end)
	{
		mid=(beg+end)/2;
		if(strcmp(sorted[mid],target[0])==0)
		{
			printf("Doctor %s's name is present at the array position a[%d]\
n",target[0],mid);
			found=1;
			break;
		}
		else if(strcmp(sorted[mid],target[0])>0)
		{
			end=mid-1;
		}
		else
		{
			beg=mid+1;
		}
	}
	if(beg>end || found==0)
	{
		printf("Doctor %s's name does not exist in the List\n",target[0]);
	}
}

struct imptree *insert_element(struct imptree *tree)
{
	int value;
	struct imptree *ptr,*nodeptr,*parentptr;
	ptr=(struct imptree *)malloc(sizeof(struct imptree));
	while(getchar() != '\n');
	printf("Enter the details of the Staff Members :\n");
	printf("Enter the Staff's Name ");
	gets(ptr->name);
```

```
        printf("Enter the Staff's Full Name ");
        gets(ptr->fname);
        printf("Enter the Staff's Age ");
        scanf("%d",&ptr->age);
        while(getchar() != '\n');
        printf("Enter the Staff's Gender ");
        gets(ptr->gender);
        printf("Enter the Staff's Phone Number ");
        gets(ptr->ph);
        printf("Enter the Staff's Designation ");
        gets(ptr->desig);
        printf("Enter the Staff's Salary ");
        gets(ptr->sal);
        printf("Enter the Staff's allocated Lab's Number ");
        gets(ptr->lab_no);
        ptr->left=NULL;
        ptr->right=NULL;
        if(tree==NULL)
        {
                tree=ptr;
                tree->left=NULL;
                tree->right=NULL;
        }
        else
        {
                parentptr=NULL;
                nodeptr=tree;
                while(nodeptr!=NULL)
                {
                        parentptr=nodeptr;
                        if(strcmp(nodeptr->name,ptr->name)>0)
                        {
                                nodeptr=nodeptr->left;
                        }
                        else
                        {
                                nodeptr=nodeptr->right;
                        }
                }
                if(strcmp(parentptr->name,ptr->name)>0)
                {
                        parentptr->left=ptr;
                }
                else
                {
                        parentptr->right=ptr;
                }
        }
        return tree;
}

struct imptree *search_element(struct imptree *tree)
```

```
{
	struct imptree *ptr,*nodeptr,*parentptr;
	char target[50][50];
	parentptr=NULL;
	nodeptr=tree;
	printf("Please Enter the Staff Member to be searched ");
	scanf("%s",target[0]);
	if(tree==NULL)
	{
		printf("The Staff Member List is Empty\n");
		return tree;
	}
	while(strcmp(nodeptr->name,target[0])!=0)
	{
		parentptr=nodeptr;
		if(strcmp(nodeptr->name,target[0])>0)
		{
			nodeptr=nodeptr->left;
		}
		else
		{
			nodeptr=nodeptr->right;
		}
	}
	printf("\n");
	if(strcmp(nodeptr->name,target[0])==0)
	{
		printf("Staff's Name : %s\n",nodeptr->name);
		printf("Staff's Full Name : %s\n",nodeptr->fname);
		printf("Staff's Age : %d\n",nodeptr->age);
		printf("Staff's Gender : %s\n",nodeptr->gender);
		printf("Staff's Phone Number : %s\n",nodeptr->ph);
		printf("Staff's Designation : %s\n",nodeptr->desig);
		printf("Staff's Salary : %s\n",nodeptr->sal);
		printf("Staff's allocated Lab Number : %s\n",nodeptr->lab_no);
	}
	else
	{
		printf("The Staff Mamber Name %s is not found\n",target[0]);
	}
	return tree;
}

int totalNodes(struct imptree *tree)
{
	if(tree==NULL)
	{
		return 0;
	}
	else
	{
		return (totalNodes(tree->left)+totalNodes(tree->right)+1);
```

```
        }
}

struct imptree *deleteTree(struct imptree *tree)
{
        if(tree!=NULL)
        {
                deleteTree(tree->left);
                deleteTree(tree->right);
                free(tree);
        }
        tree=NULL;
        return tree;
}
```

Output :

```
nirish@Nirish:~$ cd Desktop
nirish@Nirish:~/Desktop$ gcc HospitalManagement.c
nirish@Nirish:~/Desktop$ ./a.out

                                ***** Welcome To Jupiter Hospital *****


*****Main Menu*****
1. Appointment Registration
2. Pateint List
3. Doctor Availability List
4. Staff Members Details List
5. Exit
Enter your choice: 1


*****Doctor's Appointment Registration Form*****


******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue
3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 1

Enter the data of the pateint :
Enter the Pateint's Name Mahesh
Enter the Pateint's Full Name Mahesh Kumar
Enter the Pateint's Age 31
Enter the Pateint's Blood Group A+
Enter the Pateint's Gender Male
```

Enter the Pateint's Phone Number 9819399454
Enter the Pateint's Disease Cold

Success!!! Your Appointment is fixed with the Doctor.
You will soon receive a message regarding the same conveying the time of appoinment.
Thanks for choosing Jupiter Hospital.

******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue
3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 1

Enter the data of the pateint :
Enter the Pateint's Name Rohan
Enter the Pateint's Full Name Rohan Singh
Enter the Pateint's Age 42
Enter the Pateint's Blood Group B+
Enter the Pateint's Gender Male
Enter the Pateint's Phone Number 982565765
Enter the Pateint's Disease Joint Pain

Success!!! Your Appointment is fixed with the Doctor.
You will soon receive a message regarding the same conveying the time of appoinment.
Thanks for choosing Jupiter Hospital.

******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue
3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 1

Enter the data of the pateint :
Enter the Pateint's Name Ramesh
Enter the Pateint's Full Name Ramesh Patel
Enter the Pateint's Age 27
Enter the Pateint's Blood Group O+
Enter the Pateint's Gender Male
Enter the Pateint's Phone Number 9867543214
Enter the Pateint's Disease Flu

Success!!! Your Appointment is fixed with the Doctor.
You will soon receive a message regarding the same conveying the time of appoinment.
Thanks for choosing Jupiter Hospital.

******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue

3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 1

Enter the data of the pateint :
Enter the Pateint's Name Raj
Enter the Pateint's Full Name Raj Samant
Enter the Pateint's Age 62
Enter the Pateint's Blood Group O-
Enter the Pateint's Gender Male
Enter the Pateint's Phone Number 9845386453
Enter the Pateint's Disease Cold

Success!!! Your Appointment is fixed with the Doctor.
You will soon receive a message regarding the same conveying the time of appoinment.
Thanks for choosing Jupiter Hospital.

******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue
3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 2

The Next Pateint to be examined is

Pateint's Name : Mahesh
Pateint's Full Name : Mahesh Kumar
Pateint's Age : 31
Pateint's Blood Group : A+
Pateint's Gender : Male
Pateint's Phone Number : 9819399454
Pateint's Disease : Cold

******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue
3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 3

The Next Pateint in the Queue is

Pateint's Name : Rohan
Pateint's Full Name : Rohan Singh
Pateint's Age : 42
Pateint's Blood Group : B+
Pateint's Gender : Male
Pateint's Phone Number : 982565765

Pateint's Disease : Joint Pain

******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue
3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 4

Pateint's Name : Rohan
Pateint's Full Name : Rohan Singh
Pateint's Age : 42
Pateint's Blood Group : B+
Pateint's Gender : Male
Pateint's Phone Number : 982565765
Pateint's Disease : Joint Painj

Pateint's Name : Ramesh
Pateint's Full Name : Ramesh Patel
Pateint's Age : 27
Pateint's Blood Group : O+
Pateint's Gender : Male
Pateint's Phone Number : 9867543214
Pateint's Disease : Flu

Pateint's Name : Raj
Pateint's Full Name : Raj Samant
Pateint's Age : 62
Pateint's Blood Group : O-
Pateint's Gender : Male
Pateint's Phone Number : 9845386453
Pateint's Disease : Cold

******Main Menu******
1. Add Pateint in the Queue
2. Remove Pateint from the Queue
3. Next Pateint Information
4. Display all Pateints in the Queue
5. Exit
Enter your choice: 5
Exiting Application

*****Main Menu*****
1. Appointment Registration
2. Pateint List
3. Doctor Availability List
4. Staff Members Details List
5. Exit
Enter your choice: 2

*****Pateint List*****

*****Main Menu*****
1. Create a Pateint's list
2. Display the Pateint's list
3. Add a Pateint at the beginning
4. Add a Pateint at the end
5. Add a Pateint before a given Pateint
6. Add a Pateint after a given Pateint
7. Delete a Pateint at the beginning
8. Delete a Pateint at the end
9. Delete a given Pateint
10. Delete a Pateint after a given Pateint
11. Delete the entire Pateint's list
12. Exit
Enter your choice 1
Enter -1 to end

Enter the Pateint Number : 1
Enter the data of the pateint :
Enter the Pateint's Name Amit
Enter the Pateint's Full Name Amit Kumar
Enter the Pateint's Age 41
Enter the Pateint's Blood Group A+
Enter the Pateint's Gender Male
Enter the Pateint's Phone Number 9845238865
Enter the Pateint's Disease Dengue
Enter the Pateint's allocated Doctor's Name Dr Mahesh
Enter the Pateint's date of admission 12 October
Enter the Pateint's treatment type Special
Enter the Pateint's medicines Crocin

Enter the Pateint Number : 2
Enter the data of the pateint :
Enter the Pateint's Name Mohan
Enter the Pateint's Full Name Mohan Thenua
Enter the Pateint's Age 65
Enter the Pateint's Blood Group AB+
Enter the Pateint's Gender Male
Enter the Pateint's Phone Number 9856473453
Enter the Pateint's Disease Malaria
Enter the Pateint's allocated Doctor's Name Dr Kumar
Enter the Pateint's date of admission 11 October
Enter the Pateint's treatment type Special++
Enter the Pateint's medicines Erithromicin

Enter the Pateint Number : 3
Enter the data of the pateint :
Enter the Pateint's Name Aditya

Enter the Pateint's Full Name Aditya Samant
Enter the Pateint's Age 52
Enter the Pateint's Blood Group A+
Enter the Pateint's Gender Male
Enter the Pateint's Phone Number 9867534213
Enter the Pateint's Disease Cholera
Enter the Pateint's allocated Doctor's Name Dr Patel
Enter the Pateint's date of admission 15 October
Enter the Pateint's treatment type Special
Enter the Pateint's medicines Norflox

Enter the Pateint Number : 4
Enter the data of the pateint :
Enter the Pateint's Name Raj
Enter the Pateint's Full Name Raj Singh
Enter the Pateint's Age 21
Enter the Pateint's Blood Group B+
Enter the Pateint's Gender Male
Enter the Pateint's Phone Number 9865414321
Enter the Pateint's Disease Fracture
Enter the Pateint's allocated Doctor's Name Dr Shah
Enter the Pateint's date of admission 17 October
Enter the Pateint's treatment type Normal++
Enter the Pateint's medicines D Tablets
Enter the Pateint Number : -1
Linked List Created

*****Main Menu*****
1. Create a Pateint's list
2. Display the Pateint's list
3. Add a Pateint at the beginning
4. Add a Pateint at the end
5. Add a Pateint before a given Pateint
6. Add a Pateint after a given Pateint
7. Delete a Pateint at the beginning
8. Delete a Pateint at the end
9. Delete a given Pateint
10. Delete a Pateint after a given Pateint
11. Delete the entire Pateint's list
12. Exit
Enter your choice 2

Pateint's Number : 1
Pateint's Name : Amit
Pateint's Full Name : Amit Kumar
Pateint's Age : 41
Pateint's Blood Group : A+
Pateint's Gender : Male
Pateint's Phone Number : 9845238865Dengue
Pateint's Disease : Dengue
Pateint's allocated Doctor's Name : Dr Mahesh
Pateint's date of admission : 12 OctoberSpecial

Pateint's treatment type : Special
Pateint's medicines : Crocin

Pateint's Number : 2
Pateint's Name : Mohan
Pateint's Full Name : Mohan Thenua
Pateint's Age : 65
Pateint's Blood Group : AB+
Pateint's Gender : Male
Pateint's Phone Number : 9856473453Malaria
Pateint's Disease : Malaria
Pateint's allocated Doctor's Name : Dr Kumar
Pateint's date of admission : 11 OctoberSpecial++
Pateint's treatment type : Special++
Pateint's medicines : Erithromicin

Pateint's Number : 3
Pateint's Name : Aditya
Pateint's Full Name : Aditya Samant
Pateint's Age : 52
Pateint's Blood Group : A+
Pateint's Gender : Male
Pateint's Phone Number : 9867534213Cholera
Pateint's Disease : Cholera
Pateint's allocated Doctor's Name : Dr Patel
Pateint's date of admission : 15 OctoberSpecial
Pateint's treatment type : Special
Pateint's medicines : Norflox

Pateint's Number : 4
Pateint's Name : Raj
Pateint's Full Name : Raj Singh
Pateint's Age : 21
Pateint's Blood Group : B+
Pateint's Gender : Male
Pateint's Phone Number : 9865414321Fracture
Pateint's Disease : Fracture
Pateint's allocated Doctor's Name : Dr Shah
Pateint's date of admission : 17 OctoberNormal++
Pateint's treatment type : Normal++
Pateint's medicines : D Tablets

*****Main Menu*****
1. Create a Pateint's list
2. Display the Pateint's list
3. Add a Pateint at the beginning
4. Add a Pateint at the end
5. Add a Pateint before a given Pateint
6. Add a Pateint after a given Pateint
7. Delete a Pateint at the beginning
8. Delete a Pateint at the end
9. Delete a given Pateint

10. Delete a Pateint after a given Pateint
11. Delete the entire Pateint's list
12. Exit
Enter your choice 12
Exiting Application

*****Main Menu*****
1. Appointment Registration
2. Pateint List
3. Doctor Availability List
4. Staff Members Details List
5. Exit
Enter your choice: 3

*****Doctor Availablity List*****

*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 1
Enter the Name of Available Doctor : Mahesh
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 1
Enter the Name of Available Doctor : Amit
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 1
Enter the Name of Available Doctor : Soham
*****Main Menu*****

1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 1
Enter the Name of Available Doctor : Parth
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 1
Enter the Name of Available Doctor : Vishal
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 1
Enter the Name of Available Doctor : Ramesh
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 2
The Doctor appointed is : Ramesh
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit

Enter your option 3
The next Doctor to be appointed is : Vishal
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 4
The available Doctors are :
Vishal
Parth
Soham
Amit
Mahesh
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 6
Enter the name of the Doctor to be searched Soham
The Doctor Soham's name is found at location a[2]
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 5
The Doctor Availability List in sorted order is
Amit
Mahesh
Parth
Soham
Vishal
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List

6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 7
Enter the name of the Doctor to be searched Parth
Doctor Parth's name is present at the array position a[2]
*****Main Menu*****
1. Enter a Doctor in Doctor Availablity List
2. Appoint a Doctor from Doctor Avalibility List
3. Next Doctor available for appointment
4. Display all Doctors available
5. Sort the Doctor Availability List
6. Search the Doctor's name using Linear Search
7. Search the Doctor's name using Binary Search
8. Exit
Enter your option 8
The program is terminated

*****Main Menu*****
1. Appointment Registration
2. Pateint List
3. Doctor Availability List
4. Staff Members Details List
5. Exit
Enter your choice: 4

*****Staff Member Management Portal*****

*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit
Enter your choice 1
Enter the details of the Staff Members :
Enter the Staff's Name Mahesh
Enter the Staff's Full Name Mahesh Samant
Enter the Staff's Age 52
Enter the Staff's Gender Male
Enter the Staff's Phone Number 9845623123
Enter the Staff's Designation Secretary
Enter the Staff's Salary 100000
Enter the Staff's allocated Lab's Number 210
*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit

Enter your choice 1
Enter the details of the Staff Members :
Enter the Staff's Name Ramesh
Enter the Staff's Full Name Ramesh Kumar
Enter the Staff's Age 42
Enter the Staff's Gender Male
Enter the Staff's Phone Number 9845387654
Enter the Staff's Designation HOD
Enter the Staff's Salary 1000000
Enter the Staff's allocated Lab's Number 103
*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit
Enter your choice 1
Enter the details of the Staff Members :
Enter the Staff's Name Soham
Enter the Staff's Full Name Rohan Patel
Enter the Staff's Age 25
Enter the Staff's Gender Male
Enter the Staff's Phone Number 9845877676
Enter the Staff's Designation Cheif
Enter the Staff's Salary 1000000
Enter the Staff's allocated Lab's Number 415
*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit
Enter your choice 1
Enter the details of the Staff Members :
Enter the Staff's Name Amit
Enter the Staff's Full Name Amit Pradhan
Enter the Staff's Age 34
Enter the Staff's Gender Male
Enter the Staff's Phone Number 986577656
Enter the Staff's Designation Head
Enter the Staff's Salary 300000
Enter the Staff's allocated Lab's Number 105
*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit
Enter your choice 2
Please Enter the Staff Member to be searched Soham

Staff's Name : Soham

Staff's Full Name : Rohan Patel
Staff's Age : 25
Staff's Gender : Male
Staff's Phone Number : 9845877676
Staff's Designation : Cheif
Staff's Salary : 1000000
Staff's allocated Lab Number : 415
*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit
Enter your choice 2
Please Enter the Staff Member to be searched Ramesh

Staff's Name : Ramesh
Staff's Full Name : Ramesh Kumar
Staff's Age : 42
Staff's Gender : Male
Staff's Phone Number : 9845387654
Staff's Designation : HOD
Staff's Salary : 1000000
Staff's allocated Lab Number : 103
*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit
Enter your choice 3
Total Staff Members in the Hospital is 4
*****Main Menu*****
1. Insert Staff Member
2. Search Staff Member
3. Find Total Staff Members in the Hospital
4. Delete all Staff Member's Records
5. Exit
Enter your choice 5
Exiting Application

*****Main Menu*****
1. Appointment Registration
2. Pateint List
3. Doctor Availability List
4. Staff Members Details List
5. Exit
Enter your choice: 5
Exiting Application
nirish@Nirish:~/Desktop$