DBMS ASSIGNMENT – 2

SEQUENTIAL FILE PROCESSING

***Roll Number: U19CS012***

***Name: BHAGYA VINOD RANA***

(Q1) In continuation with the earlier program 1,

Write a ‘C’ program to manage doctors information who are giving treatments to COVID patients using one text file for database of doctors and another text file to relate doctor and patient.

**Code:**

*// DOCTORS INFORMATION [U19CS012] [BHAGYA VINOD RANA]*

*#include* <stdio.h>

*#include* <string.h>

*#include* <stdbool.h>

*// -----------------GENERAL\_STRUCTURES\_TO\_STORE\_INFORMATION[START]------------------*

*// 1-Structure of Date*

typedef struct \_date

{

    int dd;

    int mm;

    int yy;

} date;

*// 2-Structure of Patient [COVID]*

typedef struct \_covid

{

    int number;

    char first[20];

    char last[20];

    char gender;

    int age;

    char area[20];

    date admission;

    date discharge;

} covid, mypat;

*// 3-Structure of Doctor*

typedef struct \_doctor

{

    int number;

    char name[20];

    char gender;

    int age;

    char area[20];

} doctor;

*// 4-Structure to Store the Relation Between Doc & Patient*

typedef struct \_relation

{

    int doc\_num;

    int patients[100];

    int pat\_num;

    char name[20];

} relation;

*// -----------------GENERAL\_STRUCTURES\_TO\_STORE\_INFORMATION[END]------------------*

*// -------------MODULAR\_PROGRAMMING\_STRUCTURE\_HELPER\_FUNCTION[START]--------------*

*// 1- Add Doctor to DataBase*

void write();

*// 2- Display List of Doctors with Corresponding Number of Patients [Summary 4A]*

void print\_num\_patients();

*// 3 - Assign a Patient to the Doctor with the Least Number of Patients*

void assign\_patient();

*// 3A - Patient to be Assigned*

int add\_pat\_util();

*// 4 - Display Various Reports Doctor-Wise*

void summary\_reports();

*// 4B - Gender-Wise Patient Summary*

void gender\_count();

*// 4C - Age-Wise Patient Summary*

void age\_count();

*// 4D - Area-Wise Patient Summary*

void area\_count();

*// 5 - Sort Patients in Ascending Order*

void sort\_ascending();

*// 5A1 - Helper Function to Compare Dates*

int compare\_dates(date d1, date d2);

void sort\_adate();

void sort\_ddate();

void sort\_doc();

void sort\_mdoc();

void sort\_fdoc();

*// 6B - Add Patients in DataBase [Incase of No Patients]*

void new\_pat\_entry();

*// 6A - Function Caller for Each Patient*

void add\_new\_patients();

*// -------------MODULAR\_PROGRAMMING\_STRUCTURE\_HELPER\_FUNCTION[END]--------------*

int main()

{

    int cont = 1, choice;

*while* (1)

    {

        printf("\nChoose What you want to do?\n");

        printf("\nNote: Add Patient's Data from Option 6 [If Not Done Earlier!]\n\n");

        printf("1 -> Add Doctor Info\n");

        printf("2 -> Display List of Doctors with Corresponding Number of Patients\n");

        printf("3 -> Assign a Patient to the Doctor with the Least Number of Patients\n");

        printf("4 -> Display Various Reports Doctor-Wise\n");

        printf("5 -> Sort Patients in Ascending Order\n");

        printf("6 -> Add Patients in DataBase [Incase of No Patients]\n");

        printf("7 -> Quit\n");

        scanf("%d", &choice);

*switch* (choice)

        {

*case* 1:

            write();

*break*;

*case* 2:

            print\_num\_patients();

*break*;

*case* 3:

            assign\_patient();

*break*;

*case* 4:

            printf("Select Parameter on which You Want Doctor-Wise Report :\n");

            printf("1 -> Display Count of Patient(s)\n");

            printf("2 -> Display Count of Patient based on Gender\n");

            printf("3 -> Display Age-Wise Count of Patient(s)\n");

            printf("4 -> Display Area-Wise Count of Patient(s)\n");

            fflush(stdin);

            summary\_reports();

*break*;

*case* 5:

            printf("Select what you wish to do\n");

            printf("1 -> Admitted on Same date\n");

            printf("2 -> Discharged on Same date\n");

            printf("3 -> Patients under Doctor\n");

            printf("4 -> Patients under Male Doctor\n");

            printf("5 -> Patients under Female Doctor\n");

            sort\_ascending();

*break*;

*case* 6:

            add\_new\_patients();

*break*;

*case* 7: *//Quit*

            cont = 0;

*break*;

*default*:

            printf("Enter a Valid choice!!!\n");

        }

*if* (!cont)

*break*;

    }

*return* 0;

}

*//-----------------------------------ADD\_DOCTORS\_SECTION[START]------------------------------------------*

*// 1- Add Doctor to DataBase*

void write()

{

    int i;

    doctor d;

    relation r;

    FILE \*fp;

    FILE \*fp1;

    fp = fopen("doctors.txt", "a+");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    fp1 = fopen("doctor\_patient\_relation.txt", "a+");

*if* (fp1 == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    printf("~~ENTER DOCTOR DETAILS~~\n");

*// DOCTORS INFORMATION SECTION*

    printf("Enter Doctor's Number :\n");

    scanf("%d", &d.number);

    printf("Enter Doctor's Name :\n");

    fflush(stdin);

    gets(d.name);

    printf("Enter Doctor's Gender [(M/F)Only] :\n");

    fflush(stdin);

    scanf("%c", &d.gender);

    printf("Enter Doctor's Age :\n");

    fflush(stdin);

    scanf("%d", &d.age);

    printf("Enter Doctor's Area of Residence :\n");

    fflush(stdin);

    gets(d.area);

*// Also Insert the Relation*

    r.doc\_num = d.number;

    strcpy(r.name, d.name);

    printf("Enter Number of Patient(s) Currently Under Doctor [Count]:\n");

    fflush(stdin);

    scanf("%d", &r.pat\_num);

*// Initialise*

*for* (i = 0; i < 100; ++i)

    {

        r.patients[i] = 0;

    }

    printf("Enter Patient ID's :\n");

    fflush(stdin);

    int x;

*for* (i = 0; i < r.pat\_num; ++i)

    {

        scanf("%d", &x);

        r.patients[x] = 1; *// Mark the Patient*

    }

    fwrite(&d, sizeof(doctor), 1, fp);

    fwrite(&r, sizeof(relation), 1, fp1);

    fclose(fp);

    fclose(fp1);

}

*//-----------------------------------ADD\_DOCTORS\_SECTION[END]------------------------------------------*

*//-------------------------------DISPLAY\_DOC\_PATIENT\_NO\_SECTION[START]---------------------------------*

*// 2- Display List of Doctors with Corresponding Number of Patients*

void print\_num\_patients()

{

    relation r;

    FILE \*fp;

    fp = fopen("doctor\_patient\_relation.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

*while* (fread(&r, sizeof(relation), 1, fp))

    {

        printf("Doctor ID : %d\n", r.doc\_num);

        printf("Number of Patient(s) : %d\n", r.pat\_num);

        printf("\n");

    }

    fclose(fp);

}

*//-------------------------------DISPLAY\_DOC\_PATIENT\_NO\_SECTION[END]---------------------------------*

*//-------------------------------ASSIGN\_PATIENT\_TO\_LEAST\_DOC[START]----------------------------------*

*// 3 - Assign a Patient to the Doctor with the Least Number of Patients*

void assign\_patient()

{

    FILE \*fp;

    fp = fopen("doctor\_patient\_relation.txt", "r+");

*if* (fp == NULL)

    {

        printf("Failed to Load File!!!!\n");

        fclose(fp);

*return*;

    }

    int p\_id = add\_pat\_util();

    relation arr[100];

    relation r;

    int count = 0;

*while* (fread((&r), sizeof(relation), 1, fp))

    {

        arr[count++] = r;

    }

    fclose(fp);

    char min\_name[] = "zzzzzzzzzzzzzz";

    int min\_count = 100000000;

    int pos = -1, i;

*for* (i = 0; i < count; ++i)

    {

*if* (arr[i].pat\_num < min\_count)

        {

            min\_count = arr[i].pat\_num;

            strcpy(min\_name, arr[i].name);

            pos = i;

        }

*else* *if* (arr[i].pat\_num == min\_count)

        {

*if* (strcmp(min\_name, arr[i].name) < 0)

            {

                strcpy(min\_name, arr[i].name);

                pos = i;

            }

        }

    }

    ++arr[pos].pat\_num;

    arr[pos].patients[p\_id] = 1;

    FILE \*temp;

    temp = fopen("temp.txt", "w");

*for* (i = 0; i < count; ++i)

    {

        fwrite(&arr[i], sizeof(relation), 1, temp);

    }

    fclose(temp);

    remove("doctor\_patient\_relation.txt");

    rename("temp.txt", "doctor\_patient\_relation.txt");

    printf("Summary : \n");

    printf("Patient Number %d referred to Dr. %s", p\_id, arr[pos].name);

}

int add\_pat\_util()

{

    FILE \*fp;

    fp = fopen("patients.txt", "a+");

    printf("~~ENTER PATIENT DETAILS~~\n");

    struct \_covid p;

    printf("Patients Number : \n");

    scanf("%d", &p.number);

    printf("Patients First Name : \n");

    fflush(stdin);

    gets(p.first);

    printf("Patients Last Name : \n");

    fflush(stdin);

    gets(p.last);

    printf("Patients Age : \n");

    scanf("%d", &p.age);

    printf("Patients Gender : \n");

    fflush(stdin);

    scanf("%c", &p.gender);

    printf("Patients Admission Date [DD/MM/YYYY] : \n");

    scanf("%d %d %d", &p.admission.dd, &p.admission.mm, &p.admission.yy);

    printf("Patients Discharge Date [DD/MM/YYYY] :  \n");

    scanf("%d %d %d", &p.discharge.dd, &p.discharge.mm, &p.discharge.yy);

    printf("Patient Area of Residence : \n");

    fflush(stdin);

    gets(p.area);

    fwrite(&p, sizeof(struct \_covid), 1, fp);

    fclose(fp);

*// Only Difference is that it Returns Patient(s) Number*

*return* p.number;

}

*//-------------------------------ASSIGN\_PATIENT\_TO\_LEAST\_DOC[START]----------------------------------*

*//-----------------------------DISPLAY\_VARIOUS\_SUMMARY\_REPORTS[START]---------------------------*

*// 4 - Display Various Reports Doctor-Wise*

void summary\_reports()

{

    int choice;

    scanf("%d", &choice);

*switch* (choice)

    {

*case* 1:

        print\_num\_patients();

*break*;

*case* 2:

        gender\_count();

*break*;

*case* 3:

        age\_count();

*break*;

*case* 4:

        area\_count();

*break*;

*default*:

        printf("Invalid choice\n");

    }

}

*// 4B - Gender-Wise Patient Summary*

void gender\_count()

{

    FILE \*fp;

    fp = fopen("doctor\_patient\_relation.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    relation r;

*while* (fread(&r, sizeof(relation), 1, fp))

    {

        printf("Doctor ID : %d\n", r.doc\_num);

        int male = 0, female = 0;

        covid p1;

        FILE \*fp1;

        fp1 = fopen("patients.txt", "r");

*while* (fread(&p1, sizeof(covid), 1, fp1))

        {

*if* (r.patients[p1.number])

            {

*if* (p1.gender == 'm' || p1.gender == 'M')

                    ++male;

*else*

                    ++female;

            }

        }

        fclose(fp1);

        printf("No. Of Male Patient(s)   : %d\n", male);

        printf("No. Of Female Patient(s) : %d\n", female);

    }

    fclose(fp);

}

*// 4C - Age-Wise Patient Summary*

void age\_count()

{

    FILE \*fp;

    fp = fopen("doctor\_patient\_relation.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    relation r;

*while* (fread(&r, sizeof(relation), 1, fp))

    {

        printf("Doctor ID : %d\n", r.doc\_num);

        covid p1;

        int i;

        FILE \*fp1;

        fp1 = fopen("patients.txt", "r");

*// Max Age = 150 [Frequency Array]*

        int age[150] = {0};

*while* (fread(&p1, sizeof(covid), 1, fp1))

        {

*if* (r.patients[p1.number])

                ++age[p1.age];

        }

        fclose(fp1);

*for* (i = 0; i < 150; ++i)

        {

*if* (age[i])

                printf("Number of Patient(s) of Age %d : %d\n", i, age[i]);

        }

    }

    fclose(fp);

}

*// 4D - Area-Wise Patient Summary*

void area\_count()

{

    FILE \*fp;

    fp = fopen("doctor\_patient\_relation.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    relation r;

*while* (fread(&r, sizeof(relation), 1, fp))

    {

        printf("Doctor ID : %d\n", r.doc\_num);

        covid p1;

        int i;

        FILE \*fp1;

        fp1 = fopen("patients.txt", "r");

*while* (fread(&p1, sizeof(covid), 1, fp1))

        {

            char area[20];

            char arr[100][20];

            int temp = 0;

*if* (r.patients[p1.number])

            {

                int count = 0, flag = 0, i;

                strcpy(area, p1.area);

*for* (i = 0; i < temp; ++i)

                {

*if* (strcmp(area, arr[i]) == 0)

                    {

                        flag = 1;

*break*;

                    }

                }

*if* (!flag)

                {

                    strcpy(arr[temp], area);

                    ++temp;

                    FILE \*fp2;

                    covid p2;

                    fp2 = fopen("patients.txt", "r");

*while* (fread(&p2, sizeof(covid), 1, fp2))

                    {

*if* (strcmp(area, p2.area) == 0)

                        {

*if* (r.patients[p2.number])

                                ++count;

                        }

                    }

                    printf("Number of Patient(s) from Area %s : %d\n", area, count);

                    fclose(fp2);

                }

            }

        }

        fclose(fp1);

    }

    fclose(fp);

}

*//-----------------------------DISPLAY\_VARIOUS\_SUMMARY\_REPORTS[START]---------------------------*

*//-----------------------------SORT\_PATIENTS\_ASCENDING\_ORDER[START]-----------------------------*

void sort\_ascending()

{

    int choice;

    scanf("%d", &choice);

*switch* (choice)

    {

*case* 1:

        sort\_adate();

*break*;

*case* 2:

        sort\_ddate();

*break*;

*case* 3:

        sort\_doc();

*break*;

*case* 4:

        sort\_mdoc();

*break*;

*case* 5:

        sort\_fdoc();

*break*;

*default*:

        printf("Invalid Input\n");

    }

}

int compare\_dates(date d1, date d2)

{

*if* (d1.yy > d2.yy)

*return* 1;

*if* (d1.yy < d2.yy)

*return* -1;

*if* (d1.mm > d2.mm)

*return* 1;

*if* (d1.mm < d2.mm)

*return* -1;

*if* (d1.dd > d2.dd)

*return* 1;

*if* (d1.dd < d2.dd)

*return* -1;

*return* 0;

}

void sort\_adate()

{

    FILE \*fp;

    fp = fopen("patients.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    covid p1, arr[100];

    date d;

    printf("Enter Date for which you want to Sort\n");

    scanf("%d %d %d", &d.dd, &d.mm, &d.yy);

    int count = 0;

*while* (fread(&p1, sizeof(covid), 1, fp))

    {

*if* (compare\_dates(d, p1.admission) == 0)

        {

            arr[count++] = p1;

        }

    }

    fclose(fp);

*if* (!count)

    {

        printf("No Patient admitted on the Given Date\n");

*return*;

    }

*// Bubble Sort the Data Based on First Name*

    int i, j;

*for* (i = 0; i < count; ++i)

    {

*for* (j = 0; j < count - i - 1; ++j)

        {

*if* (strcmp(arr[j].first, arr[j + 1].first) > 0)

            {

                p1 = arr[j];

                arr[j] = arr[j + 1];

                arr[j + 1] = p1;

            }

        }

    }

    printf("Sorted Data :\n");

*for* (i = 0; i < count; ++i)

    {

        printf("\nPatient Number is : %d", arr[i].number);

        printf("\nFirst Name is : %s", arr[i].first);

        printf("\nLast Name is : %s", arr[i].last);

        printf("\nAge is : %d", arr[i].age);

        printf("\nGender is : %c", arr[i].gender);

        printf("\nArea is : %s", arr[i].area);

        printf("\nAdmission date is : %d/%d/%d", arr[i].admission.dd, arr[i].admission.mm, arr[i].admission.yy);

        printf("\nDischarge date is : %d/%d/%d\n", arr[i].discharge.dd, arr[i].discharge.mm, arr[i].discharge.yy);

    }

}

void sort\_ddate()

{

    FILE \*fp;

    fp = fopen("patients.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    covid p1, arr[100];

    date d;

    printf("Enter Date for which you want to Sort\n");

    scanf("%d %d %d", &d.dd, &d.mm, &d.yy);

    int count = 0;

*while* (fread(&p1, sizeof(covid), 1, fp))

    {

*if* (compare\_dates(d, p1.discharge) == 0)

        {

            arr[count++] = p1;

        }

    }

    fclose(fp);

*if* (!count)

    {

        printf("No patient discharged on the given date\n");

*return*;

    }

*// Bubble Sort Based on First Name*

    int i, j;

*for* (i = 0; i < count; ++i)

    {

*for* (j = 0; j < count - i - 1; ++j)

        {

*if* (strcmp(arr[j].first, arr[j + 1].first) > 0)

            {

                p1 = arr[j];

                arr[j] = arr[j + 1];

                arr[j + 1] = p1;

            }

        }

    }

    printf("Sorted Data :\n");

*for* (i = 0; i < count; ++i)

    {

        printf("\nPatient Number is : %d", arr[i].number);

        printf("\nFirst Name is : %s", arr[i].first);

        printf("\nLast Name is : %s", arr[i].last);

        printf("\nAge is : %d", arr[i].age);

        printf("\nGender is : %c", arr[i].gender);

        printf("\nArea is : %s", arr[i].area);

        printf("\nAdmission date is : %d/%d/%d", arr[i].admission.dd, arr[i].admission.mm, arr[i].admission.yy);

        printf("\nDischarge date is : %d/%d/%d\n", arr[i].discharge.dd, arr[i].discharge.mm, arr[i].discharge.yy);

    }

}

void sort\_doc()

{

    FILE \*fp;

    fp = fopen("doctor\_patient\_relation.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    relation r, arr[100];

    int d\_id, count = 0;

    printf("Enter the Doctor's ID whose patients have to be sorted\n");

    scanf("%d", &d\_id);

*while* (fread(&r, sizeof(relation), 1, fp))

    {

        arr[count++] = r;

    }

    fclose(fp);

    int count2 = 0;

    covid arr2[100], p;

    int i;

    bool flag = false;

*for* (i = 0; i < count; i++)

    {

*if* (arr[i].doc\_num == d\_id)

        {

            FILE \*fp1;

            fp1 = fopen("patients.txt", "a+");

*while* (fread(&p, sizeof(covid), 1, fp1))

            {

*if* (arr[i].patients[p.number])

                {

                    arr2[count2++] = p;

                }

            }

            flag = true;

            fclose(fp1);

*break*;

        }

    }

*if* (!flag)

    {

        printf("Doctor with the given ID does not exist!\n");

*return*;

    }

*if* (count2 == 0)

    {

        printf("The doctor has No Patients Assigned!\n");

*return*;

    }

*// Bubble Sort the Data Based on First Name*

    int j;

*for* (i = 0; i < count2; ++i)

    {

*for* (j = 0; j < count2 - i - 1; ++j)

        {

*if* (strcmp(arr2[j].first, arr2[j + 1].first) > 0)

            {

                p = arr2[j];

                arr2[j] = arr2[j + 1];

                arr2[j + 1] = p;

            }

        }

    }

    printf("Sorted Data :\n");

*for* (i = 0; i < count2; ++i)

    {

        printf("\nPatient Number is : %d", arr2[i].number);

        printf("\nFirst Name is : %s", arr2[i].first);

        printf("\nLast Name is : %s", arr2[i].last);

        printf("\nAge is : %d", arr2[i].age);

        printf("\nGender is : %c", arr2[i].gender);

        printf("\nArea is : %s", arr2[i].area);

        printf("\nAdmission date is : %d/%d/%d", arr2[i].admission.dd, arr2[i].admission.mm, arr2[i].admission.yy);

        printf("\nDischarge date is : %d/%d/%d\n", arr2[i].discharge.dd, arr2[i].discharge.mm, arr2[i].discharge.yy);

    }

    fclose(fp);

}

void sort\_mdoc()

{

    FILE \*fp;

    fp = fopen("doctors.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    doctor d, arr[100];

    int count = 0;

*while* (fread(&d, sizeof(doctor), 1, fp))

    {

*if* (d.gender == 'm' || d.gender == 'M')

        {

            arr[count] = d;

            count++;

        }

    }

    fclose(fp);

    FILE \*fp1;

    fp1 = fopen("doctor\_patient\_relation.txt", "r+");

    relation r, arr1[100];

    int rcount = 0, i, j;

*while* (fread(&r, sizeof(relation), 1, fp1))

    {

        arr1[rcount] = r;

        rcount++;

    }

    fclose(fp1);

*for* (i = 0; i < count; ++i)

    {

        int d\_id = arr[i].number;

        covid p1, arr2[100];

        int pcount = 0;

*for* (j = 0; j < rcount; ++j)

        {

*if* (arr1[j].doc\_num == d\_id)

            {

                FILE \*fp2;

                fp2 = fopen("patients.txt", "r");

*while* (fread(&p1, sizeof(covid), 1, fp2))

                {

*if* (arr1[j].patients[p1.number])

                    {

                        arr2[pcount] = p1;

                        pcount++;

                    }

                }

                fclose(fp2);

*break*;

            }

        }

*if* (pcount == 0)

        {

            printf("The Male Doctor Don't have any Assigned Patient\n");

*continue*;

        }

        int k;

*for* (k = 0; k < pcount; ++k)

        {

*for* (j = 0; j < pcount - k - 1; ++j)

            {

*if* (strcmp(arr2[j].first, arr2[j + 1].first) > 0)

                {

                    p1 = arr2[j];

                    arr2[j] = arr2[j + 1];

                    arr2[j + 1] = p1;

                }

            }

        }

        printf("Male Doctor ID : %d\n", d\_id);

        printf("Sorted Patient(s)\n");

*for* (i = 0; i < pcount; ++i)

        {

            printf("\nPatient Number is : %d", arr2[i].number);

            printf("\nFirst Name is : %s", arr2[i].first);

            printf("\nLast Name is : %s", arr2[i].last);

            printf("\nAge is : %d", arr2[i].age);

            printf("\nGender is : %c", arr2[i].gender);

            printf("\nArea is : %s", arr2[i].area);

            printf("\nAdmission date is : %d/%d/%d", arr2[i].admission.dd, arr2[i].admission.mm, arr2[i].admission.yy);

            printf("\nDischarge date is : %d/%d/%d\n", arr2[i].discharge.dd, arr2[i].discharge.mm, arr2[i].discharge.yy);

        }

    }

}

void sort\_fdoc()

{

    FILE \*fp;

    fp = fopen("doctors.txt", "r");

*if* (fp == NULL)

    {

        printf("Failed to Load File!\n");

        fclose(fp);

*return*;

    }

    doctor d, arr[100];

    int count = 0;

*while* (fread(&d, sizeof(doctor), 1, fp))

    {

*if* (d.gender == 'f' || d.gender == 'F')

        {

            arr[count++] = d;

        }

    }

    fclose(fp);

    FILE \*fp1;

    fp1 = fopen("doctor\_patient\_relation.txt", "r+");

    relation r, arr1[100];

    int rcount = 0, i, j;

*while* (fread(&r, sizeof(relation), 1, fp1))

    {

        arr1[rcount++] = r;

    }

    fclose(fp1);

*for* (i = 0; i < count; ++i)

    {

        int d\_id = arr[i].number;

        covid p1, arr2[100];

        int pcount = 0;

*for* (j = 0; j < rcount; ++j)

        {

*if* (arr1[j].doc\_num == d\_id)

            {

                FILE \*fp2;

                fp2 = fopen("patients.txt", "r");

*while* (fread(&p1, sizeof(covid), 1, fp2))

                {

*if* (arr1[j].patients[p1.number])

                    {

                        arr2[pcount++] = p1;

                    }

                }

                fclose(fp2);

*break*;

            }

        }

*if* (pcount == 0)

        {

            printf("The Female Doctor Don't have any Assigned Patient\n");

*continue*;

        }

        int k;

*for* (k = 0; k < pcount; ++k)

        {

*for* (j = 0; j < pcount - k - 1; ++j)

            {

*if* (strcmp(arr2[j].first, arr2[j + 1].first) > 0)

                {

                    p1 = arr2[j];

                    arr2[j] = arr2[j + 1];

                    arr2[j + 1] = p1;

                }

            }

        }

        printf("Female Doctor ID : %d\n", d\_id);

        printf("Sorted Patients\n");

*for* (i = 0; i < pcount; ++i)

        {

            printf("\nPatient Number is : %d", arr2[i].number);

            printf("\nFirst Name is : %s", arr2[i].first);

            printf("\nLast Name is : %s", arr2[i].last);

            printf("\nAge is : %d", arr2[i].age);

            printf("\nGender is : %c", arr2[i].gender);

            printf("\nArea is : %s", arr2[i].area);

            printf("\nAdmission date is : %d/%d/%d", arr2[i].admission.dd, arr2[i].admission.mm, arr2[i].admission.yy);

            printf("\nDischarge date is : %d/%d/%d\n", arr2[i].discharge.dd, arr2[i].discharge.mm, arr2[i].discharge.yy);

        }

    }

}

*//-----------------------------SORT\_PATIENTS\_ASCENDING\_ORDER[END]-----------------------------*

*//-----------------------------------ADD\_PATIENTS\_SECTION[START]--------------------------------*

*// 6 - Add Patients in DataBase [Incase of No Patients]*

void new\_pat\_entry()

{

    FILE \*fp;

    fp = fopen("patients.txt", "a+");

    printf("~~ENTER PATIENT DETAILS~~\n");

    struct \_covid p;

    printf("Patients Number : \n");

    scanf("%d", &p.number);

    printf("Patients First Name : \n");

    fflush(stdin);

    gets(p.first);

    printf("Patients Last Name : \n");

    fflush(stdin);

    gets(p.last);

    printf("Patients Age : \n");

    scanf("%d", &p.age);

    printf("Patients Gender : \n");

    fflush(stdin);

    scanf("%c", &p.gender);

    printf("Patients Admission Date [DD/MM/YYYY] : \n");

    scanf("%d %d %d", &p.admission.dd, &p.admission.mm, &p.admission.yy);

    printf("Patients Discharge Date [DD/MM/YYYY] :  \n");

    scanf("%d %d %d", &p.discharge.dd, &p.discharge.mm, &p.discharge.yy);

    printf("Patient Area of Residence : \n");

    fflush(stdin);

    gets(p.area);

    fwrite(&p, sizeof(struct \_covid), 1, fp);

    fclose(fp);

*return*;

}

void add\_new\_patients()

{

    int pcnt;

    printf("Enter the Number of Patient's You Want to Add : ");

    scanf("%d", &pcnt);

*for* (int i = 0; i < pcnt; i++)

    {

        new\_pat\_entry();

    }

    printf("\nSuccessfully Added %d Patients Data in DataBase!!\n", pcnt);

}

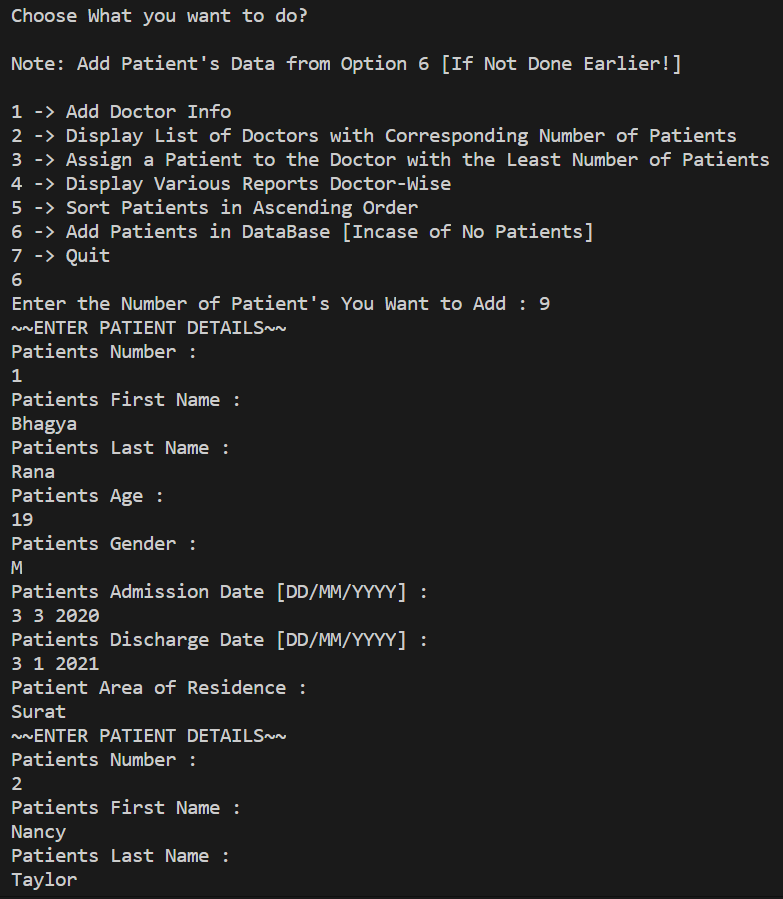
*//-----------------------------------ADD\_PATIENTS\_SECTION[END]-------------------------------*

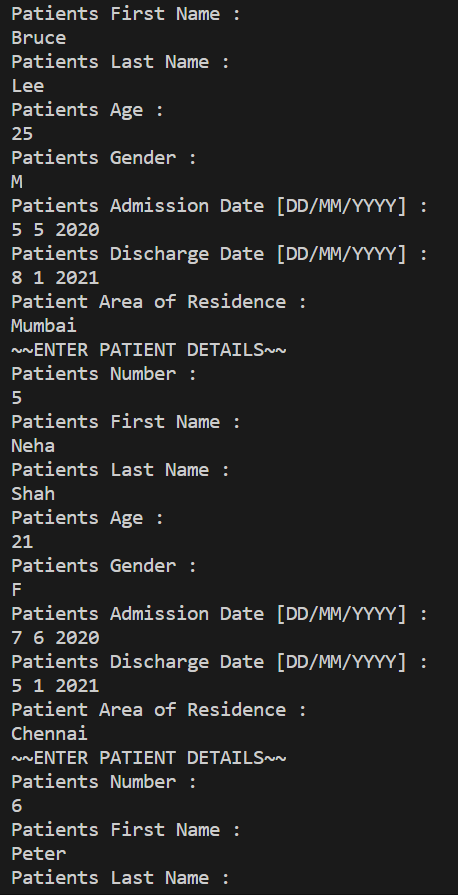
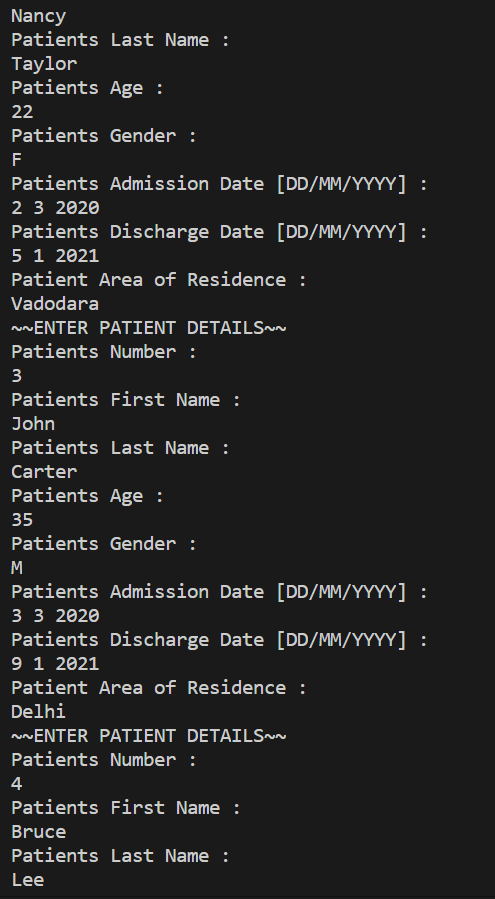
**Output:**

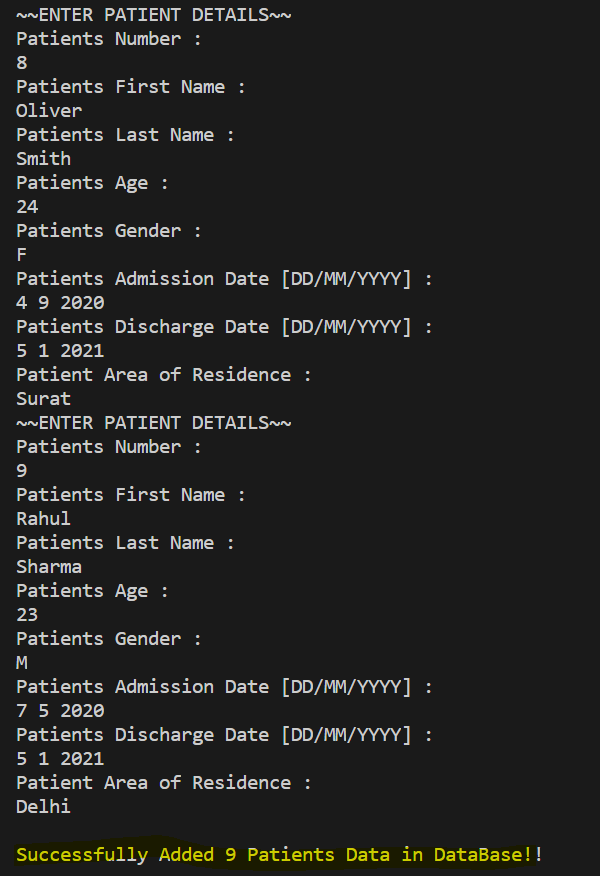
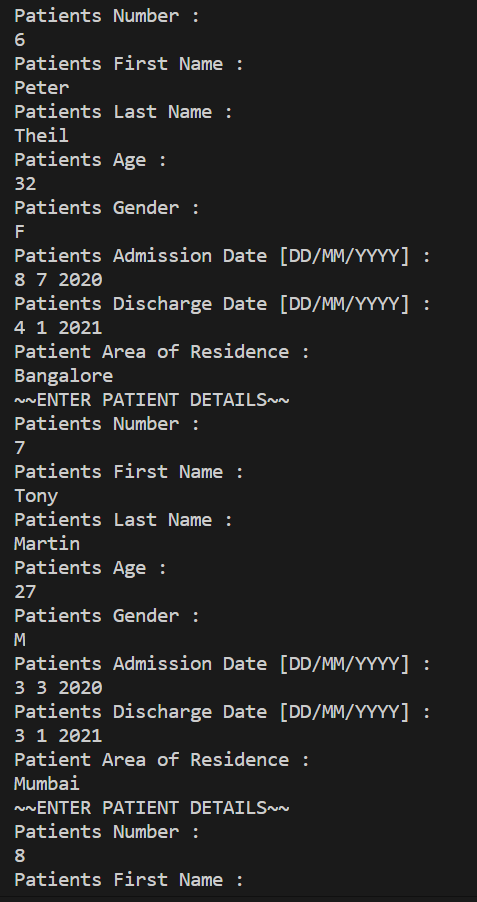
**A.) Enter the Data of 9 Patients in the Data-Base**

Data of 9 Patients Added:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| PNo. | First Name | Last Name | Age | Gender | Area | Adm. Date | Dis. Date |
| 1 | Bhagya | Rana | 19 | M | Surat | 3/3/20 | 3/1/21 |
| 2 | Nancy | Taylor | 22 | F | Vadodara | 2/3/20 | 5/1/21 |
| 3 | John | Carter | 35 | M | Delhi | 3/3/20 | 9/1/21 |
| 4 | Bruce | Lee | 25 | M | Mumbai | 5/5/20 | 8/1/21 |
| 5 | Neha | Shah | 21 | F | Chennai | 7/6/20 | 5/1/21 |
| 6 | Peter | Theil | 32 | F | Bangalore | 8/7/20 | 4/1/21 |
| 7 | Tony | Martin | 27 | M | Mumbai | 3/3/20 | 3/1/21 |
| 8 | Oliver | Smith | 24 | F | Surat | 4/9/20 | 5/1/21 |
| 9 | Rahul | Sharma | 23 | M | Delhi | 7/5/20 | 5/1/21 |

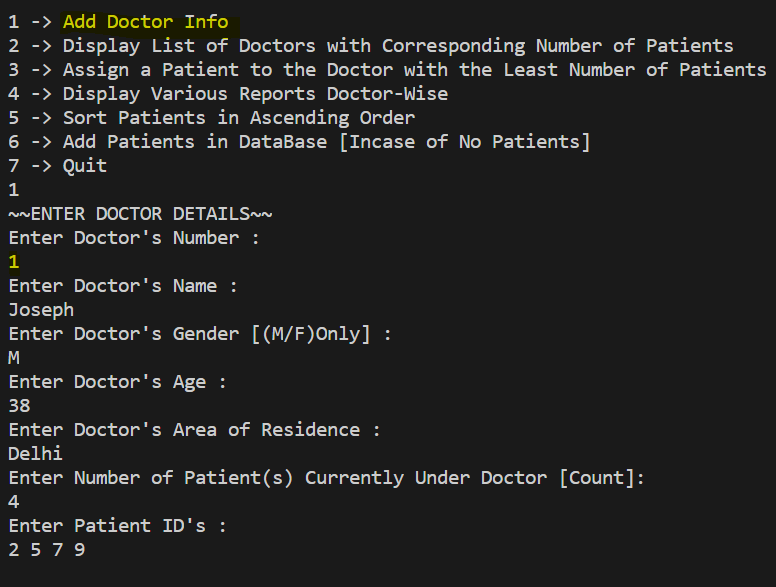


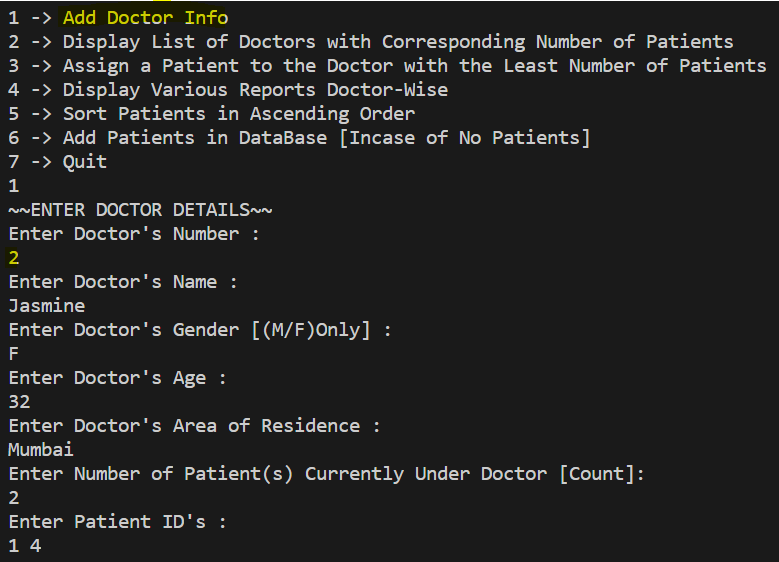


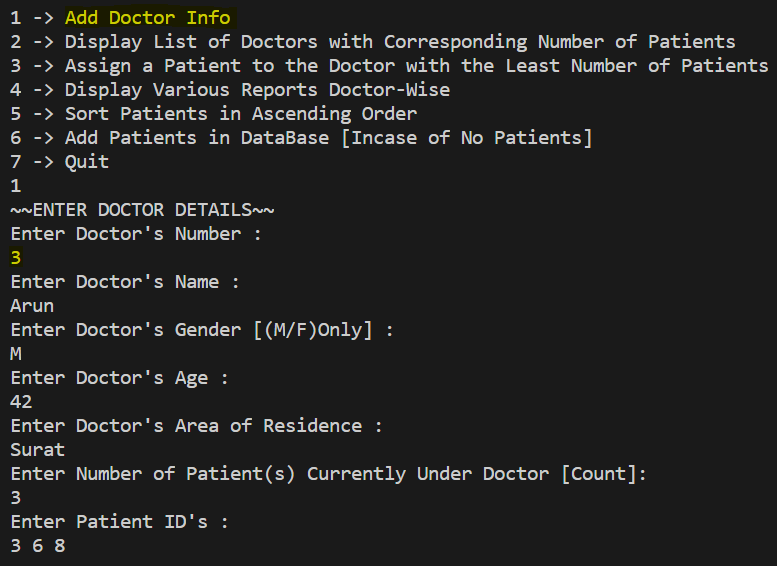


**B.) Add 3 Doctors in the Data-Base**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DNo. | Name | Gender | Age | Area | Count of Patients | Patient IDs |
| 1 | Joseph | M | 38 | Delhi | 4 | 2,5,7,9 |
| 2 | Jasmine | F | 32 | Mumbai | 2 | 1,4 |
| 3 | Arun | M | 42 | Surat | 3 | 3,6,8 |



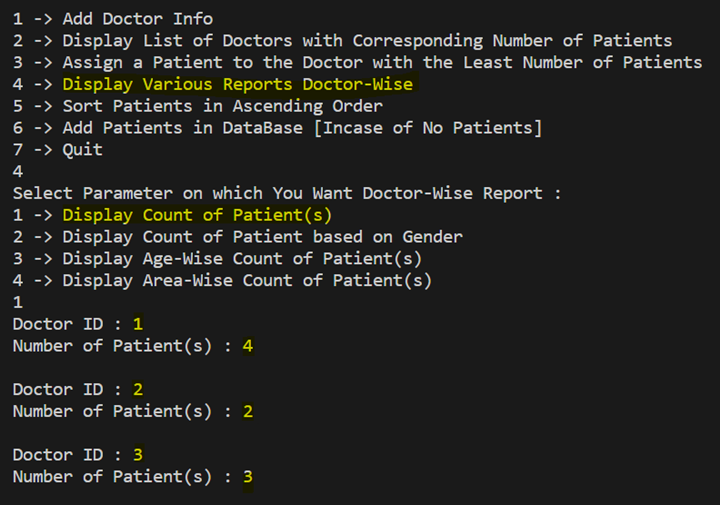




**C.) Display Various Report Doctor-Wise**

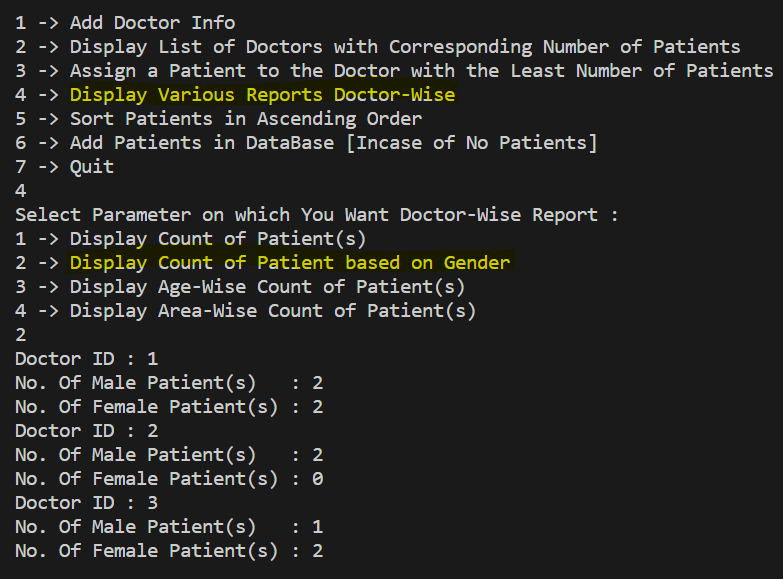
C1.) Count of Patients

|  |  |  |
| --- | --- | --- |
| DNo. | Name | Count of Patients |
| 1 | Joseph | 4 |
| 2 | Jasmine | 2 |
| 3 | Arun | 3 |

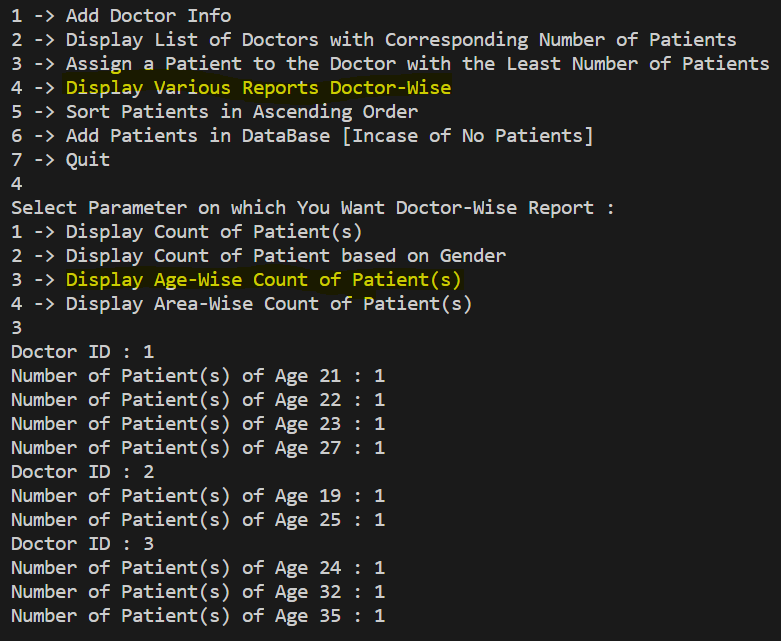


C2.) Based on Gender

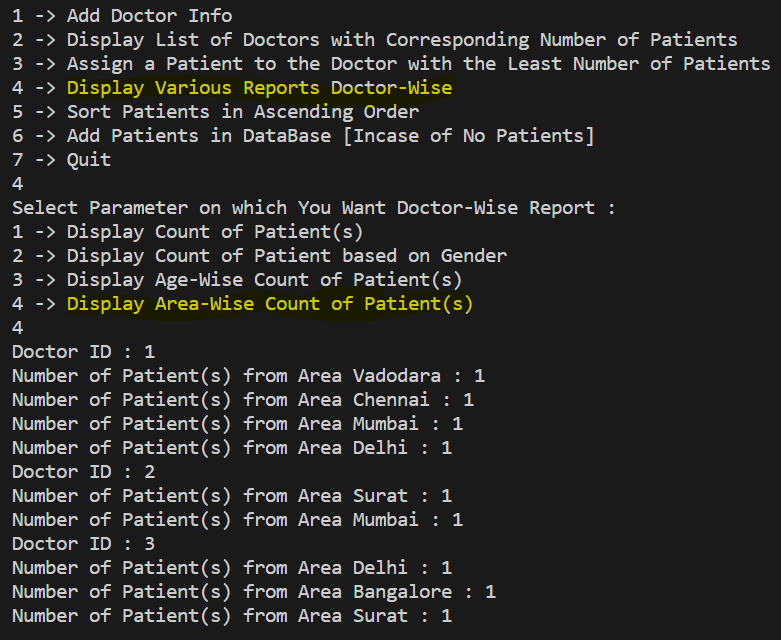
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DNo. | Name | Count of Patients | Patient IDs | Male | Female |
| 1 | Joseph | 4 | 2[F],5[F],7[M],9[M] | 2 | 2 |
| 2 | Jasmine | 2 | 1[M],4[M] | 2 | 0 |
| 3 | Arun | 3 | 3[M],6[F],8[F] | 1 | 2 |



C3.) Age-Wise Count of Patients



C4.) Area-Wise Count of Patients

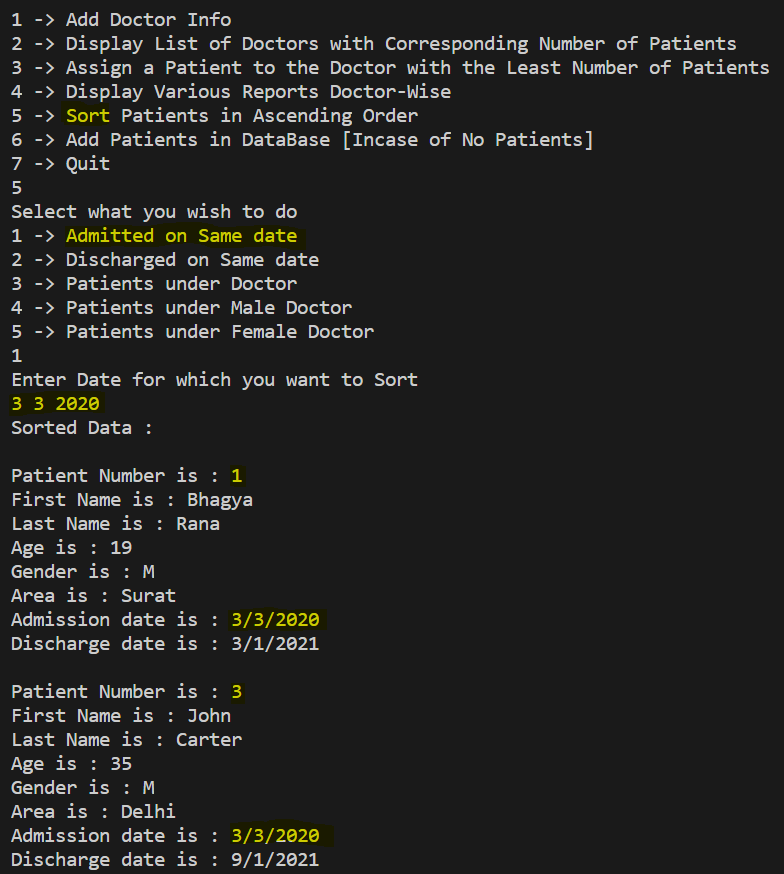


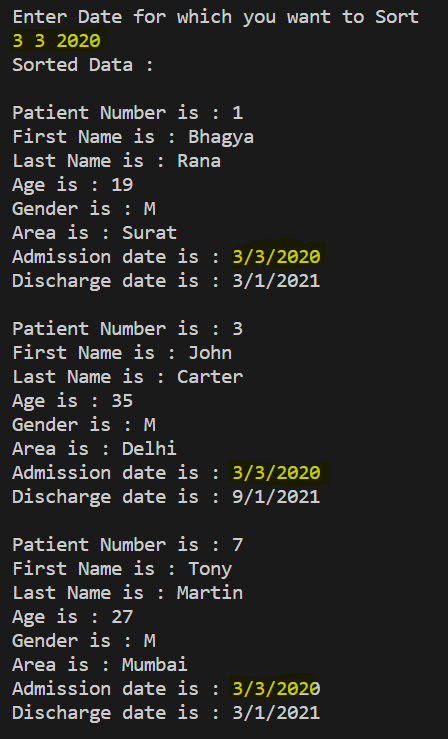
**D.) Sort Patients in Ascending Order**

**PARAMETER**

D1.) Admitted on Same Date

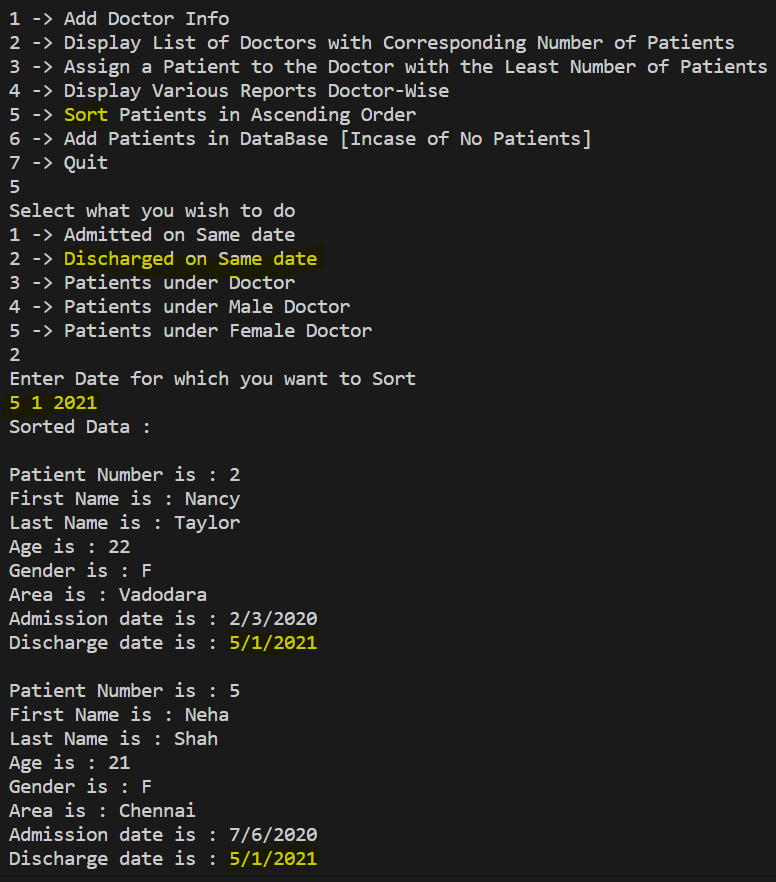
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| PNo. | First Name | Last Name | Age | Gender | Area | Adm. Date | Dis. Date |
| 1 | Bhagya | Rana | 19 | M | Surat | 3/3/20 | 3/1/21 |
| 3 | John | Carter | 35 | M | Delhi | 3/3/20 | 9/1/21 |
| 7 | Tony | Martin | 27 | M | Mumbai | 3/3/20 | 3/1/21 |

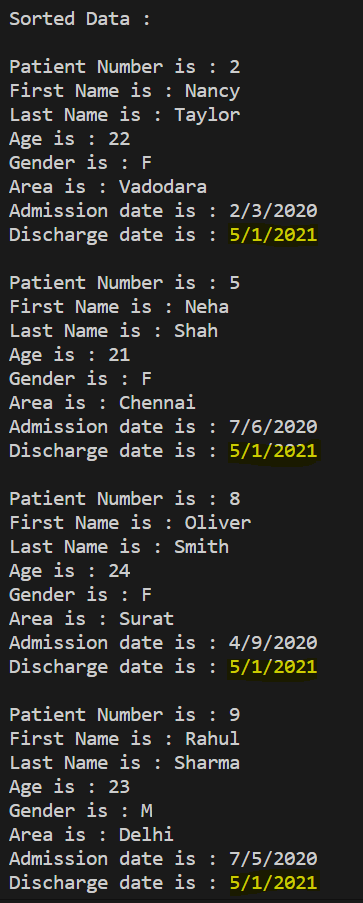




D2.) Discharged on Same Date

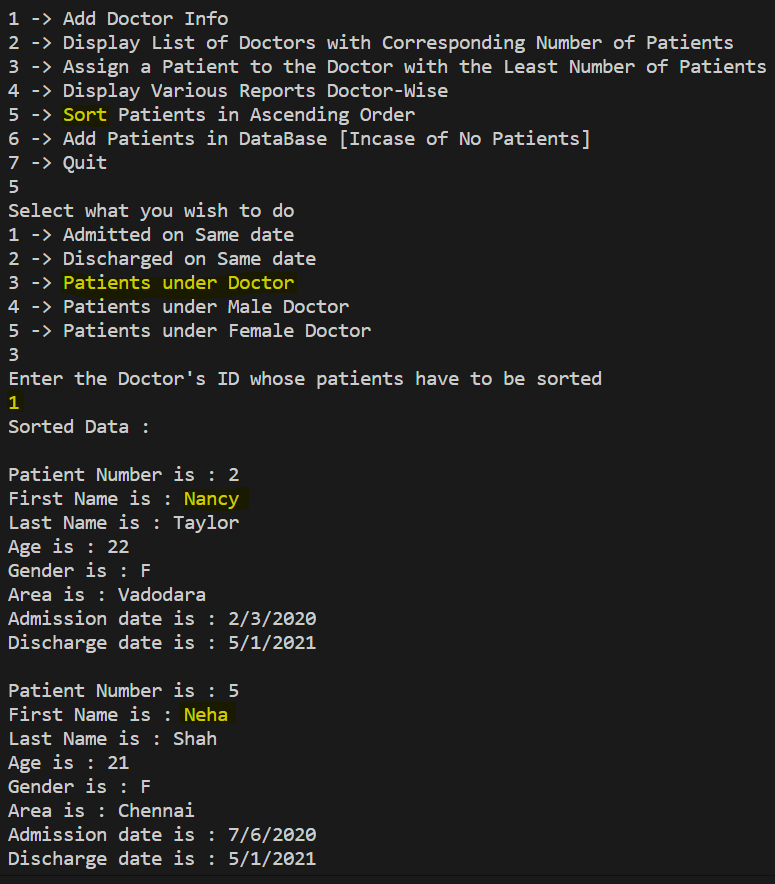
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| PNo. | First Name | Last Name | Age | Gender | Area | Adm. Date | Dis. Date |
| 2 | Nancy | Taylor | 22 | F | Vadodara | 2/3/20 | 5/1/21 |
| 5 | Neha | Shah | 21 | F | Chennai | 7/6/20 | 5/1/21 |
| 8 | Oliver | Smith | 24 | F | Surat | 4/9/20 | 5/1/21 |
| 9 | Rahul | Sharma | 23 | M | Delhi | 7/5/20 | 5/1/21 |

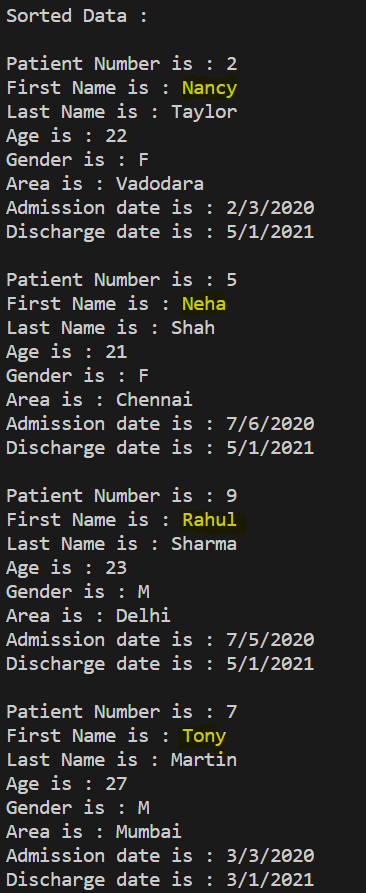




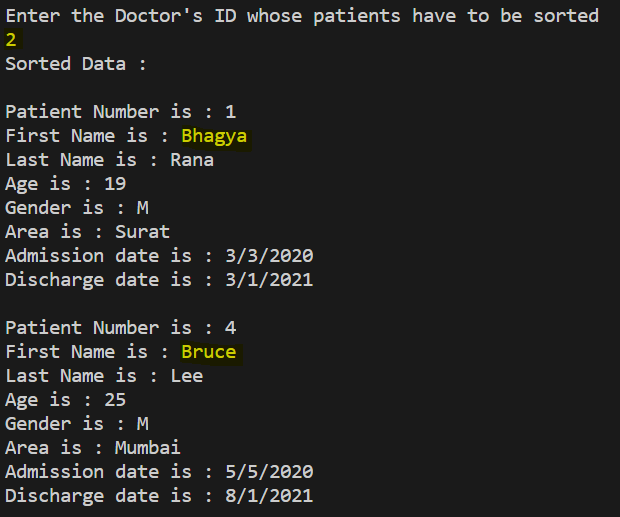
D3.) Patients under Doctor

**DOCTOR 1**

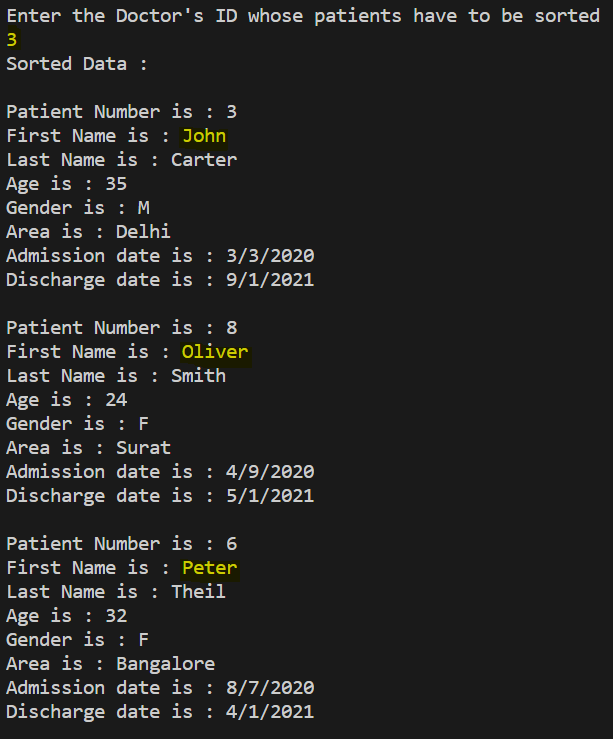




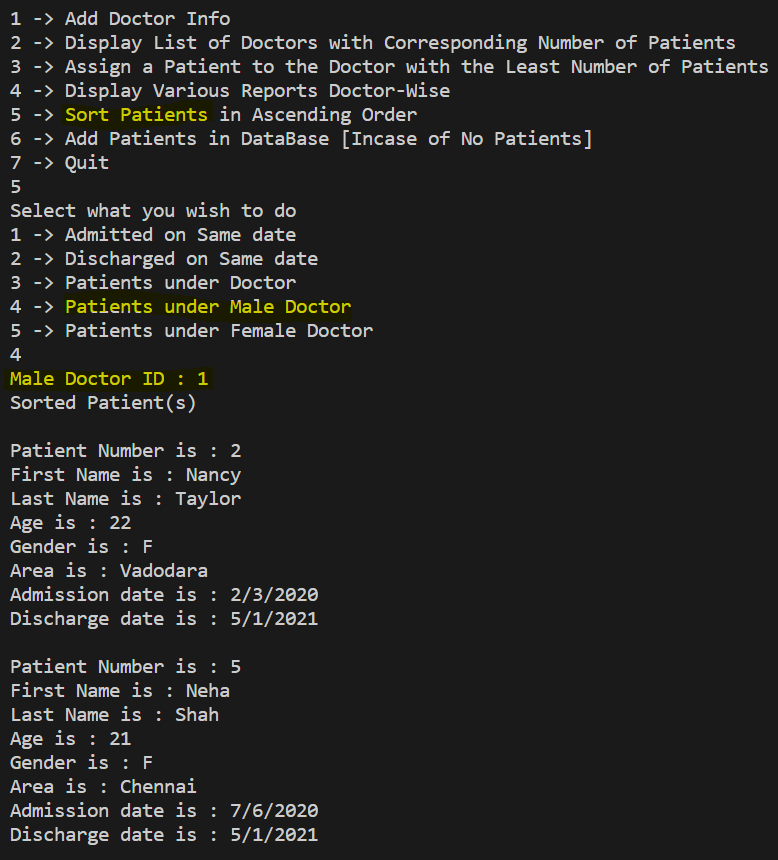
**DOCTOR 2**

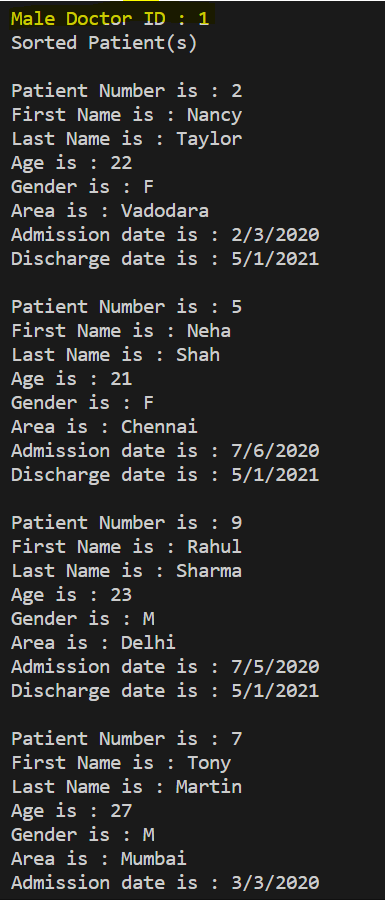


**DOCTOR 3**

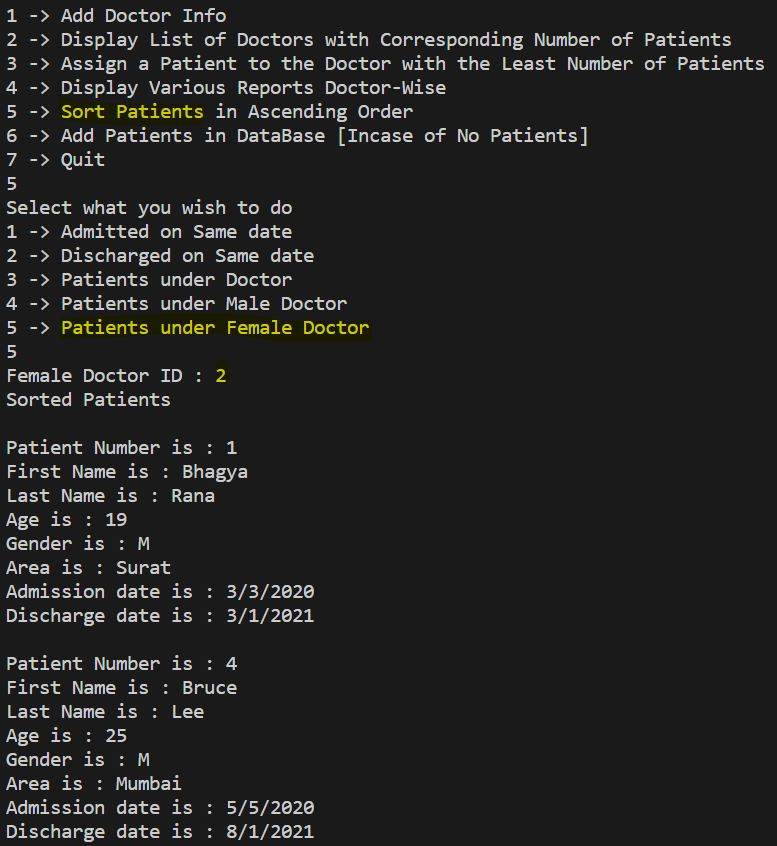


D4.) Patients under Male Doctor



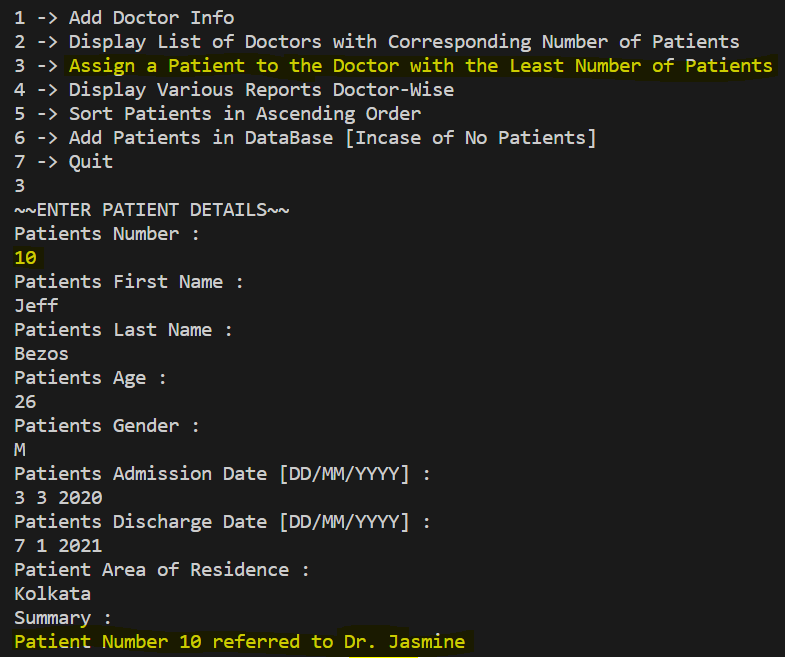


D5.) Patients under Female Doctor

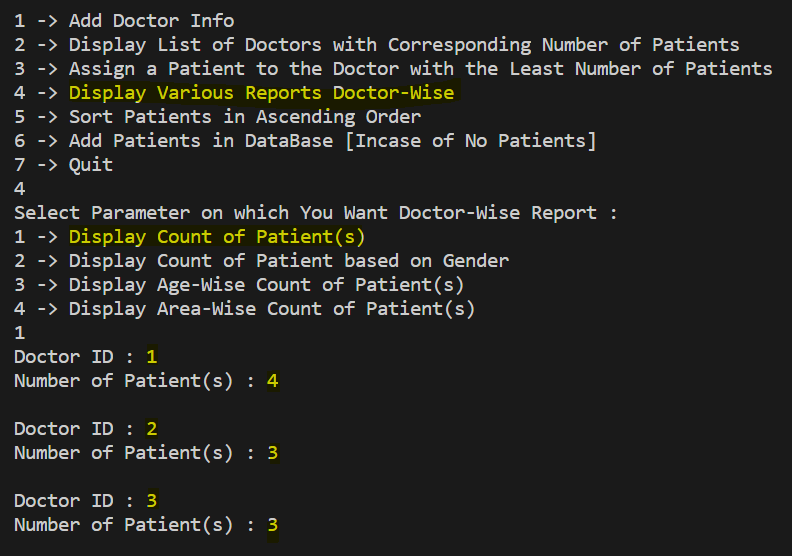


**E.) Assign a Patient to Doctor with Least Number of Patients**

Since **Dr. Jasmine** has Only **2 Patients** [as Compared to 4 & 3 Patients of Other Doctors], She will be Assigned the New Patient.



Count of Patients After Assignment of New Patient [2 Patients -> 3 Patients]



**Submitted By:**

**BHAGYA VINOD RANA**

**U19CS012**