```
/*
Requirements:
Define Data Types:
Create a structure Patient with the following fields:
patientID (integer): Unique identifier for each patient.
name (string): Name of the patient.
age (integer): Age of the patient.
testResults (array of 5 floats): Results for 5 medical tests.
averageScore (float): Average of the test results (calculated field).
Create a union ContactInfo to store either:
phoneNumber (string): Patient's contact number.
email (string): Patient's email address.
Features:
Dynamic Memory Allocation:
Dynamically allocate memory for an array of Patient structures based on
user input (N patients).
Input and Output:
Input the details of each patient, including their name, age, test
results, and contact information.
Calculate and store the averageScore for each patient based on their
test results.
Display:
Display all patient details in a tabular format, including the
averageScore.
Search:
Search for a patient by their ID and display their details.
Update:
Update a patient's test results, recalculate their averageScore, and
optionally update their contact information.
Sorting:
Sort patients by their averageScore in descending order.
Typedef:
Use typedef to define aliases for the Patient structure and the
ContactInfo union.

Program Requirements:
Menu Options:
Add new patient details.
Display all patient records.
Search for a patient by ID.
Update test results or contact information.
Sort patients by their average test score in descending order.
Exit the program.
```

```c
*/
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef struct Patient{
    int patientID;
    char name[50];
    int age;
    float testresult[5];
    float avgscore;
}patient;
typedef union {
    char phonenumber[100];
    char email[100];
}Contactinfo;
void addpatient(patient* patients, Contactinfo* contacts,int* count);
void display(patient* patients,Contactinfo* contacts,int count);
void search(patient* patients,Contactinfo* contacts,int count,int id);
void update(patient* patients,Contactinfo* contacts,int count,int id);
void sort(patient* patients,Contactinfo* contacts,int count);
int main(){
    int op,id,op2;
    int count=0,n;
    patient *patients=NULL;
    Contactinfo *contacts=NULL;
    printf("\nEnter number of Patients::");
    scanf("%d",&n);
    patients=(patient*)malloc(n*sizeof(patient));
    contacts=(Contactinfo*)malloc(n*sizeof(Contactinfo));
    while(1){
        printf("\nPATIENT MANAGEMENT");
        printf("\n1.Add new patient details.\n2.Display all patient 
records.\n3.Search for a patient by ID.\n4.Update test results or 
contact information.\n5.Sort patients by their average test score in 
descending order.\n6.Exit the program.");
        printf("\nChoose an Option:::");
        scanf("%d",&op);
        switch(op){
            case 1:

                addpatient(patients,contacts,&count);
                break;
            case 2:
                display(patients,contacts,count);
                break;
            case 3:
                printf("\nEnter ID to Search::");
                scanf("%d",&id);
                search(patients,contacts,count,id);
                break;
            case 4:
                printf("\nEnter Patient ID::");
                scanf("%d",&id);
                update(patients,contacts,count,id);
                break;
            case 5:
                sort(patients,contacts,count);
                break;

            case 6:
                printf("\nExiting...");
                free(patients);
                free(contacts);
                return 0;
                break;
        }
    }

}
void addpatient(patient* patients, Contactinfo* contacts,int* count){
    printf("\nEnter Patient ID::");
    scanf("%d",&patients[*count].patientID);
    printf("\nEnter Patient Name::");
    scanf("%s",patients[*count].name);
    printf("\nEnter Age::");
    scanf("%d",&patients[*count].age);
    printf("\nEnter Result for %d tests",5);
    for(int i=0;i<5;i++){
        scanf("%f",&patients[*count].testresult[i]);
    }
    float avg,sum=0.0;
    for(int i=0;i<5;i++){
        sum+=patients[*count].testresult[i];
    }
    avg=sum/5;
    patients[*count].avgscore=avg;
```

```c
    printf("\nEnter Contact Information::");
    printf("\n1.Phone");
    printf("\n2.Email");
    int op1;
    printf("\nChoose an Option::");
    scanf("%d",&op1);
    if(op1==1){
        printf("\nEnter Phone::");
        scanf("%s",contacts[*count].phonenumber);
    }
    else if(op1==2){
        printf("\nEnter email::");
        scanf("%s",contacts[*count].email);
    }
    else{
        printf("\nWRONG");
    }
    (*count)++;



}
void display(patient* patients,Contactinfo* contacts,int count){
    printf("\nID\tName\tAge\tAverageScore\tContact");
    for(int i=0;i<count;i++){

printf("\n%d\t%s\t%d\t%.2f\t%s",patients[i].patientID,patients[i].name,
patients[i].age,patients[i].avgscore,contacts[i].phonenumber);

printf("\n-------------------------------------------------------------
");
    }
}
void search(patient* patients,Contactinfo* contacts,int count,int id){
    for(int i=0;i<count;i++){
        if(patients[i].patientID == id){
            printf("\nData Found");
            printf("\nID\tName\tAge\tAverageScore\tContact");

printf("\n%d\t%s\t%d\t%.2f\t%s",patients[i].patientID,patients[i].name,
patients[i].age,patients[i].avgscore,contacts[i].phonenumber);
```

```c
        }
    }
}
void update(patient* patients,Contactinfo* contacts,int count,int id){
    int op2,found;
    for(int i=0;i<count;i++){
        if(patients[i].patientID == id){
            found=i;
        }
    }
    printf("\nWhich to update\n1.Result\n2.Contact");
    printf("\nChoose ::");
    scanf("%d",&op2);
    if(op2==1){
        printf("\nEnter Result for %d tests::",5);
        for(int i=0;i<5;i++){
            scanf("%f",&patients[found].testresult[i]);
        }
        float avg,sum=0.0;
        for(int i=0;i<5;i++){
            sum+=patients[found].testresult[i];
        }
        avg=sum/5;
        patients[found].avgscore=avg;

    }
    else if(op2==2){
        printf("\nEnter Contact Information::");
        printf("\n1.Phone");
        printf("\n2.Email");
        int op1;
        printf("\nChoose an Option::");
        scanf("%d",&op1);
        if(op1==1){
            printf("\nEnter Phone::");
            scanf("%s",contacts[found].phonenumber);
        }
        else if(op1==2){
            printf("\nEnter email::");
            scanf("%s",contacts[found].email);
        }
        else{
            printf("\nWRONG");
        }


    }
    else{
        printf("\nWRONG");
    }


}
void sort(patient* patients,Contactinfo* contacts,int count){
    for (int i = 0; i < count - 1; i++) {
        for (int j = i + 1; j < count; j++) {
            if (patients[i].avgscore < patients[j].avgscore) {
                patient temp = patients[i];
                patients[i] = patients[j];
                patients[j] = temp;

                Contactinfo tempContact = contacts[i];
                contacts[i] = contacts[j];
                contacts[j] = tempContact;
            }
        }
    }

    printf("\nSORTED LIST");
    display(patients,contacts,count);
}
```