/*Problem Statement: Aerospace Fleet Management System

Design and implement a program to manage Aerospace Fleet Records using C programming. The system will handle details about different aerospace vehicles, such as airplanes, helicopters, and drones, while demonstrating the use of structures, unions, dynamic memory allocation, and typedef.

Requirements

1. Define Data Types

Create a structure Vehicle with the following fields:

vehicleID (integer): Unique identifier for each vehicle.

vehicleType (string): Type of the vehicle (e.g., "Airplane," "Helicopter," "Drone").

capacity (integer): Maximum passenger or payload capacity (depending on vehicle type).

range (float): Maximum range of the vehicle in kilometers.

status (string): Current operational status (e.g., "Active," "Maintenance").

Create a union SpecialAttributes to store type-specific details:

airplane (integer): Number of engines (for airplanes).

helicopter (integer): Rotor blade count (for helicopters).

drone (float): Payload capacity in kilograms (for drones).

2. Features

Dynamic Memory Allocation:

Dynamically allocate memory for an array of Vehicle structures based on user input (N vehicles).

Input and Output:

Input details for each vehicle, including vehicleType and its specific attributes.

Allow the user to choose which specific field of the union to populate based on the vehicle type.

Display:

Display all vehicle records in a tabular format, including general details and type-specific attributes.

Search:

Allow the user to search for a vehicle by vehicleID and display its details.

Update:

Update the operational status or type-specific attributes of a vehicle.

Sorting:

Sort vehicles by their range in descending order.

3. Typedef:

Use typedef to define aliases for the Vehicle structure and the SpecialAttributes union.

Program Requirements

1. Menu Options

Add new vehicle details.

Display all vehicle records.

Search for a vehicle by ID.

Update vehicle details (status or specific attributes).

Sort vehicles by range in descending order.

Exit the program.

*/

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef union{
   int airplane;
   int helicopter;
   float drone;
}SpecialAttributes;
typedef struct{
   int vehicleID;
   char vehicleType[50];
   int capacity;
   float range;
   char status[50];
   SpecialAttributes attributes;
}Vehicle;
void addVehicle(Vehicle *vehicles,int n);
void Display(Vehicle *vehicles,int n);
void search(Vehicle *vehicles,int n,int id);
void update(Vehicle *vehicles,int n);
```

```
void sort(Vehicle *vehicles,int n);
int main(){
   int N;
   printf("Enter the number of vehicles: ");
   scanf("%d",&N);
   Vehicle *vehicles=NULL;
   vehicles = (Vehicle *)malloc(N*sizeof(Vehicle));
   int op;
   while(1){
      printf("Menu options\n");
      printf("1.Add new vehicle details\n2.Display all vehicle records\n3.Search for a vehicle by ID\n4.Update vehicle details\n5.Sort vehicles by range in descending order\n6.Exit\n");
      printf("Choose your option: ");
      scanf("%d",&op);
      switch(op){
         case 1:
            addVehicle(vehicles,N);
            break;
         case 2:
            Display(vehicles,N);
            break;
         case 3:
            int id;
            printf("Enter the vehicle ID to search: ");
            scanf("%d",&id);
            search(vehicles,N,id);
            break;
         case 4:
            update(vehicles,N);
            break;
         case 5:
```

```
            sort(vehicles,N);
            break;
        case 6:
            exit(0);
            break;
        default:
            printf("Incorrect option");
            break;
    }
  }
}
void addVehicle(Vehicle *vehicles,int n){
  for(int i=0;i<n;i++){
    printf("Enter vehicle ID: ");
    scanf("%d",&vehicles[i].vehicleID);
    printf("Enter the type of vehicle(Airplane,Helicopter,Drone): ");
    scanf("%s",&vehicles[i].vehicleType);
    printf("Enter vehicle capacity: ");
    scanf("%d",&vehicles[i].capacity);
    printf("Enter maximum range of vehicles in kilometers: ");
    scanf("%f",&vehicles[i].range);
    printf("Enter current operational status: ");
    scanf("%s",&vehicles[i].status);
    if(strcmp(vehicles[i].vehicleType,"Airplane")==0){
      printf("Enter number of engines: ");
      scanf("%d",&vehicles[i].attributes.airplane);
    }
    else if(strcmp(vehicles[i].vehicleType,"Helicopter")==0){
      printf("Enter rotor blade count: ");
      scanf("%d",&vehicles[i].attributes.helicopter);
    }
```

```
        else if(strcmp(vehicles[i].vehicleType,"Drone")==0){
            printf("Enter payload cpacity in kilograms: ");
            scanf("%f",&vehicles[i].attributes.drone);
        }
        else{
            printf("Invalid vehicle type");
        }
    }
}
void Display(Vehicle *vehicles,int n){
    for(int i=0;i<n;i++){
        printf("Vehicle ID: %d, vehicle type: %s, Capacity: %d, Range: %f, Status: %s,
",vehicles[i].vehicleID,vehicles[i].vehicleType,vehicles[i].capacity,vehicles[i].range,vehicles[i].status);
        if(strcmp(vehicles[i].vehicleType,"Airplane")==0){
            printf("Number of engines: %d\n",vehicles[i].attributes.airplane);
        }
        else if(strcmp(vehicles[i].vehicleType,"Helicopter")==0){
            printf("Rotor blade count: %d\n",vehicles[i].attributes.helicopter);
        }
        else if(strcmp(vehicles[i].vehicleType,"Drone")==0){
            printf("Payload capacity: %.2f\n",vehicles[i].attributes.drone);
        }
    }
}
void search(Vehicle *vehicles,int n,int id){
    int found=0;
    for(int i=0;i<n;i++){
        if(vehicles[i].vehicleID==id){
            found=1;
            printf("Vehicle found\n");
```

```
            printf("Vehicle ID: %d vehicle type: %s Capacity: %d Range: %f Status:
%s",vehicles[i].vehicleID,vehicles[i].vehicleType,vehicles[i].capacity,vehicles[i].range,vehicles[i].status
);
        if(strcmp(vehicles[i].vehicleType,"Airplane")==0){
            printf("Number of engines: %d",vehicles[i].attributes.airplane);
        }
        else if(strcmp(vehicles[i].vehicleType,"Helicopter")==0){
            printf("Rotor blade count: %d",vehicles[i].attributes.helicopter);
        }
        else if(strcmp(vehicles[i].vehicleType,"Drone")==0){
            printf("Payload capacity: %.2f",vehicles[i].attributes.drone);
        }
        }
    }
    if(!found){
        printf("Vehicle not found");
    }
}
void update(Vehicle *vehicles,int n){
    int id;
    printf("Enter the id of vehicle to update details: ");
    scanf("%d",&id);
    int found=0;
    for(int i=0;i<n;i++){
        if(vehicles[i].vehicleID==id){
            found=1;
            printf("Vehicle found.What would you like to update?\n");
            printf("1.Status\n");
            printf("2.Specific attributes\n");
            int choice;
            printf("Enter your choice: ");
            scanf("%d",&choice);
```

```
if(choice==1){
   printf("Enter the new operational status: ");
   scanf("%s",&vehicles[i].status);
   printf("Updated successfully");
}
else if(choice==2){
   printf("Enter the vehicle type(Airplane,Helicopter,Drone): ");
   char vehicletype[50];
   scanf("%s",&vehicletype);
   if(strcmp(vehicletype,"Airplane")==0){
      printf("Enter new number of engines: ");
      scanf("%d",&vehicles[i].attributes.airplane);
      printf("Updated successfully");
   }
   else if(strcmp(vehicletype,"Helicopter")==0){
      printf("Enter new number of rotor blade: ");
      scanf("%d",&vehicles[i].attributes.helicopter);
      printf("Updated successfully");
   }
   else if(strcmp(vehicletype,"Drone")==0){
      printf("Enter new payload capacity: ");
      scanf("%f",&vehicles[i].attributes.drone);
      printf("Updated successfully");
   }
   else{
      printf("Wrong type of vehicle");
   }
}
else{
   printf("Invalid choice");
}
```

```
            break;
        }
        if(!found){
            printf("Vehicle not found");
        }
    }
}
void sort(Vehicle *vehicles, int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (vehicles[j].range < vehicles[j + 1].range) {
                Vehicle temp = vehicles[j];
                vehicles[j] = vehicles[j + 1];
                vehicles[j + 1] = temp;
            }
        }
    }
    printf("Vehicles sorted by range in descending order.\n");
}
```