ASSESMENT 2

/*Define Data Types

Vehicle Sensor Data Structure:

Create a structure Sensor to represent each sensor's data.

Fields:

sensorID (integer): Unique ID for the sensor.

sensorType (string): Type of sensor (e.g., LIDAR, GPS, Ultrasonic, Camera).

dataValue (float): Current reading of the sensor.

timestamp (string): Timestamp of the latest reading.

Union for Sensor Status:

Create a union SensorStatus to store either:

statusMessage (string): A text message describing the sensor's health status (e.g., "OK", "ERROR").

errorCode (integer): A numerical error code if the sensor fails.

2. Features to Implement

Dynamic Memory Allocation:

Allocate memory dynamically to store an array of Sensor structures based on user input (N sensors).

Input and Output:

Input the details of each sensor, including its type, data value, and status (either a message or error code).

Display the details of all sensors, including their status.

Sensor Analysis:

Identify and display the sensor with the highest dataValue.

Display all sensors of a particular type (e.g., all LIDAR sensors).

Update Sensor Data:

Update the dataValue of a sensor by searching for its sensorID.

Update the sensor status (either a new status message or error code).

Sorting:

Sort sensors by dataValue in descending order.

Typedef Usage:

Use typedef to create aliases for Sensor and SensorStatus to simplify the code.

Menu-Driven Interface:

Provide a user-friendly menu with options:

Add Sensor Data

Display All Sensors

Search for Sensor by ID

Update Sensor Data or Status

Sort Sensors by Data Value

Exit

*/

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef struct Sensor
{
  int sensorID ;//(integer): Unique ID for the sensor.
  char sensorType[10];// (string): Type of sensor (e.g., LIDAR, GPS, Ultrasonic, Camera).
  float dataValue;// (float): Current reading of the sensor.
  char timestamp[10];// (string):
}sensor;
typedef union SensorStatus
{
  char statusMessage[10];//(string): A text message describing the sensor's health status (e.g., "OK", "ERROR").
  int errorCode;// (integer)

}sensorstatus;
void inputsensor(sensor*,sensorstatus[],int);
void displaysensor(sensor*,sensorstatus[],int);
void searchsensor(sensor*,sensorstatus[],int);
void updatesensor(sensor*,sensorstatus[],int);
```

```
void sortsensor(sensor*,sensorstatus[],int);


int main()
{
   int n;
   int choice;
   printf("enter the number of sensors \n");
   scanf("%d",&n);
   sensorstatus status[n];
   sensor *sensordata;
   sensordata=(sensor*)malloc(n*sizeof(sensor));
   if(sensordata==NULL)
   {
      printf("memory not allocated \n");
      return 0;
   }
   else
   {
      printf("memory allocated \n");
   }
   while(1)
   {
      printf("1.Add Sensor Data\n");
      printf("2.Display All Sensors\n");
      printf("3.Search for Sensor by ID\n");
      printf("4.update Sensor Data or Status\n");
      printf("5.Sort Sensors by Data Value\n");
      printf("6.exit \n");
      printf("enter a choice \n");
      scanf("%d",&choice);
      switch (choice)
```

```
        {
        case 1:
            inputsensor(sensordata,status,n);
            break;
        case 2:
            displaysensor(sensordata,status,n);
            break;
        case 3:
            searchsensor(sensordata,status,n);
            break;
        case 4:
            updatesensor(sensordata,status,n);
            break;
        case 5:
            sortsensor(sensordata,status,n);
            break;
        case 6:
            return 0;
            break;



        default:
            break;
        }
    }
     free(sensordata);
    return 0;
}
void inputsensor(sensor*sensordata,sensorstatus status[],int n)
{
     for(int i=0;i<n;i++)
```

```
    {
      printf("enter sensorid \n");
      scanf("%d",&sensordata[i].sensorID);
      printf("enter sensor type(LIDAT ,GPS etc)\n");
      scanf("%s",sensordata[i].sensorType);
      printf("enter datavalue \n");
      scanf("%f",&sensordata[i].dataValue);
      printf("enter timestamp \n");
      scanf("%s",sensordata[i].timestamp);
      printf("enter status message \n");
      scanf("%s",status[i].statusMessage);
    }
    /*int sensorID ;//(integer): Unique ID for the sensor.
  char sensorType[10];// (string): Type of sensor (e.g., LIDAR, GPS, Ultrasonic, Camera).
  float dataValue;// (float): Current reading of the sensor.
  char timestamp[10];// (string):
}sensor;
typedef union SensorStatus
{
  char statusMessage[10];//(string): A text message describing the sensor's health status (e.g., "OK",
"ERROR").
  int errorCode;// (integer)*/

}
void displaysensor(sensor*sensordata,sensorstatus status[],int n)
{
   for(int i=0;i<n;i++)
   {
       printf("the sensorid is %d \n",sensordata[i].sensorID);
       printf("sensor type:%s \n",sensordata[i].sensorType);
       printf("datavalue :%f \n",sensordata[i].dataValue);
```

```
        printf("timestamp:%s \n",sensordata[i].timestamp);
        printf("status message: %s \n",status[i].statusMessage);

    }
}
void searchsensor(sensor*sensordata,sensorstatus status[],int n)
{
    int id;
    printf("enter an id \n");
    scanf("%d",&id);
    for(int i=0;i<n;i++)
    {
        if(id==sensordata[i].sensorID)
        {
            printf("id found \n");
            printf("the sensorid is %d \n",sensordata[i].sensorID);
            printf("sensor type:%s \n",sensordata[i].sensorType);
            printf("datavalue :%f \n",sensordata[i].dataValue);
            printf("timestamp:%s \n",sensordata[i].timestamp);
            printf("status message: %s \n",status[i].statusMessage);

        }
    }
}
void updatesensor(sensor*sensordata,sensorstatus status[],int n)
{
    int id;
    printf("enter the id of sensor you want to change status \n");
    scanf("%d",&id);
    char stat[10];
    printf("enter the status \n");
```

```
    scanf("%s",stat);
    for(int i=0;i<n;i++)
    {
        if(id==sensordata[i].sensorID)
        {
            strcpy(status[i].statusMessage,stat);
        }
    }
    for(int i=0;i<n;i++)
    {
        printf("the sensorid is %d \n",sensordata[i].sensorID);
        printf("sensor type:%s \n",sensordata[i].sensorType);
        printf("datavalue :%f \n",sensordata[i].dataValue);
        printf("timestamp:%s \n",sensordata[i].timestamp);
        printf("status message: %s \n",status[i].statusMessage);

    }

}
void sortsensor(sensor*sensordata,sensorstatus status[],int n)
{
    sensor temp;
    for(int i=0;i<n;i++)
    {
        for(int j=i+1;j<n;j++)
        {
            if(sensordata[i].dataValue<sensordata[j].dataValue)
            {
                temp=sensordata[i];
                sensordata[i]=sensordata[j];
                sensordata[j]=temp;
```

```
            }
        }


    }
    printf("after sorting \n");
    for(int i=0;i<n;i++)
    {
        printf("the sensorid is %d \n",sensordata[i].sensorID);
        printf("sensor type:%s \n",sensordata[i].sensorType);
        printf("datavalue :%f \n",sensordata[i].dataValue);
        printf("timestamp:%s \n",sensordata[i].timestamp);
        printf("status message: %s \n",status[i].statusMessage);
    }


}
```

```
PS D:\learning c> cd 'd:\learning c\output'
PS D:\learning c\output> & .\'qn17.exe'
enter the number of sensors
2
memory allocated
1.Add Sensor Data
2.Display All Sensors
3.Search for Sensor by ID
4.update Sensor Data or Status
5.Sort Sensors by Data Value
6.exit
enter a choice
1
enter sensorid
1
enter sensor type(LIDAT ,GPS etc)
LIDAT
enter datavalue
2
enter timestamp
1
enter status message
OK
enter sensorid
2
enter sensor type(LIDAT ,GPS etc)
GPS
enter datavalue
6
enter timestamp
1
enter status message
OK
1.Add Sensor Data
2.Display All Sensors
3.Search for Sensor by ID
4.update Sensor Data or Status
5.Sort Sensors by Data Value
6.exit
enter a choice
2
```

```
enter a choice
2
the sensorid is 1
sensor type:LIDAT
datavalue :2.000000
timestamp:1
status message: OK
the sensorid is 2
sensor type:GPS
datavalue :6.000000
timestamp:1
status message: OK
1.Add Sensor Data
2.Display All Sensors
3.Search for Sensor by ID
4.update Sensor Data or Status
5.Sort Sensors by Data Value
6.exit
enter a choice
3
enter an id
2
id found
the sensorid is 2
sensor type:GPS
datavalue :6.000000
timestamp:1
status message: OK
1.Add Sensor Data
2.Display All Sensors
3.Search for Sensor by ID
4.update Sensor Data or Status
5.Sort Sensors by Data Value
6.exit
enter a choice
4
enter the id of sensor you want to change status
2
enter the status
ERROR
```

```
2
enter the status
ERROR
the sensorid is 1
sensor type:LIDAT
datavalue :2.000000
timestamp:1
status message: OK
the sensorid is 2
sensor type:GPS
datavalue :6.000000
timestamp:1
status message: ERROR
1.Add Sensor Data
2.Display All Sensors
3.Search for Sensor by ID
4.update Sensor Data or Status
5.Sort Sensors by Data Value
6.exit
enter a choice
2
the sensorid is 1
sensor type:LIDAT
datavalue :2.000000
timestamp:1
status message: OK
the sensorid is 2
sensor type:GPS
datavalue :6.000000
timestamp:1
status message: ERROR
1.Add Sensor Data
2.Display All Sensors
3.Search for Sensor by ID
4.update Sensor Data or Status
5.Sort Sensors by Data Value
6.exit
enter a choice
5
after sorting
the sensorid is 2
```

```
after sorting
the sensorid is 2
sensor type:GPS
datavalue :6.000000
timestamp:1
status message: OK
the sensorid is 1
sensor type:LIDAT
datavalue :2.000000
timestamp:1
status message: ERROR
1.Add Sensor Data
2.Display All Sensors
3.Search for Sensor by ID
4.update Sensor Data or Status
5.Sort Sensors by Data Value
6.exit
enter a choice
6
PS D:\learning c\output>
```