Structure & File:

==========

1. Create a structure called Shape which has two components, length and width.
   Create a structure variable Rectangle and take its length and width as input from the user.
   Implement the two functions int findArea(struct Shape R) and int findPerimeter(struct Shape R).
   From the main function, call these two functions to get the area and perimeter of the rectangle.

```
struct Shape
{
        double length;
        double width;
};
```

```
#include <stdio.h>

struct shape{

   double length;
   double width;
};
```

```
int findArea(struct shape R){

   double area;

   area=(R.length*R.width);
   printf("\nThe area of the rectangle:%.3lf", area);
   return 0;
}

int findPerimeter(struct shape R){

   double perimeter;

   perimeter=2*(R.length+R.width);
   printf("\nThe perimeter of the rectangle:%.3lf", perimeter);
   return 0;
}

int main(){

   struct shape rectangle;

   printf("Enter the length of the rectangle:");
   scanf("%lf", & rectangle.length);

   printf("Enter the width of the rectangle:");
   scanf("%lf", & rectangle.width);

   findArea(rectangle);
```

```
    findPerimeter(rectangle);


    return 0;
}
```

2. Write a program to add two complex numbers using structure.

Create a structure called Complex with two components, real and imag.

Write a function that takes two structure variables as input, then sum up the two complex number.

```
struct Complex
{
        float real;
        float imag;
};
struct Complex add(struct Complex n1, struct complex n2);
```

```
#include <stdio.h>

struct complex{

    float real;
    float imag;
};

struct complex add(struct complex n1 , struct complex n2);
```

```
int main(){

   struct complex num_1 , num_2 , result;

   printf("Enter the real part of the 1st complex number:");
   scanf("%f", & num_1.real);
   printf("Enter the imag part of the 1st complex number:");
   scanf("%f", & num_1.imag);

   printf("Enter the real part of the 1st complex number:");
   scanf("%f", & num_2.real);
   printf("Enter the imag part of the 1st complex number:");
   scanf("%f", & num_2.imag);

   result=add(num_1 , num_2);

   printf("\nThe sum of two complex number is:");
   printf("\n\tSum= %.2f + %.2fi", result.real , result.imag);

   return 0;
}

struct complex add(struct complex n1 , struct complex n2){

   struct complex add;

   add.real=n1.real+n2.real;
   add.imag=n1.imag+n2.imag;
```

```
   return add;
}
```

3. Define a structure named MovieStar which will have the following elements:

Name (string),Rating(float), TotalFans(int). Declare a structure array of MovieStar for 5 movie stars.

Now take N user reviews as input. Each review will consist of a Movie star name and his rating by a new fan.

Now adjust each Movie Star's rating according to the reviews and show the results.

Rating of a movie star is the average rating given by fans those who rated him.

```
#include<stdio.h>

struct Moviestar{
   char name[100];
   float rating;
   int TotalFans;
};

int main(){
   int n,i,j,sum;
   struct Moviestar num[5];

   printf("Enter the number of user review you want:");
   scanf("%d",&n);
```

```
   for(i=0 ; i<5 ; i++){
   printf("enter the name:");
   fflush(stdin);
   scanf("%[^\n]" ,num[i].name);
   printf("enter the Total fan:");
   scanf("%d", &num[i].TotalFans);

   for(j=0 ; j<n ; j++){
      printf("Review for star name:%s fan num %d:",num[i].name, j+1);
      scanf("%f", &num[j].rating);
   }
   sum=0;

   for(j=0 ; j<n ; j++){
      sum=num[j].rating+sum;
   }
    num[i].rating=sum/(float)n;
    printf("Movie star name is %s",num[i].name);
    printf("\n %s rating is %0.2f",num[i].name,num[i].rating);
    printf(" \ntotal number of fans %s has %d\n",num[i].name,num[i].TotalFans);
    printf("\n");

}

return 0;

}
```

4.Define a structure named Gamer which will have the following elements:

Number_of_ favorite_games (int), List_of_favorite_games (2D string). Now declare a structure array of Gamer

for 5 gamers and take inputs for them. Now generate a rank list of the games.

(Hint: The game which appeared most in the favorite games list will be the top game. In case of tie, print the game which comes alphabetically before).

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

struct Gamer{
   int Number_of_favorite_games;
   char **List_of_favorite_games;
};

int main(){

   struct Gamer gamer[5];
   int i,counter;
   int Total_number_of_games = 0;
   char games_list[Total_number_of_games][100];
   int times[Total_number_of_games],length=0,k,j;

   for(i=0; i<5; i++){

      printf("Enter the number of favorite games of Gamer %d : ",i+1);
      scanf("%d",&gamer[i].Number_of_favorite_games);
```

```
    gamer[i].List_of_favorite_games = (char
*)malloc(gamer[i].Number_of_favorite_games*sizeof(char));
    printf("Enter the list of favorite games of Gamer %d :\n",i+1);
    for(counter=0; counter<gamer[i].Number_of_favorite_games; counter++)
    {
      gamer[i].List_of_favorite_games[counter] = (char *)malloc(100*sizeof(char));
      scanf("%s",gamer[i].List_of_favorite_games[counter]);
    }
  }


  for(i=0; i<5; i++)
  {
    Total_number_of_games += gamer[i].Number_of_favorite_games;
  }


  for(i=0; i<5; i++)
  {
    for(counter=0; counter<gamer[i].Number_of_favorite_games; counter++)
    {
      strcpy(games_list[length],gamer[i].List_of_favorite_games[counter]);
      times[length]=1;
      length++;
    }
  }

  for(k=0; k<Total_number_of_games-1; k++)
```

```
{
   for(j=k+1; j<Total_number_of_games; j++)
   {
      if(strcmp(games_list[k],games_list[j])==0 && times[k]!=0)
      {
         times[k]++;
         times[j] = 0;
      }
   }
}

int t;
char temp[100];

for(k=0; k<Total_number_of_games-1; k++)
{
   for(j=k+1; j<Total_number_of_games; j++)
   {
      if(times[k]<times[j])
      {
         t = times[k];
         times[k] = times[j];
         times[j] = t;
         strcpy(temp,games_list[k]);
         strcpy(games_list[k],games_list[j]);
         strcpy(games_list[j],temp);
      }
      else if(times[k]==times[j])
      {
```

```
            if(strcmp(games_list[k],games_list[j])>0)
            {
               strcpy(temp,games_list[k]);
               strcpy(games_list[k],games_list[j]);
               strcpy(games_list[j],temp);
            }
         }
      }
   }

   printf("\nRank List of Games\n");
   for(i=0; i<Total_number_of_games; i++)
   {
      if(times[i]!=0)
      {
         printf("%d - %s\n",i+1,games_list[i]);
      }
   }
   return 0;
}
```

5.Create a structure called BarcelonaPlayer with the following members.

```
struct BarcelonaPlayer
{
char name[20];
int age;
char country[20];
char Position[20];
double Salary;
double Rating;
};
```

First, create an array of BarcelonaPlayer structures. Now, write a function that takes an array of BarcelonaPlayer

structures as input and find out the highest paid player among all the players.

void highestPaidPlayer(struct BarcelonaPlayer *pl, int size);

Create another function that finds all the players from Argentina.

```
#include<stdio.h>
#include<string.h>

struct BarcelonaPlayer{
   char name[20];
   int age;
   char country[20];
   char Position[20];
   double Salary;
```

```
    double Rating;
};

void highestPaidPlayer(struct BarcelonaPlayer *pl, int size);
void findPlayers(struct BarcelonaPlayer *pl, int size);

int main()
{
    int n,i;
    printf("Enter the number of player details you want to enter ");
    scanf("%d",&n);
    struct BarcelonaPlayer pl[n];
    printf("Enter the details of %d barcelona players\n",n);
    for(i=0; i<n; i++)
    {
        printf("Enter name of player %d\n",i+1);
        scanf("%s",pl[i].name);
        fflush(stdin);
        printf("Enter age of player %d\n",i+1);
        scanf("%d",&pl[i].age);
        printf("Enter country of player %d\n",i+1);
        scanf("%s",pl[i].country);
        printf("Enter position of player %d\n",i+1);
        scanf("%s",pl[i].Position);
        printf("Enter salary of player %d\n",i+1);
        scanf("%lf",&pl[i].Salary);
        printf("Enter Rating of player %d\n",i+1);
        scanf("%lf",&pl[i].Rating);
    }
```

```
   highestPaidPlayer(pl,n);
   findPlayers(pl,n);

   return 0;
}



void highestPaidPlayer(struct BarcelonaPlayer *pl, int size)
{
   int i;
   double high = pl[0].Salary;
   for(i=0; i<size; i++)
   {
      if(pl[i].Salary>high)
      {
         high=pl[i].Salary;
      }
   }
   for(i=0; i<size; i++)
   {
      if(pl[i].Salary==high)
      {
         printf("Details of highest Paid
Player\n%s\n%d\n%s\n%s\n%lf\n%lf",pl[i].name,pl[i].age,pl[i].country,pl[i].Position,pl[i].Salary,pl[i].Rati
ng);
      }
   }
}
void findPlayers(struct BarcelonaPlayer *pl, int size)
```

```
{
   int i;
   printf("\nDetails of Players from Argentina:");
   for(i=0; i<size; i++)
   {
      if (strcmp(pl[i].country,"Argentina") == 0)
      {

printf("\n%s\n%d\n%s\n%s\n%lf\n%lf",pl[i].name,pl[i].age,pl[i].country,pl[i].Position,pl[i].Salary,pl[i].Rat
ing);
      }
   }
}
```

6. Struct employee{

Int id;

Char name[30];

Int age;

Float salary;

};

Using the given structure , write a c program that asks for ten employee's name, id, age and salary from the user ,

then, it writes the data in file named out.txt.(Sir final).

```
#include <stdio.h>

struct employee{
   int id;
   char name[30];
   int age;
   float salary;
};

int main(){

   struct employee num[10];
   int i;

   for(i=0 ; i<10 ; i++){

      printf("Enter ID:");
      scanf("%d", & num[i].id);
      printf("Enter the employee name:");
      scanf("%s", num[i].name);
      printf("Enter age of the employee:");
      scanf("%d", & num[i].age);
      printf("Enter salary:");
      scanf("%f", & num[i].salary);
   }

   FILE *fp;
```

```
    fp=fopen("out.txt", "w");

    for(i=0 ; i<10 ; i++){
        fprintf(fp, "%d\t",num[i].id);
        fprintf(fp, "%s\t",num[i].name);
        fprintf(fp, "%d\t",num[i].age);
        fprintf(fp, "%f\t",num[i].salary);
    }

    fclose(fp);

    return 0;
}
```

7. For the same structure of no 6, read the contents of the file out.txt

and print the name of the highest salaried employee and the youngest employee names in the output screen.(Sir final)

```
#include <stdio.h>
#include <string.h>

struct employee{
   int id;
   char name[30];
   int age;
   float salary;
};

int main(){

   struct employee num[10];
   int i;
   FILE *fp;

   fp=fopen("out.txt", "r");

   for(i=0 ; i<10 ; i++){

      fscanf(fp, "%d", &num[i].id);
      fscanf(fp, "%s", num[i].name);
      fscanf(fp, "%d", &num[i].age);
```

```
    fscanf(fp, "%f", &num[i].salary);
  }

  fclose(fp);

  float max_salary=num[0].salary;
  int min_age=num[0].age;
  char maxEmployee[20];
  char youngEmployee[20];

  for(i=0 ; i<10 ; i++){
    if(num[i].salary>max_salary){
      max_salary=num[i].salary;
      strcpy(maxEmployee,num[i].name);
    }
  }

  for(i=0 ; i<10 ; i++){
    if(num[i].age<min_age){
      min_age=num[i].age;
      strcpy(youngEmployee,num[i].name);
    }
  }

*****// /*  if you want to use strcpy
        firstly find out the max_salary & min_age
        then=>
        for(i=0 ; i<10 ; i++){
              if(num[i].salary==max_salary)
```

```
                printf("\nThe highest salaried employee:%s",maxEmployee);
            }
        for(i=0 ; i<10 ; i++){
                if(num[i].age==max_age)
                printf("\nThe youngest employee:%s", youngEmployee);
            } */  //*****


    printf("\nThe highest salaried employee:%s",maxEmployee);
    printf("\nThe youngest employee:%s", youngEmployee);

    return 0;
}
```

8.Write a program that reads a file in.txt, find out how many words in that file.

```
#include <stdio.h>

int main(){

    char str[30];
    int count=0;
    FILE *fp;

    fp=fopen("in.txt" , "r");
```

```
    while(!feof(fp)){

        fscanf(fp , "%s" , str);

        if(feof(fp))
            break;

        printf("%s\n", str);
        count++;
    }

    fclose(fp);
    printf("Total words:%d", count);
    return 0;
}
```

9.Write a program to read a file if there is any word “NSU” or not. If there is not append “NSU” at end of the file.

```
#include <stdio.h>
#include <string.h>

int main(){

    char str[20];
    int flag=0;
```

```
FILE *fp;

fp=fopen("out.txt" , "r+");

while(!feof(fp)){

   fscanf(fp , "%s" , str);

   if(feof(fp))
      break;

   else if(stricmp(str,"NSU")==0){
      flag=1;
      break;
   }
}

if(flag==1){
   printf("The word NSU has found in the file.");
}
else{
   printf("NSU is not found.....Appending NSU word at the end of the file.....");

   fseek(fp , 0 , SEEK_END);
   fprintf(fp ,"NSU");
}

fclose(fp);
```

```
    return 0;
}
```

10.Write a program to count all the occurrences of “NSU” in a text file. Produce another output text file to write your output.

```
#include <stdio.h>
#include <string.h>

int main(){

    char str[20];
    int count=0;
    FILE *fp;
    FILE *fx;

    fp=fopen("out.txt" , "r");

    while(!feof(fp)){

        fscanf(fp , "%s" , str);
```

```
        if(feof(fp))
            break;

        else if(stricmp(str,"NSU")==0){
            count++;
            fx=fopen("in.txt" , "w");

            fprintf(fx , "Total occurrences of NSU:%d", count);

            fclose(fx);
        }
    }

    fclose(fp);

    return 0;
}
```

11.Write a program that counts line numbers from a file.

```
#include <stdio.h>

int main(){

    char str[200];
```

```
    int count=0;
    FILE *fp;

    fp=fopen("out.txt" , "r");

    while(!feof(fp)){

        fgets(str , 100 , fp);

        if(feof(fp))
            break;
        count++;
    }

    fclose(fp);

    printf("Total lines:%d", count);
}
```

12. Write a program that reads a file of some integer numbers and counts total even, odd numbers separately in that file and prints all the even, odd numbers.

```
#include <stdio.h>

int main(){

   int num,i=0,j=0,k;
   int even[20],odd[20];
   FILE *fp;

   fp=fopen("out.txt" , "r");

   while(!feof(fp)){

      fscanf(fp , "%d" , &num);

      if(feof(fp))
         break;

      else if(num%2==0){
         even[i]=num;
         i++;
      }
```

```
        else if(num%2!=0){
            odd[j]=num;
            j++;
        }

    }

    fclose(fp);

    printf("All even numbers:");
    for(k=0 ; k<i ; k++){
        printf("%d , ",even[k]);
    }

    printf("\nAll odd numbers:");
    for(k=0 ; k<j ; k++){
        printf("%d , ",odd[k]);
    }

    return 0;
}
```

13. Consider you have an input file that have the following product information described by the structure as follows:

```
struct product {
        char name[20];
        int ID;
        int quantity;
        float price; }
```

Write down a C program to take input from the in.txt file and compute the total price of all the products and the least cost product name in the store.

```
#include <stdio.h>

struct product{

   char name[20];
   int ID;
   int quantity;
   float price;

};

int main(){

   struct product num[20];
```

```
FILE *fp;
int n,i;
float total_price=0;

fp=fopen("out.txt" , "r");

fscanf(fp , "%d" , &n);

for(i=0 ; i<n ; i++){
   fscanf(fp , "%s" , num[i].name);
   fscanf(fp , "%d" , &num[i].ID);
   fscanf(fp , "%d" , &num[i].quantity);
   fscanf(fp , "%f" , &num[i].price);

   total_price=total_price+num[i].price;
}

fclose(fp);

float least_pro=num[0].price;

for(i=0 ; i<n ; i++){
   if(num[i].price<least_pro){
      least_pro=num[i].price;
   }
}

printf("Total price of all products :%.2f\n", total_price);
```

```
    for(i=0 ; i<n ; i++){
        if(num[i].price==least_pro)
            printf("Least cost item name:%s", num[i].name);
    }

    return 0;
}
```