Name: P. SHIVAM
Roll NO: 2203A51386

ASSIGNMENT

Question 1 : Structures

A Define a struture named "Employee" with the following members: name (string), id (integer), salary (float). Declare a variable of the structure type and initialize its members. Print the details of the employee.

CODE:

```
#include<stdio.h>
//Define the Employee structure
Struct Employee {
    Char name[100];
    int id;
   float salary;
};
int main() {
  // Deciare a variable of the Employee structure
  Struct Employee emp;
  // Input the details of the employee
  Printf ("Enter the name of the employee:");
  fgets (emp.name, sizeof (emp.name), stdin);

  Printf ("Enter the ID of the employee:");
  Scanf ("%d", &empid);

  Printf ("Enter the salary of the employee:");
  Scanf ("%f ", & emp.salary);

  // Print the details of the employee
   Printf ("\nEmployee Details:\n");
   Printf ("Name: %s", emp.name);
   Printf ("ID: %d\n", emp.id);
   Printf ("Salary: %.2f\n", emp.salary);

   return 0;
}
```

B. write a c program to calculate the total salary of a group of employees using an array of structures. Allow the user to input the employees and their details. Display the total salary.

CODE:

```
#include<stdio.h>
// creating a employee structure
 struct Employee
{
   Char name[50];
   int id;
   double salary;
};
void main(){
 int n,i;
 double total_salary=0;
 Printf("Enter the number of employees:"); //entering
   no.of employees
 Scanf("%d",&n);
 struct Employee e1[n];//creating employee structure
    variable with size "n"
  for (i=0;i<n;i++)
{
 // asking user to enter the values of employee structure
 Printf("\nEnter details of employee %d:\n",i+1);
 Printf("Name:");
 Scanf("%s",e1[i].name);
 Printf("ID:");
 Scanf("%d",&e1[i].id);
 Printf("Salary:");
 Scanf("%lf",&e1[i].salary);
```

```
total_salary += e1[i].salary; // Adding the salaries
}
Printf("\n Total salary of all employees: %.2f\n",
 total_salary); //displaying the total salary
  return 0;
}
```

Question 2: Nested structures

A. Define a structure named "Address" with members: street ~~(street~~ (string), city (string), and pin (integer). Declare another structure named "person" with members: name (string) and address (of type Address). create a variable of the type person and initialize its members. Print the person's name and address details.

CODE:

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

Struct Address
{
    Char street[50];
    Char city[50];
     int pin;
};
  Struct person
{
  Char name[50];
  Struct Address a;
};
int main()
```

```
{
    struct Person a1 = {"Shivam", {"1931 noivida
    street", "berlin", 1002983};
    Printf ("Name: %.s \n", a1.name);
    Printf ("Address: %s, %s - %d\n", a1.a.street, a1.a.city,
            a1.a.pin);
    return 0;
}
```

B. Extend the above program to input details of multiple persons and their addresses using an array of structures. Print the details of all persons.

CODE:

```
//2.B
# include <stdio.h>
# define MAX_PERSONS 10
// Address structure
struct Address{
    Char street [100];
    char city [100];
    int pin;
};
// person structure
struct person{
    Char name [100];
    struct Address address;
};
int main(){
```

```
struct Person people[MAX_PERSONS];
int numPersons;

Printf("Enter the number of persons:");
Scanf("%d", &numPersons);
// Input details for each person
for(int i=0; i<numPersons;i++){
    Printf("Enter details for person %d:\n", i+1);
    Printf("Name:");
    Scanf("%s", people[i].name);
    Printf("Street:");
    Scanf("%s", people[i].address.street);
    Printf("City:");
    Scanf("%s", people[i].address.city);
    Printf("PIN:");
    Scanf("%d", &people[i].address.pin);
    Printf("\n");
}

//Print details of all persons
Printf("Details of all persons:\n");
for(int i=0; i<numPersons; i++){
Printf("Name:%s\n", people[i].name);
Printf("Address:%s, %s, PIN:%d\n", people[i].address.
street, people[i].address, city, people[i].address.
pin);
Printf("\n");
}
return 0;
}
```

Question 3: Array of structures

A. Define a structure named "Book" with members: title (string), author (string), Price (double) and rating (float). Declare an array of structures to store information about multiple books. Allow the user to input the details of the books and display them.

CODE:

```
#include <stdio.h>
#define MAX_BOOKS 10
// Book structure
struct Book {
    char title[100];
    char author[100];
    double price;
    float rating;
};
int main()
    struct Book books[MAX_BOOKS];
    int numBooks;

    Printf ("Enter the number of books:");
    Scanf ("%d", &numBooks);

    // Input details for each book
    for (int i=0; i<numBooks; i++) {

    Printf ("Enter details for Book %d:\n", i+1);
    Printf ("Title:");
    Scanf ("%s", books [i].title);
    Printf ("Author:");
    Scanf ("%s", books[i].author);
    Printf ("Price:");
    Scanf ("%.1f", &books[i].price);
    printf ("Rating:");
    .
```

```
    scanf("%f", &books[i].rating);
    printf("\n");
}
printf("Details of all books:\n");
for(int i=0; i<numBooks; i++){
    printf("Book %d:\n", i+1);
    printf("Title: %s\n", book[i].title);
    printf("Author: %s\n", books[i].author);
    printf("Price: %.2f\n", books[i].price);
    printf("Rating: %.1f\n", book.s[i].rating);
    printf("\n");
}
return 0;
}
```

B. Write a c program to find the book with the highest rating and lowest price from the array of structures.

CODE:

```
#include <stdio.h>
#include <string.h>
struct Book //Defining BOOK structure
{
  Char Title[50];
  Char Author[50];
  double Price;
  float Rating;
};
  Void main()
```

```
{
  int i,n,x;
  Struct Book b1[n]; // Creating book variable
  Printf("Enter no.of Books:"); // Entering no.of books
  Scanf("%d",& x);
  for(i=0;i<x;i++) // Using for loop to enter multiple
                                          inputs.
  {
    // Asking User to input values
    Printf("\n Enter %d Book details", i+1);
    Printf("\n Enter Title:");
    Scanf("%.s", b1[i].Title);
    Printf("Enter Author:");
    Scanf("%.s",b1[i].Author);
    Printf("Enter price:");
    Scanf("%.1f",&b1[i].Price);
    Printf("Enter Rating:");
    Scanf("%.f",& b1[i].Rating);
  }
for(i=0;i<x;i++)
 {
     if(b1[i].Price<500 && b1[i].Rating>4.0)
   {
     Printf("\n <<<< %d Book details >>>>\n",i+1);
     Printf("\n Book-Title |Author |Price |Rating:");
     Printf("\n %s %s %-7.1f %-8.1f\n",b1[i].Title,
      b1[i].Author, b1[i].Price,b1[i].Rating);
   }
  else
  {
     Printf("\n Book NOT FOUND");
  }
}
```

Question 4 : Pointers to structures

What is the difference between a pointer to a structure and a pointer to an array in c Programming?

1. Pointer to a structure :-

- A pointer to a structure is a pointer that points to an instance of a structure. structuress in c allows you to define custom data types that contain multiple variables of different types. when you declare a pointer to a structure, you can use it to access the members (variables) of the structure using the arrow operator ('->').

You can allocate memory for a structure dynamically using functions like malloc() or create a structure variable and take its address using the address-of operator('&').

2. Pointer to an Array.

- A pointer to an array is a pointer that points to the first element of an array.
- Arrays in c are contiguous blocks of memory that store elements of the same type.

when you declare a pointer to an array, you can use pointer arithmetic to access individual elements of the array.

The pointer to an array can be assigned to address of the first element of statically declared array or obtained by using the address-of operator with the array variable's name.

Question 5: Pointers to structures

A. Define a structure named "student" with members: name (string), roll (integer), and marks (float). Create a pointer to the structure and dynamically allocate memory for it. Input the details of the student using the pointer and display them.

CODE:

```
#include <stdio.h>
  Struct student {
      Char name[20];
       int roll;
      float marks;
   };
  Void main()
  {
    Struct students;
    Strut student *ps;
    ps=&s;
 //Print("\n student -name\troll no\tmarks:\n");
  Print("\n_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ ");
  Printf("\n%-15s%-7d%.1f",ps->name,ps->roll,ps->marks);
}
```

B. write a c program to sort an array of structures using pointers. sort the array based on roll numbers in ascending order and display the sorted list.

CODE:

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

```
struct student {
 int roll;
 Char name[20];
};

 int compare (const void* a, const void *b)
{
  Const struct student * s1 = *(const struct student **);
  Const struct student *s2 = *(const struct student **);
 return s1 -> roll - s2 -> roll;
}

int main():
{
  int n;
  Printf ("Enter the number of students:");
  Scanf ("%d", & n);

   Struct student * students[n];
   for (int i=0; i<n; i++){
     students[i]: malloc(sizeof(struct student));
   Printf ("Enter roll and name of student %d:", i+1);
   scanf ("%d %s", & students[i]-> roll, students[i] -> name);
   }
   qsort (students, n, sizeof (struct student*), compare);
   Printf ("Sorted list based on roll number in order:\n");
   for (int i=0; i<n; i++)
   {
     Printf ("%d %s\n", students [i]-> roll, students[i] -> name);
      free (students[i]);
   }
  return 0;
}
```

Question 6: Files

Explain the difference between text files and binary files in C programming.

Text File:

* It contains alphabets and numbers which are easily are understood by humans.
* An error in a text file can be eliminated when seen.
* In text file, the text and characters will store one char per byte.
* For example, the integer value 12567 will occupy 2 bytes in memory, but, it will occupy 5 bytes in text file.
* The data format is usually line-oriented. Here, each line a separate command.

Binary file:

* It contains 1's and 0's, which are easily understood by computers
* The error in a binary file corrupts the file and is not easy to detect.
* In binary file, the integer value 12567 will occupy 2 bytes in memory and in file.
* A binary file always matching software to read or write it.
* For example, an MP3 file can be produced by a a sound recorder or audio editor, and it can be played in a music player
* MP3 file will not play in an image viewer or data-base software.

Question 7: Files and Random File Access

A. Write a C program to create a text file named "data.txt" and allow the user to enter text. Append the entered text to the file.

CODE:

```
#include<stdio.h>
#include<stdlib.h>
#define MAX_LINE_LEN 1024
int main(){
  char data[MAX_LINE_LEN];
  Char file_name[]="data.txt";
  FILE *fp;
  /*Take user input*/
  Printf("Enter text to append: ");
  fgets(data, MAX_LINE_LEN, stdin);
  fp=fopen(file_name, "a");
  if(fp==NULL){
    Printf("Error opening file!");
    return 1;
  }
  fprintf(fp, "%s", data);
  fclose(fp)
    Printf("Data appended successfully to file %s.\n",
           file_name);
    return 0;
}
```

B. write a c program to read the contents of a text file named "data.txt" and display them on the console.

CODE:

```
#include <stdio.h>
int main() {
FILE *fp;
 Char ch;
fp = fopen ("data.txt", "r");
   if (fp == NULL){
 Printf ("Error Openingfile.");
 return 1;
}
 Printf ("contents of data.txt:\n\n");
 while ((ch = getc(fp)) != EOF)
     Putchar (ch);
 fclose (fp);
 return 0; }
```

C. write a C program to create a binary file named "students.dat" to store information about students (name, roll, and marks). Allow the user to input to input the details of multiple students and write them to the file.

CODE:

```
#include <stdio.h>
#include <stdlib.h>
struct student{
  char name[50];
  int roll;
  float marks;
};
int main(){
   int n;
   printf("Enter the number of students:");
   scanf("%d",&n);
   struct student s[n];
  FILE *fp;
  fp=fopen("students.dat","wb");
  if (fp==NULL){
    printf("Error opening file");
    exit(1);
  }
  for(int i=0;i<n;i++){
    printf("\n Enter details of student %d:\n",i+1);
    printf("Enter name:");
    fflush(stdin);scanf("%s",s[i].name);
    printf("Enter roll number:");
    scanf("%d",&s[i].roll);
    printf("Enter marks:");
    scanf("%f",&s[i].marks);
    fwrite(&s[i],sizeof(struct student),1,fp);
```

```
}
fclose(fp);
    printf("\n Data written to file successfully.\n");
    return 0;
}
```

D. write a c program to randomly access and read the details of a student from the binary file "Students.dat" based on the roll number entered by the user.

CODE:

```
#include<stdio.h>
struct student{
    int roll_number;
    char name[50];
    float name[50];
};
int main() {
    FILE *file;
     struct students;
    int roll_number, found=0;
    printf("Enter the roll number of the student to search:");
    scanf("%d", &roll_number);
    file=fopen("students.dat", "rb");
    if(file==NULL)
```

```
{
  Printf("Error opening file.");
  return 1;
}
while (fread(&s, sizeof(struct student), 1, file)){
if (s.roll_number == roll_number)
{
  found = 1;
  break;
}
}
  if (found)
{
  Printf("Roll Number: %d \n", s.roll_number);
  Printf("Name: %s\n", s.name);
  Printf("Marks: %.2f \n", s.marks);
} else {
  Printf("Student not found \n");
}
fclose(file);
return 0;
}
```

Question 8: files and Random File Access

Create a C program to implement a file management system with CRUD (Create, Read, Update, Delete) Operations.

CODE:-

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

struct student
{
 int id;
 char name[50];
 int age;
float percentage;
};
 void create()
 {
  FILE *fp;
  struct students;
  fp=fopen("students.dat","ab");
  printf("\nEnter student details:\n");
  printf("ID: ");
  scanf("%d",&s.id);
  printf("Name:");
```

```
  scanf("%s", s.name);
  printf("Age:");
  scanf("%d", &s.age);
  printf("Percentage:");
  scanf("%f", &s.percentage);
 fwrite(&s, sizeof(s), 1, fp);
fclose(fp);
}
 void read()
{
  FILE *fp;
 struct student s;
  fp=fopen("students.dat", "rb");
   printf("\n student details:\n");
  while (fread(&s, sizeof(s), 1, fp)==1)
  {
    printf("ID:%d\n", s.id);
    printf("Name :%s\n", s.name);
    printf("Age:%d \n", s.age);
     printf("Percentage:%.2f \n", s.percentage);
     printf("\n");
   }
 fclose(fp);
}
```

```
void updata()
{
 FILE *fp;
 struct students;
 int id, found=0;
fp=fopen("students.dat","rb+");
printf("\nEnter student ID to update:");
scanf("%d",&id);
while(fread(&s,sizeof(s),1,fp)==1)
{
  if(s.id==id)
  {
   found=1;
   printf("\n Enter new student details:\n");
    printf("ID:");
   scanf("%d",&s.id);
   printf("Name:");
   scanf("%s",s.name);
   printf("Age:");
   scanf("%d",&s.age);
   printf("percentage:");
   scanf("%f",&s.percentage);
  fseek(fp,-sizeof(s),SEEK_CUR);
  fwrite(&s,sizeof(s),1,fp);
   break;
  }
}
```

```
  if (!found)
  {
    Printf ("\n student not found.\n");
  }
  fclose (fp);
}
 Void delete()
{
  FILE *fp,*temp;
  struct student s;
  int id,found=0;
 fp=fopen ("temp.dat","wb");
  Printf("\n Enter student ID to delete:");
  scanf("%d",&id);
  while (fread(&s, sizeof(s), 1,fp)==1)
  {
   if (s.id != id)
    {
    fwrite (&s,sizeof(s),1,temp);
 }else
  {
    found=1;
  }
 }
  if (!found)
    {
     Printf ("\n student not found.\n");
  }
```

```
    else
    {
     Printf("\nStudent deleted successfully.\n");
    }
  fclose(fp);
  fclose(temp);
  remove("students.dat");
  rename("temp.dat", "students.dat");
}
int main()
{
 int choice;
 while(1)
 {
  printf("\n File Management System\n");
  Printf("-----------\n");
  Printf("1. create\n");
  Printf("2. Read\n");
  Printf("3. Update\n");
  Printf("4. Delete\n");
  Printf("5. Exit\n");
  Printf("Enter your choice:");
  Scanf("%d", &choice);
   switch(choice)
  {
   case 1:
      create();
      break;
```

```
Case 2:
  read();
  break;
Case 3:
   Update();
   break;
Case 4:
    delete();
    break;
Case 5:
  exit(0);
default:
      Printf("\nInvalid choice.\n");
   }
}
  return 0;
}
```

Question 9: Enumerations

A. Define an enumeration named "Month" with constants representing the months of the year. Write a program to input a month number from the user and display the corresponding month name using the enumeration.

CODE:

```
#include <stdio.h>
 enum Month
{
  JANUARY= 1, FEBRUARY, MARCH, APRIL, MAY, JUNE,
  JULY, AUGUST, SEPTEMBER, OCTOBER, NOVEMBER,
  DECEMBER.
};
int main()
{
 int monthNum;
 Printf("Enter a month number (1-12):");
 scanf("%d", &monthNum);
 switch (monthNUm)
 {
  Case JANUARY:
  Printf("January\n");
    break;
 Case FEBRUARY:
```

```
Printf("February\n");
break;
Case MARCH:
Printf("March\n");
break;
Case APRIL;
Printf("April\n");
break;
case MAY;
Printf("may\n");
break;
Case JUNE:
Printf("June\n");
break;
Case AUGUST JULY:
Printf("JULY\n");
break;
Case AUGUST:
Printf("August\n");
break;
Case OCTOBER:
Printf("October\n");
break;
Case NOVEMBER:
Printf("November\n");
break;
```

```
    Case DECEMBER:
     Printf("December\n");
      break;
    default;
        printf("Invalid month number\n");
       break;
  }
  return 0;
}
```

B. Extend the above program to print the number of days in the entered month.

CODE:

```
#include<stdio.h>
enum Month
{
January= 1, February, March, April, May, JUNE,
 July, August, september, October, November,
  December
};
 int main()
{
  int month_number
```

```
printf("enter a month number(1-12):");
scanf("%d", &month_number);
switch (month_number)
{
    case January:
    printf("January has 31 days.\n");
    break;
    case February:
        printf("February has 29 days.\n");
        break;
    case March:
        printf("March has 31 days.\n");
        break;
    case April:
        printf("April has 30 days.\n");
        break;
    case May:
        printf("May has 31 days.\n");
        break;
    case June:
        printf("June has 30 days.\n");
        break;
    case July:
        printf("July has 31 days.\n");
        break;
```

```
Case August:
    Printf ("August has 31 days.\n");
    break;
Case September:
    Printf ("September has 30 days.\n");
    break;
Case October:
    Printf ("October has 31 days.\n");
    break;
Case November:
    Printf ("November has 30 days.\n");
    break;
Case December
    Printf (December has 31 days.\n");
    break;
  default:
     Printf ("Invalid month number.\n");
     break;
  }
  return 0;
}
```

Question 10: Command Line Parameters

Write a C program that takes two command line arguments: an integer and a filename. The program should read the contents of the given file and print the first 'n' lines, where 'n' is the integer provided as a command line argument.

CODE:

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_LINE_LEN 1000
int main(int argc, char* argv[])
{
  if (argc != 3)
  {
    printf("Usage: %s <n> <filename>\n", argv[0]);
    return 1;
  }

  int n = atoi(argv[1]);
  if (n <= 0)
  {
    Printf("n must be a positive integer\n");
    return 1;
  }
```

```
FILE *fp = fopen(argv[2], "r");
  if(fp==NULL)
 {
  printf("could not open file: %s\n", argv[2]);
 return 1;
}
   Char line[MAX_LINE_LEN];
for(int i=0;i<n&&fgets(line, MAX_LINE_LEN, fp)!
  !=NULL;i++)
 {
  printf("%s", line);
}
fclose(fp);
return 0;
}
```