**Problem Statement:**

**Write a program that defines a custom data type Complex using typedef to represent a complex number with real and imaginary parts. Implement functions to:**

**Add two complex numbers.**

**Multiply two complex numbers.**

**Display a complex number in the format "a + bi".**

**Input Example**

**Enter first complex number (real and imaginary): 3 4**

**Enter second complex number (real and imaginary): 1 2**

**Output Example**

**Sum: 4 + 6i**

**Product: -5 + 10i**

```
#include<stdio.h>
typedef int Complex;
void sum(Complex,Complex,Complex,Complex);
void product(Complex,Complex,Complex,Complex);
int main(){
   Complex real1,real2,img1,img2;
   printf("Enter First complex number(real and imaginary)\n");
   scanf("%d%d",&real1,&img1);
   printf("Enter second complex number(real and imaginary)\n");
   scanf("%d%d",&real2,&img2);
   sum(real1,img1,real2,img2);
   product(real1,img1,real2,img2);
   return 0;

}

void sum(Complex r1,Complex i1,Complex r2,Complex i2){
   printf("sum = %d + %d i",r1+r2,i1+i2);
```

```
}

void product(Complex r1,Complex i1,Complex r2,Complex i2){
   printf(" product = %d + %d i",(r1*r2-i1*i2),(r1*i2+i1*r2));
}
```

**Typedef for Structures**

**Problem Statement:**

**Define a custom data type Rectangle using typedef to represent a rectangle with width and height as float values. Write functions to:**

**Compute the area of a rectangle.**

**Compute the perimeter of a rectangle.**

**Input Example:**

**Enter width and height of the rectangle: 5 10**

**Output Example:**

**Area: 50.00**

**Perimeter: 30.00**

```
#include<stdio.h>
typedef struct rect{
   float height;
   float width;
}Rectangle;

void area(Rectangle);
void perimeter(Rectangle);

int main(){
   Rectangle myRectangle={5,10};
   area(myRectangle);
```

```
    perimeter(myRectangle);

}

void area(Rectangle a){
    printf("Area = %.2f\n",a.height*a.width);
}
void perimeter(Rectangle a){
    printf("Perimeter = %.2f\n",2*(a.height+a.width));
}
```

**Simple Calculator Using Function Pointers**

**Problem Statement:**

**Write a C program to implement a simple calculator. Use function pointers to dynamically call functions for addition, subtraction, multiplication, and division based on user input.**

**Input Example:**

**Enter two numbers: 10 5**

**Choose operation (+, -, *, /): ***

**Output Example:**

**Result: 50**

```
#include<stdio.h>
void add(int,int);
void sub(int,int);
void mul(int,int);
void div(int,int);

int main(){
    char operation;
    int a,b;
    printf("Enter number 1:");
```

```
    scanf("%d",&a);
    printf("Enter number 2: ");
    scanf("%d",&b);

    void (*fun_ptr_arr[])(int,int)={add,sub,mul,div};
    printf("Choose operation (+, -, *, /): ");
    scanf(" %c", &operation);

    switch (operation) {
        case '+':
            (*fun_ptr_arr[0])(a,b);
            break;
        case '-':
           (*fun_ptr_arr[1])(a,b);
            break;
        case '*':
            (*fun_ptr_arr[2])(a,b);
            break;
        case '/':
             (*fun_ptr_arr[3])(a,b);
            break;
        default:
            printf("Invalid operation!\n");
            return 1;
    }
}



void add(int a,int b){
    printf("Result of Sum = %d",a+b);
```

```
}
void sub(int a,int b){
   printf("Result of Subtraction = %d",a-b);
}
void mul(int a,int b){
   printf("Result of Multiplication = %d",a*b);
}
void div(int a,int b){
   printf("Result of Division= %d",a/b);
}
```

**Array Operations Using Function Pointers**

**Problem Statement:**

**Write a C program that applies different operations to an array of integers using function pointers. Implement operations like**

**finding the maximum, minimum, and sum of elements.**

**Input Example:**

**Enter size of array: 4**

**Enter elements: 10 20 30 40**

**Choose operation (1 for Max, 2 for Min, 3 for Sum): 3**

**Output Example:**

**Result: 100**

```
#include<stdio.h>
void arr_max(int[],int);
void arr_min(int[],int);
void arr_sum(int[],int);

int main(){
   int n;
```

```
    printf("Enter size of array: ");
    scanf("%d",&n);

    int array[n];



    for(int i=0;i<n;i++){
        printf("Enter %d th element: ",i+1);
        scanf("%d",&array[i]);
    }
    printf("\nARRAY ELEMENTS\n");
    for(int i=0;i<n;i++){
        printf("%d ",array[i]);

    }printf("\n");

    void (*fun_ptr_arr[])(int[],int)={arr_max,arr_min,arr_sum};
    (*fun_ptr_arr[0])(array,n);
    (*fun_ptr_arr[1])(array,n);
    (*fun_ptr_arr[2])(array,n);

}

void arr_max(int a[],int n){
    int max=a[0];
    for(int i=0;i<n;i++){
        if(a[i]>max){
            max=a[i];
        }
    }printf("Maximum value = %d\n",max);
```

```
}

void arr_min(int a[],int n){
   int min=a[0];
   for(int i=0;i<n;i++){
      if(a[i]<min){
         min=a[i];
      }
   }printf("Minimum value = %d\n",min);
}
void arr_sum(int a[],int n){
   int sum=0;
   for(int i=0;i<n;i++){
      sum=sum+a[i];

   }printf("Array sum = %d\n",sum);
}
```

**Event System Using Function Pointers**

**Problem Statement:**

**Write a C program to simulate a simple event system.**

**Define three events: onStart, onProcess, and onEnd.**

**Use function pointers to call appropriate event handlers dynamically based on user selection.**

**Input Example:**

**Choose event (1 for onStart, 2 for onProcess, 3 for onEnd): 1**

**Output Example:**

**Event: onStart**

**Starting the process...**

```
#include<stdio.h>
void onStart();
void onProcess();
void onEnd();
int main(){
   int ch;
   printf("Choose event (1 for onStart, 2 for onProcess, 3 for onEnd)");
   scanf("%d",&ch);

   void (*fun_ptr_arr[])()={onStart,onProcess,onEnd};

   switch (ch)
   {
   case 1:
      (*fun_ptr_arr[0])();
      break;
   case 2:
      (*fun_ptr_arr[1])();
      break;
   case 3:
      (*fun_ptr_arr[2])();
      break;

   default:
   printf("Invalid option\n");
      break;
   }
}
void onStart(){
   printf("Event:onStart\n");
```

```
    printf("Starting the process...\n");
}
void onProcess(){
    printf("Event:onProcess\n");
    printf("Processing the process...\n");
}
void onEnd(){
    printf("Event:onEnd\n");
    printf("Finished the process...\n");
}
```

**Matrix Operations with Function Pointers**

**Problem Statement:**

**Write a C program to perform matrix operations using function pointers. Implement functions to add, subtract, and multiply matrices. Pass the function pointer to a wrapper function to perform the desired operation.**

**Input Example:**

**Enter matrix size (rows and columns): 2 2**

**Enter first matrix:**

**1 2**

**3 4**

**Enter second matrix:**

**5 6**

**7 8**

**Choose operation (1 for Add, 2 for Subtract, 3 for Multiply): 1**

**Output Example:**

**Result:**

**6 8**

**10 12**

```
#include<stdio.h>
void add_matrix(int m,int n, int a[m][n],int b[m][n]);
void sub_matrix(int m,int n, int a[m][n],int b[m][n]);
void mul_matrix(int m,int n, int a[m][n],int b[m][n]);

int main(){
   int row,col;
   printf("Enter matrix size (rows and columns):");
   scanf("%d%d",&row,&col);
   int matA[row][col];
   int matB[row][col];

   printf("Enter first matrix:\n");
   for(int i=0;i<row;i++){
      for(int j=0;j<col;j++){
         scanf("%d",&matA[i][j]);
      }
   }

   printf("Enter second matrix:\n");
   for(int i=0;i<row;i++){
      for(int j=0;j<col;j++){
         scanf("%d",&matB[i][j]);
      }
   }

   printf("MATRIX A\n");
   for(int i=0;i<row;i++){
      for(int j=0;j<col;j++){
         printf("%d ",matA[i][j]);
      }printf("\n");
```

```
}

printf("MATRIX B\n");
for(int i=0;i<row;i++){
   for(int j=0;j<col;j++){
      printf("%d ",matB[i][j]);
   }printf("\n");
}

void (*fun_ptr_array[])(int,int,int[row][col],int[row][col]) = {add_matrix,sub_matrix,mul_matrix};

int ch;
printf("Choose operation (1 for Add, 2 for Subtract, 3 for Multiply):\n");
scanf("%d",&ch);

switch (ch)
{
case 1:
   (*fun_ptr_array[0])(row,col,matA,matB);
   break;
case 2:
   (*fun_ptr_array[1])(row,col,matA,matB);
   break;
case 3:
   (*fun_ptr_array[2])(row,col,matA,matB);
   break;

default:
printf("Choose right options..\n");
   break;
}
```

```
}

void add_matrix(int m,int n, int a[m][n],int b[m][n]){
   printf("SUM OF MATRIX A AND MATRIX B\n");
   for(int i=0;i<m;i++){
      for(int j=0;j<n;j++){
         printf("%d ",a[i][j]+b[i][j]);
      }printf("\n");
   }
}
void sub_matrix(int m,int n, int a[m][n],int b[m][n]){
   printf("SUBTRACTION OF MATRIX A AND MATRIX B\n");
   for(int i=0;i<m;i++){
      for(int j=0;j<n;j++){
         printf("%d ",a[i][j]-b[i][j]);
      }printf("\n");
   }
}
void mul_matrix(int m,int n, int a[m][n],int b[m][n]){
   printf("MULTIPLICATION OF MATRIX A AND MATRIX B\n");
   for(int i=0;i<m;i++){
      for(int j=0;j<n;j++){
         printf("%d ",a[i][j]*b[i][j]);
      }printf("\n");
   }
}
```

**Problem Statement: Vehicle Management System**

**Write a C program to manage information about various vehicles. The program should demonstrate the following:**

**Structures: Use structures to store common attributes of a vehicle, such as vehicle type, manufacturer name, and model year.**

**Unions: Use a union to represent type-specific attributes, such as:**

**Car: Number of doors and seating capacity.**

**Bike: Engine capacity and type (e.g., sports, cruiser).**

**Truck: Load capacity and number of axles.**

**Typedefs: Define meaningful aliases for complex data types using typedef (e.g., for the structure and union types).**

**Bitfields: Use bitfields to store flags for vehicle features like airbags, ABS, and sunroof.**

**Function Pointers: Use a function pointer to dynamically select a function to display specific information about a vehicle based on its type.**

**Requirements**

**Create a structure Vehicle that includes:**

**A char array for the manufacturer name.**

**An integer for the model year.**

**A union VehicleDetails for type-specific attributes.**

**A bitfield to store vehicle features (e.g., airbags, ABS, sunroof).**

**A function pointer to display type-specific details.**

**Write functions to:**

**Input vehicle data, including type-specific details and features.**

**Display all the details of a vehicle, including the type-specific attributes.**

**Set the function pointer based on the vehicle type.**

**Provide a menu-driven interface to:**

**Add a vehicle.**

**Display vehicle details.**

**Exit the program.**

**Example Input/Output**

**Input:**

**1. Add Vehicle**

**2. Display Vehicle Details**

**3. Exit**

**Enter your choice: 1**

**Enter vehicle type (1: Car, 2: Bike, 3: Truck): 1**

**Enter manufacturer name: Toyota**

**Enter model year: 2021**

**Enter number of doors: 4**

**Enter seating capacity: 5**

**Enter features (Airbags[1/0], ABS[1/0], Sunroof[1/0]): 1 1 0**

**1. Add Vehicle**

**2. Display Vehicle Details**

**3. Exit**

**Enter your choice: 2**

**Output:**

**Manufacturer: Toyota**

**Model Year: 2021**

**Type: Car**

**Number of Doors: 4**

**Seating Capacity: 5**

**Features: Airbags: Yes, ABS: Yes, Sunroof: No**

```
#include <stdio.h>
#include <stdbool.h>
#include <stdlib.h>
typedef struct
{
```

```
    unsigned int airbag : 1;
    unsigned int abs : 1;
    unsigned int sunroof : 1;
} vehicleFeatures;

typedef union
{
    struct
    {
        int doors;
        int seating_capacity;
    } car;

    struct
    {
        int engineCapacity;
        char bikeType[30];
    } bike;

    struct
    {
        int loadCapacity;
        int axels;
    } truck;

} vehicleDetails;

typedef struct
{
    char manufacture_name[40];
    int modelYear;
```

```
    int type;
    vehicleFeatures features;
    vehicleDetails details;
} vehicle;

void add_vehicle();
void display_vehicle();

vehicle VEHICLE[100];
int count = 0;

int main()
{

    while (1)
    {
        int ch;
        printf("Enter choice\n1. Add Vehicle\n2. Display Vehicle Details\n3. Exit");
        scanf("%d", &ch);

        switch (ch)
        {
        case 1:
            add_vehicle();
            break;
        case 2:
            display_vehicle();
            break;
        case 3:
            exit(0);
            break;
```

```
        default:
            printf("invalid option\n");
            break;
        }
    }

    return 0;
}

void add_vehicle()
{
    vehicle v;

    printf("Enter vehicle type (1: Car, 2: Bike, 3: Truck): ");
    scanf("%d", &v.type);

    switch (v.type)
    {
    case 1:
        printf("Enter manufacturer name: ");
        scanf("%s", v.manufacture_name);

        printf("Enter model year: ");
        scanf("%d", &v.modelYear);

        printf("Enter number of doors: ");
        scanf("%d", &v.details.car.doors);

        printf("Enter the seating capacity: ");
        scanf("%d", &v.details.car.seating_capacity);
```

```
    break;

case 2:
    printf("Enter manufacturer name: ");
    scanf("%s", v.manufacture_name);

    printf("Enter model year: ");
    scanf("%d", &v.modelYear);

    printf("Enter engine capavity: ");
    scanf("%d", &v.details.bike.engineCapacity);

    printf("Enter bike type: ");
    scanf("%s", v.details.bike.bikeType);
    break;

case 3:
    printf("Enter manufacturer name: ");
    scanf("%s", v.manufacture_name);

    printf("Enter model year: ");
    scanf("%d", &v.modelYear);

    printf("Enter load capacity: ");
    scanf("%d", &v.details.truck.loadCapacity);

    printf("Enter number of axels: ");
    scanf("%d", &v.details.truck.axels);
    break;

default:
```

```
        printf("invalid option");
        exit(1);
    }

    printf("Enter features (Airbags[1/0], ABS[1/0], Sunroof[1/0]):");
    int airbags, abs, sunroof;
    scanf("%d %d %d", &airbags, &abs, &sunroof);
    v.features.airbag = airbags;
    v.features.abs = abs;
    v.features.sunroof = sunroof;

    VEHICLE[count] = v;
    count = count + 1;

    printf("Details added successfully\n");
}

void display_vehicle()
{

    for (int i = 0; i < count; i++)
    {
        printf("Manufacturer: %s\n", VEHICLE[i].manufacture_name);
        printf("Model Year: %d\n", VEHICLE[i].modelYear);
        if (VEHICLE[i].type == 1)
        {
            printf("Type: car\n");
            printf("Number of Doors: %d\n ", VEHICLE[i].details.car.doors);
            printf("Seating Capacity: %d\n", VEHICLE[i].details.car.seating_capacity);
            printf("Features: Airbags: %s, ABS: %s, Sunroof: %s\n",
                VEHICLE[i].features.airbag ? "YES" : "NO",
```

```
            VEHICLE[i].features.abs ? "YES" : "NO",
            VEHICLE[i].features.sunroof ? "YES" : "NO");
        }
        else if (VEHICLE[i].type == 2)
        {
            printf("Type : Bike\n");
            printf("Engine Capacity: %d\n", VEHICLE[i].details.bike.engineCapacity);
            printf("Bike Yype: %s\n", VEHICLE[i].details.bike.bikeType);
            printf("Features: Airbags: %s, ABS: %s, Sunroof: %s\n",
                VEHICLE[i].features.airbag ? "YES" : "NO",
                VEHICLE[i].features.abs ? "YES" : "NO",
                VEHICLE[i].features.sunroof ? "YES" : "NO");
        }
        else if (VEHICLE[i].type == 3)
        {
            printf("Type : Truck");
            printf("Load apacity :%d\n", VEHICLE[i].details.truck.loadCapacity);
            printf("Number of Axels :%d\n", VEHICLE[i].details.truck.axels);
            printf("Features: Airbags: %s, ABS: %s, Sunroof: %s\n",
                VEHICLE[i].features.airbag ? "YES" : "NO",
                VEHICLE[i].features.abs ? "YES" : "NO",
                VEHICLE[i].features.sunroof ? "YES" : "NO");
        }
        else
        {
            printf("ivalid input");
        }
    }
}
```

**WAP to find factorial using recursion**

```
#include<stdio.h>
int factorial(int);
int main(){
   int n;
   printf("enter the limit");
   scanf("%d",&n);
   int result=factorial(n);
   printf("result = %d",result);
   return 0;
}

int factorial(int n){

   if(n==0){
      return 1;
   }else{
      return n*factorial(n-1);
   }
   }
```

OUTPUT

```
enter the limit: 5
result = 120
```

**WAP to find the sum of digits of a number using recursion.**

```
#include<stdio.h>
int sumOfDigits(int);
```

```
int main(){
   int num;
   printf("Enter number");
   scanf("%d",&num);
   int result=sumOfDigits(num);
   printf("SUM OF DIGITS= %d",result);
   return 0;

}

int sumOfDigits(int n){
    if(n==0){
       return 0;
   }else{
       return (n%10)+sumOfDigits(n/10);
   }
}
```

OUTPUT

Enter number: 12345

SUM OF DIGITS= 15

**With Recursion Findout the maximum number in a given array**

```
#include<stdio.h>
int max_array(int,int[]);

int main(){
   int n;
   printf("enter number of elements");
   scanf("%d",&n);
   int arr[n];
```

```
    for(int i=0;i<n;i++){
        printf("enter %d th element\n",i+1);
        scanf("%d",&arr[i]);
    }
    int res=max_array(n,arr);
    printf("maximum number= %d",res);
}
int max_array(int n,int a[]){
    if(n==1){
        return a[0];
    }

    int max=max_array(n-1,a);
    if (a[n-1]>max){
         return a[n-1];
    }else{
        return max;
    }
}
```

OUTPUT

```
enter number of elements: 5
enter 1 th element: 40
enter 2 th element: 20
enter 3 th element: 80
enter 4 th element: 10
enter 5 th element: 50
maximum number= 80
```

**4. With recurion calculate the power of a given number**

```
#include<stdio.h>
int power(int ,int);
int main(){
   int num,pow;
   printf("enter num");
   scanf("%d",&num);
   printf("enter power");
   scanf("%d",&pow);
   int res=power(num,pow);
   printf("result = %d",res);
   return 0;
}

int power(int n,int p){
   if(p==0){
      return 1;
   }

   return n*power(n,p-1);
}
```

OUTPUT

enter number: 2

enter power: 5

result = 32

**With Recursion calculate the length of a string.**

```
#include<stdio.h>
int lenghtOfSring(char[]);
```

```
int main(){
   char str[50];
   printf("enter the string");
   scanf("%[^\n]",str);

   int res=lenghtOfSring(str);
   printf("LENGTH OF STRING = %d",res);
   return 0;
}

int lenghtOfSring(char str[]){
   if (str[0]=='\0'){
      return 0;
   }
  return 1+lenghtOfSring(str+1);

}
```

OUTPUT

```
enter the string: Amritha Rajeevan M
LENGTH OF STRING = 18
```

**With recursion revrsal of a string**

```
#include<stdio.h>
void reverseOfSring(char[],int);
int main(){
   char str[50];
   printf("enter the string");
   scanf("%[^\n]",str);
   reverseOfSring(str,0);
```

```
    return 0;
}

void reverseOfSring(char str[],int index){
    if (str[index]=='\0'){
        return;
    }
    reverseOfSring(str,index+1);
    printf("%c",str[index]);



}
```

OUTPUT

enter the string: Amritha Rajeevan M

Reverse of string is :M naveejaR ahtirmA