Assignment

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 struct {

  int real;

  int imag;

}Complex;

int main()

{

  Complex c1,c2,sum,product;

  printf("Enter first complex number(real and imaginary):");

  scanf("%d %d",&c1.real,&c1.imag);

  printf("Enter second complex number(real and imaginary):");

  scanf("%d %d",&c2.real,&c2.imag);

  sum.real=c1.real+c2.real;

  sum.imag=c1.imag+c2.imag;

  product.real=c1.real*c2.real-c1.imag*c2.imag;

  product.imag=c1.real*c2.imag+c1.real*c2.imag;

  printf("Sum:%d+%di\n",sum.real,sum.imag);

  printf("Product:%d+%di\n",product.real,product.imag);

  return 0;
}
```

Output:

```
Enter first complex number(real and imaginary):3 4
Enter second complex number(real and imaginary):1 2
Sum:4+6i
Product:-5+12i
```

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 {
   float width;
   float height;
}Rectangle;
int main()
{
   Rectangle rect;
   float area,perimeter;
   printf("Enter width and height of the rectangle:");
   scanf("%f %f",&rect.width,&rect.height);
   area=rect.width*rect.height;
   perimeter=2*(rect.width+rect.height);
```

```
    printf("Area:%.2f",area);
    printf("Perimeter:%.2f",perimeter);
    return 0;
}
```

Output:

```
Enter width and height of the rectangle:5 10
Area:50.00Perimeter:30.00
```

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>
int add(int , int);
int sub(int , int);
int mul(int , int);
int div(int , int);
int main()
{
    int a,b,res;
    char op;
    int (*operation)(int , int);
    printf("Enter two numbers:");
    scanf("%d %d",&a,&b);
    printf("Choose operation(+,-,*,/):");
    scanf(" %c",&op);
    switch (op)
```

```
   {
      case '+':
      operation=add;
      break;
      case '-':
      operation=sub;
      break;
      case '*':
      operation=mul;
      break;
      case '/':
      operation=div;
      break;
      default:
      printf("Invalid operation \n");
   }
   res=operation(a,b);
   printf("Result:%d",res);
   return 0;
}
int add(int a, int b)
{
   return a+b;
}
int sub(int a, int b)
{
   return a-b;
}
int mul(int a, int b)
{
   return a*b;
```

```
}
int div(int a, int b)
{
    return a/b;
}
```

Output:

```
Enter two numbers:10 5
Choose operation(+,-,*,/):*
Result:50
```

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>
int min(int [],int );
int max(int [],int );
int sum(int [],int );
int main()
{
    int n,choice,res;
    printf("Enter size of array:");
    scanf("%d",&n);
    int arr[n];
    printf("Enter elements:");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("Choose operation(1 for Max,2 for Min,3 for Sum):");
    scanf("%d",&choice);
    int(*operation[])(int[],int)={max,min,sum};
    if(choice>=1 && choice<=3)
    {
        printf("Result:%d \n",operation[choice-1](arr,n));
    }
    else
    {
        printf("Invalid choice\n");
    }
    return 0;
}
int max(int arr[],int n)
{
    int max=arr[0];
    for(int i=0;i<n;i++)
    {
        if(arr[i]>max)
        {
            max=arr[i];
        }
    }
    return max;
}
int min(int arr[],int n)
{
    int min=arr[0];
    for(int i=0;i<n;i++)
    {
        if(arr[i]<min)
        {
            min=arr[i];
        }
    }
    return min;
}
int sum(int arr[],int n)
{
    int sum=0;
    for(int i=0;i<n;i++)
    {
        sum+=arr[i];
    }
    return sum;
}
```

Output:

```
Enter size of array:4
Enter elements:10 20 30 40
Choose operation(1 for Max,2 for Min,3 for Sum):3
Result:100
```

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);
void onProcess(void);
void onEnd(void);
int main()
{
   int choice;
   void(*event[])()={onStart,onProcess,onEnd};
   printf("Choose event(1 for onStart,2 for onProcess,3 for onEnd):");
   scanf("%d",&choice);
   if(choice>=1 && choice<=3)
   {
      event[choice-1]();
   }
   else
   {
      printf("Invalid choice \n");
   }
   return 0;
}
void onStart()
{
   printf("Event:onStart \nStarting the process...\n");
}
void onProcess()
{
    printf("Event:onProcess \nProcessing the event...\n");
}
void onEnd()
{
    printf("Event:OnEnd \nEnding the event...\n");
}
```

Output:

```
Choose event(1 for onStart,2 for onProcess,3 for onEnd):1
Event:onStart
Starting the process...
```

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
```

Output:

```
#include<stdio.h>
int add(int m,int n,int a[m][n],int b[m][n],int res[m][n]);
int sub(int m,int n,int a[m][n],int b[m][n],int res[m][n]);
int mul(int m,int n,int a[m][n],int b[m][n],int res[m][n]);

int main()
{
   int m,n,choice;
   printf("Enter matrix size(rows and columns):");
   scanf("%d %d",&m,&n);
   int a[m][n],b[m][n],res[m][n];
   printf("Enter first matrix:\n");
   for(int i=0;i<m;i++)
   {
      for(int j=0;j<n;j++)
      {
         scanf("%d",&a[i][j]);
      }
   }
   printf("Enter second matrix:\n");
   for(int i=0;i<m;i++)
   {
      for(int j=0;j<n;j++)
      {
         scanf("%d",&b[i][j]);
      }
   }
   printf("Choose operation(1 for Add,2 for subtract,3 for Multiply):");
   scanf("%d",&choice);
   int (*operation)(int, int, int[m][n], int[m][n], int[m][n]);
   switch (choice) {
      case 1:
      operation = add;
      break;
      case 2:
      operation = sub;
      break;
      case 3:
      operation = mul;
      break;
      default:
         printf("Invalid choice\n");
         return 1;
   }
   operation(m, n, a, b, res);
   printf("Result:\n");
   for (int i = 0; i < m; i++) {
      for (int j = 0; j < n; j++)
         printf("%d ", res[i][j]);
      printf("\n");
   }

   return 0;

}
int add(int m, int n, int a[m][n], int b[m][n], int res[m][n]) {
   for (int i = 0; i < m; i++)
      for (int j = 0; j < n; j++)
         res[i][j] = a[i][j] + b[i][j];
}

int sub(int m, int n, int a[m][n], int b[m][n], int res[m][n]) {
   for (int i = 0; i < m; i++)
      for (int j = 0; j < n; j++)
         res[i][j] = a[i][j] - b[i][j];
}
int mul(int m, int n, int a[m][n], int b[m][n], int res[m][n]) {
   for (int i = 0; i < m; i++)
      for (int j = 0; j < n; j++) {
         res[i][j] = 0;
         for (int k = 0; k < n; k++)
             res[i][j] += a[i][k] * b[k][j];
      }
}
```

Output:

```
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
Result:
6 8
10 12
```

Problem Statement: Vehicle Management System

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

1. **Structures**: Use structures to store common attributes of a vehicle, such as vehicle type, manufacturer name, and model year.
2. **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.
3. **Typedefs**: Define meaningful aliases for complex data types using typedef (e.g., for the structure and union types).
4. **Bitfields**: Use bitfields to store flags for vehicle features like **airbags**, **ABS**, and **sunroof**.
5. **Function Pointers**: Use a function pointer to dynamically select a function to display specific information about a vehicle based on its type.

Requirements

1. 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.
2. 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.
3. 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
```

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef union{
   struct{
      int doors;
      int seating_capacity;
   }car;
   struct{
      int engine_capacity;
      char type[20];
   }bike;
   struct{
        int load_capacity;
        int axles;
    }truck;
}VehicleDetails;
typedef struct{
    unsigned int airbags:1;
    unsigned int ABS:1;
    unsigned int sunroof:1;
}Features;
typedef struct Vehicle{
    int vehicle_type;
    char manufacturer_name[50];
    int model_year;
    VehicleDetails details;
    Features features;
    void(*displayDetails)(struct Vehicle*);
}Vehicle;
void inputVehicle(Vehicle *v);
void displayCarDetails(Vehicle *v);
void displayBikeDetails(Vehicle *v);
void displayTruckDetails(Vehicle *v);
void setFunctionPointer(Vehicle *v);
void displayVehicle(Vehicle *v);
int main()
{
    Vehicle vehicles[100];
    int vehicle_count=0;
    int choice;
    while(1)
    {
        printf("1.Add a vehicle\n");
        printf("2.Display vehicle details\n");
        printf("3.Exit\n");
        printf("Enter your choice:");
        scanf("%d",&choice);
        switch(choice)
        {
           case 1:
           if(vehicle_count<100)
           {
              inputVehicle(&vehicles[vehicle_count]);
              setFunctionPointer(&vehicles[vehicle_count]);
              vehicle_count++;
           }
           else
           {
              printf("Vehicle storage full \n");
           }
           break;
           case 2:
           if(vehicle_count==0)
           {
              printf("No vehicles are to display\n");
           }
           else
           {
              for(int i=0;i<vehicle_count;i++)
              {
                 displayVehicle(&vehicles[i]);
              }
           }
           break;
```

```
            case 3:
            printf("Exit the program\n");
            exit(0);
            default:
            printf("Invalid choice \n");
        }
    }
    return 0;
}
void inputVehicle(Vehicle *v)
{
    int airbags,ABS,sunroof;
    printf("Enter vehicle type(1:Car, 2:Bike, 3:Truck):");
    scanf("%d",&v->vehicle_type);
    printf("Enter manufacturer name:");
    scanf("%s",v->manufacturer_name);
    printf("Enter model year:");
    scanf("%d",&v->model_year);
    if(v->vehicle_type==1)
    {
        printf("Enter number of doors:");
        scanf("%d",&v->details.car.doors);
        printf("Enter seating capacity:");
        scanf("%d",&v->details.car.seating_capacity);
    }
    else if(v->vehicle_type==2)
    {
      printf("Enter engine capacity(in CC):");
      scanf("%d",&v->details.bike.engine_capacity);
      printf("Enter type(e.g., Sports,Cruiser):");
      scanf("%s",v->details.bike.type);
    }
    else if(v->vehicle_type==3)
    {
        printf("Enter load capacity:");
        scanf("%d",&v->details.truck.load_capacity);
        printf("Enter number of axles:");
        scanf("%d",&v->details.truck.axles);
    }
    else
    {
        printf("Invalid vehicle type \n");
    }
    printf("Enter features (Airbags[1/0], ABS[1/0], Sunroof[1/0]): ");
    scanf("%u %u %u", &airbags, &ABS, &sunroof);
    v->features.airbags=airbags;
    v->features.ABS=ABS;
    v->features.sunroof=sunroof;
}
void setFunctionPointer(Vehicle *v)
{
    if(v->vehicle_type==1)
    {
        v->displayDetails=displayCarDetails;
    }
    else if(v->vehicle_type==2)
    {
        v->displayDetails=displayBikeDetails;
    }
    else if(v->vehicle_type==3)
    {
        v->displayDetails=displayTruckDetails;
```

```
   }
}
void displayVehicle(Vehicle *v)
{
   printf("Manufacturer:%s \n",v->manufacturer_name);
   printf("Model Year:%d\n",v->model_year);
   if(v->displayDetails)
   {
      v->displayDetails(v);
   }
   printf("Features:Airbag:%s,ABS:%s,Sunroof:%s\n",
   v->features.airbags?"Yes":"No",
   v->features.ABS?"Yes":"No",
   v->features.sunroof?"Yes":"No");
}
void displayCarDetails(Vehicle *v)
{
   printf("Type:Car\n");
   printf("Number of Doors:%d \n",v->details.car.doors);
   printf("Seating Capacity:%d\n",v->details.car.seating_capacity);
}
void displayBikeDetails(Vehicle *v)
{
   printf("Type:Bike\n");
   printf("Engine Capacity:%d CC\n",v->details.bike.engine_capacity);
   printf("Model:%s\n",v->details.bike.type);
}
void displayTruckDetails(Vehicle *v)
{
   printf("Type:Truck \n");
   printf("Load Capacity:%d tons\n",v->details.truck.load_capacity);
```

```
    printf("Number of Axles:%d\n",v->details.truck.axles);
}
```

Output:

```
1.Add a 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 1
1.Add a vehicle
2.Display vehicle details
3.Exit
Enter your choice:1
Enter vehicle type(1:Car, 2:Bike, 3:Truck):2
Enter manufacturer name:Honda
Enter model year:2024
Enter engine capacity(in CC):150
Enter type(e.g., Sports,Cruiser):Cruiser
Enter features (Airbags[1/0], ABS[1/0], Sunroof[1/0]): 0 0 0
1.Add a vehicle
2.Display vehicle details
3.Exit
Enter your choice:1
Enter vehicle type(1:Car, 2:Bike, 3:Truck):3
Enter manufacturer name:Eicher
Enter model year:2010
Enter load capacity:12
Enter number of axles:4

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

1.Add a vehicle

2.Display vehicle details

3.Exit

Enter your choice:2

Manufacturer:Toyota

Model Year:2021

Type:Car

Number of Doors:4

Seating Capacity:5

Features:Airbag:Yes,ABS:Yes,Sunroof:Yes

Manufacturer:Honda

Model Year:2024

Type:Bike

Engine Capacity:150 CC

Model:Cruiser

Features:Airbag:No,ABS:No,Sunroof:No

Manufacturer:Eicher

Model Year:2010

Type:Truck

Load Capacity:12 tons

Number of Axles:4

Features:Airbag:No,ABS:No,Sunroof:No

1.Add a vehicle

2.Display vehicle details

3.Exit

Enter your choice:3

Exit the program
```

1.WAP to find out the factorial of a number using recursion.

```
#include<stdio.h>
int factorial(int);
int main()
{
   int n;printf("Enter the number to calculate factorial:");
   scanf("%d",&n);
   if(n<0)
   {
      printf("Factorial cannot be negative numbers \n");
   }
   else
   {
      printf("Factorial is:%d",factorial(n));
   }
}
int factorial(int n)
{
   if(n==0 || n==1)
   {
      return 1;
   }
   return n*factorial(n-1);
}
```

Output:

```
Enter the number to calculate factorial:5
Factorial is:120
```

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

```
#include<stdio.h>
int sumOfDigits(int n);
int main()
{
  int num;
  printf("Enter the number:");
  scanf("%d",&num);
  printf("Sum of digits:%d\n",sumOfDigits(num));
  return 0;
}
int sumOfDigits(int n)
{
    if(n==0)
    {
        return 0;
    }
    return (n%10)+sumOfDigits(n/10);
}
```

Output:

```
Enter the number:1234
Sum of digits:10
```

3.With Recursion Findout the maximum number in a given array

```
#include<stdio.h>
int maximum(int arr[],int size);
int main()
{
    int n;
    printf("Enter size of array:");
    scanf("%d",&n);
    int arr[n];
    printf("Enter elements in array:");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("Maximum number in the array:%d\n",maximum(arr,n));
    return 0;
}
int maximum(int arr[],int size)
{
    if(size==1)
    {
        return arr[0];
    }
    int max=maximum(arr,size-1);
    if(arr[size-1]>max)
    {
        return arr[size-1];
    }
    else
    {
        return max;
    }
}
```

Output:

```
Enter size of array:5
Enter elements in array:6 8 9 3 2
Maximum number in the array:9
```

4. With recursion calculate the power of a given number

```
#include<stdio.h>
int power(int m,int n);
int main()
{
   int m,n;
   printf("Enter the number:");
   scanf("%d",&m);
   printf("Enter the exponent:");
   scanf("%d",&n);
   printf("Result is:%d\n",power(m,n));
   return 0;
}
int power(int m,int n)
{
   if(n==0)
   {
      return 1;
   }
   return m*power(m,n-1);
}
```

Output:

```
Enter the number:2
Enter the exponent:3
Result is:8
```

5. With Recursion calculate the length of a string.

```
#include<stdio.h>
int strLen(char str[]);
int main()
{
   char str[50];
   printf("Enter a string:");
   scanf("%[^\n]",str);
```

```
    printf("Length of string:%d",strLen(str));
    return 0;
}
int strLen(char str[])
{
    if(str[0]=='\0')
    {
        return 0;
    }
    return 1+strLen(str+1);
}
```

Output:

```
Enter a string:Hello, Good morning
Length of string:19
```

6. With recursion reversal of a string

```
#include<stdio.h>
void reverseString(char str[],int len);
int main()
{
    char str[50];
    printf("Enter the string:");
    scanf("%[^\n]",str);
    int len=0;
    while(str[len]!='\0')
    {
        len++;
    }
    reverseString(str,len);
    printf("Reversed string:%s\n",str);
    return 0;
}
void reverseString(char str[],int len)
{
    static int index=0;
    if(index>=len/2)
    {
        return;
    }
    char temp=str[index];
    str[index]=str[len-index-1];
    str[len-index-1]=temp;
    index++;
    reverseString(str,len);
}
```

Output:

```
Enter the string:Hello World
Reversed string:dlroW olleH
```