1)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

ANSWER

```
#include<stdio.h>
typedef struct {
    int real;
    int imaginary;
}Complex;

Complex addComplex(Complex c1, Complex c2);
Complex multiplyComplex(Complex c1, Complex c2);
void displayComplex(Complex c);
int main() {
    Complex c1, c2, sum, product;

       printf("Enter first complex number (real and imaginary): ");
    scanf("%d %d", &c1.real, &c1.imaginary);

      printf("Enter second complex number (real and imaginary): ");
    scanf("%d %d", &c2.real, &c2.imaginary);

       sum = addComplex(c1, c2);
    product = multiplyComplex(c1, c2);

        printf("Sum: ");
    displayComplex(sum);
    printf("Product: ");
    displayComplex(product);

    return 0;
}

Complex addComplex(Complex c1, Complex c2) {
```

```
    Complex result;
    result.real = c1.real + c2.real;
    result.imaginary = c1.imaginary + c2.imaginary;
    return result;
}

Complex multiplyComplex(Complex c1, Complex c2) {
    Complex result;
    result.real = c1.real * c2.real - c1.imaginary * c2.imaginary;
    result.imaginary = c1.real * c2.imaginary + c1.imaginary * c2.real;
    return result;
}

void displayComplex(Complex c) {
    if (c.imaginary >= 0)
        printf("%d + %di\n", c.real, c.imaginary);
    else
        printf("%d - %di\n", c.real, -c.imaginary);
}
```

2)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

ANSWER:

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

float computeArea(Rectangle rect);
float computePerimeter(Rectangle rect);

int main() {
```

```
    Rectangle rect;


    printf("Enter width and height of the rectangle: ");
    scanf("%f %f", &rect.width, &rect.height);


    float area = computeArea(rect);
    float perimeter = computePerimeter(rect);


    printf("Area: %.2f\n", area);
    printf("Perimeter: %.2f\n", perimeter);

    return 0;
}
float computeArea(Rectangle rect) {
    return rect.width * rect.height;
}


float computePerimeter(Rectangle rect) {
    return 2 * (rect.width + rect.height);
}
```

3)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

ANSWER:

```
#include <stdio.h>
float add(float a, float b);
float subtract(float a, float b);
float multiply(float a, float b);
float divide(float a, float b);

int main() {
```

```
    float num1, num2;
    char op;
    float (*operation)(float, float);


    printf("Enter two numbers: ");
    scanf("%f %f", &num1, &num2);
    printf("Choose operation (+, -, *, /): ");
    scanf(" %c", &op); // Space before %c to consume any leftover whitespace


    switch (op) {
        case '+':
            operation = add;
            break;
        case '-':
            operation = subtract;
            break;
        case '*':
            operation = multiply;
            break;
        case '/':
            operation = divide;
            break;
        default:
            printf("Invalid operation\n");
            return 1;
    }


    printf("Result: %.2f\n", operation(num1, num2));
    return 0;
}

float add(float a, float b) {
    return a + b;
}

float subtract(float a, float b) {
    return a - b;
}

float multiply(float a, float b) {
    return a * b;
```

```
}

float divide(float a, float b) {
    if (b != 0)
        return a / b;
    else {
        printf("Error: Division by zero\n");
        return 0;
    }
}
```

4)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

ANSWER:

```
#include <stdio.h>
int findMax(int arr[], int size);
int findMin(int arr[], int size);
int findSum(int arr[], int size);

int main() {
    int size, choice, result;

    printf("Enter size of array: ");
    scanf("%d", &size);
    int arr[size];
    printf("Enter elements: ");
    for (int i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }


    int (*operation)(int[], int);
```

```
    printf("Choose operation (1 for Max, 2 for Min, 3 for Sum): ");
    scanf("%d", &choice);


    switch (choice) {
        case 1:
            operation = findMax;
            break;
        case 2:
            operation = findMin;
            break;
        case 3:
            operation = findSum;
            break;
        default:
            printf("Invalid choice\n");
            return 1;
    }


    result = operation(arr, size);
    printf("Result: %d\n", result);

    return 0;
}


int findMax(int arr[], int size) {
    int max = arr[0];
    for (int i = 1; i < size; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
    }
    return max;
}

int findMin(int arr[], int size) {
    int min = arr[0];
    for (int i = 1; i < size; i++) {
        if (arr[i] < min) {
            min = arr[i];
        }
    }
```

```
    return min;
}

int findSum(int arr[], int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += arr[i];
    }
    return sum;
}
```

5)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...

ANSWER:

```
#include <stdio.h>
void onStart();
void onProcess();
void onEnd();

int main() {
    int choice;

    void (*events[3])() = {onStart, onProcess, onEnd};
    printf("Choose event (1 for onStart, 2 for onProcess, 3 for onEnd): ");
    scanf("%d", &choice);
    if (choice >= 1 && choice <= 3) {
        events[choice - 1
    } else {
        printf("Invalid event\n");
    }

    return 0;
}
```

```
void onStart() {
   printf("Event: onStart\nStarting the process...\n");
}

void onProcess() {
   printf("Event: onProcess\nProcessing the data...\n");
}

void onEnd() {
   printf("Event: onEnd\nEnding the process...\n");
}
```

6)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

ANSWER:

```
#include <stdio.h>
#define MAX_SIZE 10
void addMatrices(int mat1[][MAX_SIZE], int mat2[][MAX_SIZE], int
result[][MAX_SIZE], int rows, int cols);
void subtractMatrices(int mat1[][MAX_SIZE], int mat2[][MAX_SIZE], int
result[][MAX_SIZE], int rows, int cols);
void multiplyMatrices(int mat1[][MAX_SIZE], int mat2[][MAX_SIZE], int
result[][MAX_SIZE], int rows, int cols);
int main() {
   int rows, cols, choice;
   int mat1[MAX_SIZE][MAX_SIZE], mat2[MAX_SIZE][MAX_SIZE],
```

```
result[MAX_SIZE][MAX_SIZE];


    printf("Enter matrix size (rows and columns): ");
    scanf("%d %d", &rows, &cols);

    printf("Enter first matrix:\n");
    for (int i = 0; i < rows; i++)
        for (int j = 0; j < cols; j++)
            scanf("%d", &mat1[i][j]);

    printf("Enter second matrix:\n");
    for (int i = 0; i < rows; i++)
        for (int j = 0; j < cols; j++)
            scanf("%d", &mat2[i][j]);


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


    void (*operation)(int[][MAX_SIZE], int[][MAX_SIZE], int[][MAX_SIZE], int,
int);


    switch (choice) {
        case 1:
            operation = addMatrices;
            break;
        case 2:
            operation = subtractMatrices;
            break;
        case 3:
            operation = multiplyMatrices;
            break;
        default:
            printf("Invalid operation\n");
            return 1;
    }
    operation(mat1, mat2, result, rows, cols);


    printf("Result:\n");
    for (int i = 0; i < rows; i++) {
```

```
        for (int j = 0; j < cols; j++) {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }

    return 0;
}


void addMatrices(int mat1[][MAX_SIZE], int mat2[][MAX_SIZE], int
result[][MAX_SIZE], int rows, int cols) {
    for (int i = 0; i < rows; i++)
        for (int j = 0; j < cols; j++)
            result[i][j] = mat1[i][j] + mat2[i][j];
}

void subtractMatrices(int mat1[][MAX_SIZE], int mat2[][MAX_SIZE], int
result[][MAX_SIZE], int rows, int cols) {
    for (int i = 0; i < rows; i++)
        for (int j = 0; j < cols; j++)
            result[i][j] = mat1[i][j] - mat2[i][j];
}

void multiplyMatrices(int mat1[][MAX_SIZE], int mat2[][MAX_SIZE], int
result[][MAX_SIZE], int rows, int cols) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            result[i][j] = 0;
            for (int k = 0; k < cols; k++) {
                result[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }
}
```

7)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

ANSWER:

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

struct Features {
    unsigned int airbags : 1;
    unsigned int abs : 1;
    unsigned int sunroof : 1;
};

union VehicleDetails {
    struct {
        int doors;
        int seating_capacity;
    } car;
    struct {
        int engine_cap;
        char type[100];
    } bike;
    struct {
        int load_capacity;
        int axles;
    } truck;
};

typedef struct Vehicle {
    char manufacturer[100];
    int model_year;
    union VehicleDetails details;
    struct Features features;
    void (*displayDetails)(struct Vehicle);
} Vehicle;

void displayCarDetails(struct Vehicle v);
void displayBikeDetails(struct Vehicle v);
void displayTruckDetails(struct Vehicle v);
void inputVehicleData(Vehicle *v);
```

```
int main() {
    Vehicle vehicle;
    int choice;

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

        if (choice == 1) {
            inputVehicleData(&vehicle);
        } else if (choice == 2) {
            if (vehicle.displayDetails) {
                vehicle.displayDetails(vehicle);
            } else {
                printf("No vehicle data available. Please add a vehicle first.\n");
            }
        } else if (choice == 3) {
            break;
        } else {
            printf("Invalid choice, please try again.\n");
        }
    }

    return 0;
}

void displayCarDetails(struct Vehicle v) {
    printf("Manufacturer: %s\n", v.manufacturer);
    printf("Model Year: %d\n", v.model_year);
    printf("Vehicle Type: Car\n");
    printf("Number of Doors: %d\n", v.details.car.doors);
    printf("Seating Capacity: %d\n", v.details.car.seating_capacity);
    printf("Features: Airbags: %s, ABS: %s, Sunroof: %s\n",
           v.features.airbags ? "Yes" : "No",
           v.features.abs ? "Yes" : "No",
           v.features.sunroof ? "Yes" : "No");
}

void displayBikeDetails(struct Vehicle v) {
    printf("Manufacturer: %s\n", v.manufacturer);
    printf("Model Year: %d\n", v.model_year);
    printf("Vehicle Type: Bike\n");
```

```
    printf("Engine Capacity: %d\n", v.details.bike.engine_cap);
    printf("Type: %s\n", v.details.bike.type);
    printf("Features: Airbags: %s, ABS: %s\n",
           v.features.airbags ? "Yes" : "No",
           v.features.abs ? "Yes" : "No");
}

void displayTruckDetails(struct Vehicle v) {
    printf("Manufacturer: %s\n", v.manufacturer);
    printf("Model Year: %d\n", v.model_year);
    printf("Vehicle Type: Truck\n");
    printf("Load Capacity: %d\n", v.details.truck.load_capacity);
    printf("Number of Axles: %d\n", v.details.truck.axles);
    printf("Features: Airbags: %s, ABS: %s, Sunroof: %s\n",
           v.features.airbags ? "Yes" : "No",
           v.features.abs ? "Yes" : "No",
           v.features.sunroof ? "Yes" : "No");
}
void inputVehicleData(Vehicle *v) {
    int type;
    unsigned int airbags, abs, sunroof;

    printf("Enter Vehicle Type (1.Car 2.Bike 3.Truck): ");
    scanf("%d", &type);
    getchar();
    printf("Enter Manufacturer Name: ");
    fgets(v->manufacturer, sizeof(v->manufacturer), stdin);
    v->manufacturer[strcspn(v->manufacturer, "\n")] = 0;
    printf("Enter Model Year: ");
    scanf("%d", &v->model_year);
    getchar();

    if (type == 1) {
        printf("Enter Number of Doors: ");
        scanf("%d", &v->details.car.doors);
        getchar();
        printf("Enter Seating Capacity: ");
        scanf("%d", &v->details.car.seating_capacity);
        getchar();
        v->displayDetails = displayCarDetails;
    } else if (type == 2) {
        printf("Enter Engine Capacity: ");
        scanf("%d", &v->details.bike.engine_cap);
        getchar();
```

```
        printf("Enter Bike Type (e.g., Sports, Cruiser): ");
        fgets(v->details.bike.type, sizeof(v->details.bike.type), stdin);
        v->details.bike.type[strcspn(v->details.bike.type, "\n")] = 0;
        v->displayDetails = displayBikeDetails;
    } else if (type == 3) {
        printf("Enter Load Capacity: ");
        scanf("%d", &v->details.truck.load_capacity);
        getchar();
        printf("Enter Number of Axles: ");
        scanf("%d", &v->details.truck.axles);
        getchar();
        v->displayDetails = displayTruckDetails;
    } else {
        printf("Invalid vehicle type! Please enter a valid type (1, 2, or 3).\n");
        return;
    }

    printf("Enter Features (Airbags [1/0], ABS [1/0], Sunroof [1/0]): ");
    scanf("%u %u %u", &airbags, &abs, &sunroof);
    getchar();

    v->features.airbags = airbags;
    v->features.abs = abs;
    v->features.sunroof = sunroof;
}
```

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

ANSWER:

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

```
    }
    else{
        printf("Factorial of %d is %lld",num,factorial(num));
    }
    return 0;
}
```

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

ANSWER:

```
#include<stdio.h>
int SumDigits(int num){
    if(num==0)
        return 0;
    return(num%10)+SumDigits(num/10);
}
int main(){
    int number;
    printf("Enter the number:");
    scanf("%d",&number);
    if(number<0){
        printf("Enter a non-negative number.");
        return 1;
    }
    int result=SumDigits(number);
    printf("The sum of digits of number %d is %d",number,result);
    return 0;
}
```

10)With recursion calculate the power of a given number

ANSWER:

```
#include <stdio.h>

int power(int base, int exponent) {
    if (exponent == 0)
        return 1; // Base case: any number to the power of 0 is 1
    return base * power(base, exponent - 1);
}

int main() {
    int base, exponent;

    printf("Enter the base: ");
```

```
    scanf("%d", &base);
    printf("Enter the exponent: ");
    scanf("%d", &exponent);

    printf("%d to the power of %d is: %d\n", base, exponent, power(base, exponent));
    return 0;
}
```

11) With Recursion calculate the length of a string.

ANSWER:

```
#include <stdio.h>
int stringLength(char *str) {
    if (*str == '\0')
        return 0;
    return 1 + stringLength(str + 1);
}

int main() {
    char str[100];

    printf("Enter a string: ");
    scanf("%s", str);

    printf("The length of the string is: %d\n", stringLength(str));
    return 0;
}
```

12)With recursion revrsal of a string

ANSWER:

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

void reverseString(char *str, int start, int end) {
    if (start >= end)
        return;
    char temp = str[start];
    str[start] = str[end];
    str[end] = temp;


    reverseString(str, start + 1, end - 1);
}
```

```
int main() {
    char str[100];

    printf("Enter a string: ");
    scanf("%s", str);

    int len = strlen(str);
    reverseString(str, 0, len - 1);

    printf("Reversed string is: %s\n", str);
    return 0;
}
```