1.

```c
//1.Array Concept Declaration and printing 1D
#include <stdio.h>
int main(){
    int n=10;
    int ar[n];
    printf("Enter the value :");
    for(int i=0;i<10;i++){
        scanf("%d",&ar[i]);


    }
    printf("\n");
    for(int j=0;j<10;j++){
        printf("%d->",ar[j]);
    }
    return 0;
}
```

OUTPUT :

Enter the value :10

40

70

80

90

100

10->20->30->40->50->60->70->80->90->100->

2.

```c
//2.Array Concept Declaration and printing 2D
#include <stdio.h>
int main(){
    int n=5;
    int k=2;
    int ar[n][k];
    printf("Enter the value :");
    for(int i=0;i<5;i++){
        for(int j=0;j<2;j++)
        {
            scanf("%d",&ar[i][j]);
        }
```

```
    }
    printf("\n");
    for(int i=0;i<5;i++){
        for(int j=0;j<2;j++){
            printf(" %d",ar[i][j]);
    }
    printf("\n");
    }

    return 0;
}
```

OUTPUT :

Enter the value :1

2

3

4

5

6

7

8

9

10

1 2

3 4

5 6

7 8

9 10

3.

```
//Functions 1
```

```c
#include <stdio.h>
void add_num(){
    int a=10,b=20;
    int sum = 0;
    sum = a+b;
    printf("Sum = %d",sum);
}
int main(){

    add_num();
    return 0;

}
```

4.

```c
//Functions

#include <stdio.h>
void add_num(int,int);//Tells the compiler that in the upcoming lines
execution u will find a fn that don't return
int main(){
    int a=10,b=20;
    add_num(a,b);
    return 0;

}
void add_num(int a,int b){
    int sum = 0;
    sum = a+b;
    printf("Sum = %d",sum);
}
```

5.

```c
//Function example  3
/*/*Function Implelemtation
==============================
1. Function Prototype (Function Declaration)
        which is defined before the main() function
        Syntax: return_type function_name(If you are passing paraemter,
mention the datatypes of the parameter);
2. Function call
        This is implmented inside the main() function
        Syntax: function_name(pass the varible only the variable names);
3. Function Definition
        This is implemented after the main() Function
```

```
        Syntax:
        return_type function_name(If you are passing paraemter, mention the
datatypes of the parameter){
            ========
            Function Body
            ========
        }
*/

//WAP to add two number using the add function by parameter and the function
is not going to return
//any Data


*/
#include <stdio.h>
void add_num(int,int);//Tells the compiler that in the upcoming lines
execution u will find a fn that don't return
int main(){
    int a=10,b=20;
    printf("\n001a = %p",&a);
    printf("\n001b = %p",&b);
    add_num(a,b);//Here we r passing the copy of value a and b
    printf("\nThe value of a and b is %d and %d",a,b);
    return 0;

}
void add_num(int a,int b){
    a=40;
    b=50;
    printf("\n002a = %p",&a);
    printf("\n002b = %p",&b);
    int sum = 0;
    sum = a+b;
    printf("\nSum = %d",sum);
}
```

6.

```
//Function example  3
/*/*Function Implelemtation
==============================
1. Function Prototype (Function Declaration)
        which is defined before the main() function
        Syntax: return_type function_name(If you are passing paraemter,
mention the datatypes of the parameter);
2. Function call
        This is implmented inside the main() function
```

```
        Syntax: function_name(pass the varible only the variable names);
3. Function Definition
        This is implemented after the main() Function
        Syntax:
        return_type function_name(If you are passing paraemter, mention the
datatypes of the parameter){
            ========
            Function Body
            ========
        }
*/

//WAP to add two number using the add function by parameter and the function
is not going to return
//any Data
#include <stdio.h>
int add_num(int,int);//Tells the compiler that in the upcoming lines execution
u will find a fn that don't return
int main(){
    int a=10,b=20;
    int sum = 0;
    sum = add_num(a,b);//Here we r passing the copy of value a and b
    printf("\nSum = %d",sum);
    return 0;

}
int add_num(int a,int b){

    int sum = a+b;
    return sum;

}
```

7.

```
/*Function Implelemtation
==============================
1. Function Prototype (Function Declaration)
        which is defined before the main() function
        Syntax: return_type function_name(If you are passing paraemter,
mention the datatypes of the parameter);
2. Function call
        This is implmented inside the main() function
        Syntax: function_name(pass the varible only the variable names);
3. Function Definition
        This is implemented after the main() Function
        Syntax:
```

```
        return_type function_name(If you are passing paraemter, mention the
datatypes of the parameter){
            ========
            Function Body
            ========
        }


Assignment :
1. Create a C program that defines a function to increment an integer by 1.
The function should demonstrate call by value, showing that the original value
remains unchanged.
*/
#include <stdio.h>
void increment_num(int);//Tells the compiler that in the upcoming lines
execution u will find a fn that don't return
int main(){
    int a=10;
    printf("\n001a = %p",&a);

    increment_num(a);//Here we r passing the copy of value a and b
    printf("\nThe value of a is %d ",a);
    return 0;

}
void increment_num(int a){



    printf("\nAfter incrementing the number by 1 the result = %d",++a);
}
```

OUTPUT :

001a = 0061FF1C

After incrementing the number by 1 the result = 11

The value of a is 10

8.

```
/*Function Implelemtation
==============================
1. Function Prototype (Function Declaration)
        which is defined before the main() function
        Syntax: return_type function_name(If you are passing paraemter,
mention the datatypes of the parameter);
2. Function call
        This is implmented inside the main() function
```

```
        Syntax: function_name(pass the varible only the variable names);
3. Function Definition
        This is implemented after the main() Function
        Syntax:
        return_type function_name(If you are passing paraemter, mention the
datatypes of the parameter){
            ========
            Function Body
            ========
        }


Assignment :
2. Write a C program that attempts to swap two integers using a function that
employs call by value. Show that the original values remain unchanged after
the function call.
*/
#include <stdio.h>
void swap_num(int,int);//Tells the compiler that in the upcoming lines
execution u will find a fn that don't return
int main(){
    int a=10,b=20;
    printf("\n001a = %p",&a);
    printf("\n001b = %p",&b);
    swap_num(a,b);//Here we r passing the copy of value a and b
    printf("\nThe value of a and b is:\n a=%d and b=%d",a,b);
    return 0;

}
void swap_num(int a,int b){
    a=a+b;
    b=a-b;
    a=a-b;
    printf("\n002a = %p",&a);
    printf("\n002b = %p",&b);


    printf("\nThe value of a and b after swapping is :\n");
    printf("a=%d and b=%d",a,b);
}
```

OUTPUT : 001b = 0061FF18

002a = 0061FF00

002b = 0061FF04

The value of a and b is:

a=10 and b=20

9.

```
/*Function Implelemtation
==============================
1. Function Prototype (Function Declaration)
        which is defined before the main() function
        Syntax: return_type function_name(If you are passing paraemter,
mention the datatypes of the parameter);
2. Function call
        This is implmented inside the main() function
        Syntax: function_name(pass the varible only the variable names);
3. Function Definition
        This is implemented after the main() Function
        Syntax:
        return_type function_name(If you are passing paraemter, mention the
datatypes of the parameter){
            ========
            Function Body
            ========
        }


Assignment :
3. Develop a C program that calculates the factorial of a number using call by
value.
*/
#include <stdio.h>
void fact_num(int);//Tells the compiler that in the upcoming lines execution u
will find a fn that don't return
int main(){
    int a=10;

    fact_num(a);//Here we r passing the copy of value a and b

    return 0;

}
void fact_num(int a){
    int fact=1;
    for(int i=1;i<=a;i++)
    {
        fact=fact*i;
    }



    printf("\nThe factorial of %d :%d",a,fact);

}
```

OUTPUT : The factorial of 10 :3628800

10.

```
/*Function Implelemtation
==============================
1. Function Prototype (Function Declaration)
        which is defined before the main() function
        Syntax: return_type function_name(If you are passing paraemter,
mention the datatypes of the parameter);
2. Function call
        This is implmented inside the main() function
        Syntax: function_name(pass the varible only the variable names);
3. Function Definition
        This is implemented after the main() Function
        Syntax:
        return_type function_name(If you are passing paraemter, mention the
datatypes of the parameter){
            ========
            Function Body
            ========
        }


Assignment :
4. Create a C program that defines a function to find the maximum of two
numbers using call by value.
*/
#include <stdio.h>

void max_num(int,int);//Tells the compiler that in the upcoming lines
execution u will find a fn that don't return
int main(){
    int a=10,b=20;

    max_num(a,b);//Here we r passing the copy of value a and b

    return 0;

}
void max_num(int a,int b){


    int m=(a > b) ? a : b;
    printf("\nThe maximum of %d and %d is :%d",a,b,m);

}
```

OUTPUT : The maximum of 10 and 20 is :20

11.

Problem Statement 1: Arithmetic Operations Calculator

Description: Write a C program that performs basic arithmetic operations (addition, subtraction, multiplication, and division) on two numbers provided by the user. The program should use functions to perform each operation and demonstrate call by value.

Requirements:
Create separate functions for addition, subtraction, multiplication, and division.
Each function should take two parameters (the numbers) and return the result.
Use appropriate data types for the variables.
Use operators for arithmetic calculations.

Example Input/Output:

Enter first number: 10
Enter second number: 5
Addition: 15
Subtraction: 5
Multiplication: 50
Division: 2.0

```c
#include <stdio.h>

void add_num(int,int);//Tells the compiler that in the upcoming lines
execution u will find a fn that don't return
void sub_num(int,int);
void mul_num(int,int);
void div_num(int,int);
int main(){
    int a,b;
    printf("\nEnter first number : ");
    scanf("%d",&a);
    printf("\nEnter second number : ");
    scanf("%d",&b);

    add_num(a,b);//Here we r passing the copy of value a and b
    sub_num(a,b);
    mul_num(a,b);
    div_num(a,b);
```

```
    return 0;

}
void add_num(int a,int b){


    int sum = 0;
    sum = a+b;
    printf("\nAddition : = %d",sum);

}
void sub_num(int a,int b){


    int diff = 0;
    diff = a-b;
    printf("\nSubtraction: %d",diff);

}
void mul_num(int a,int b){


    int product = 0;
    product = a*b;
    printf("\nMultiplication: %d",product);

}
void div_num(int a,int b){


    float div = 0;
    div = a/b;
    printf("\nDivision: %.1f",div);

}
```

12.

Problem Statement 2: Temperature Conversion

Description: Develop a C program that converts temperatures between Celsius and Fahrenheit. The program should use functions to handle the conversions and demonstrate call by value.

Requirements:
Create two functions: one for converting Celsius to Fahrenheit and another for converting Fahrenheit to Celsius.
Each function should accept a temperature value as an argument and return the converted temperature.
Use appropriate data types for temperature values.
Use arithmetic operators to perform the conversion calculations.

Example Input/Output:

Enter temperature in Celsius: 25
Temperature in Fahrenheit: 77.0

Enter temperature in Fahrenheit: 77
Temperature in Celsius: 25.0

```c
#include <stdio.h>

void to_Fahren(int);
void to_Celc(int);
int main(){
    int temp1,temp2;
    printf("\nEnter temperature in Celsius:");
    scanf("%d",&temp1);

    to_Fahren(temp1);//Here we r passing the copy of value a and b
    printf("\nEnter temperature in Fahrenheit:");
    scanf("%d",&temp2);
    to_Celc(temp2);

    return 0;

}
void to_Fahren(int temp1){

    float F = temp1 * 9/5 + 32;

    printf("\nTemperature in Fahrenheit: %.1f",F);
}
void to_Celc(int temp2){

    float C =  (temp2 - 32) * 5/9;
```

```
    printf("\nTemperature in Celsius: %.1f",C);
}
```

13.

Problem Statement 3: Simple Interest Calculator

Description: Develop a C program that calculates simple interest based on user input for principal amount, rate of interest, and time period. The program should use a function to compute interest and demonstrate call by value.

Requirements:
Implement a function that takes three parameters (principal, rate, time) and returns the calculated simple interest.
Use appropriate data types for financial calculations (e.g., float or double).
Utilize arithmetic operators to compute simple interest using the formula
SI = P×R×T/100

Example Input/Output:

Enter principal amount: 1000
Enter rate of interest: 5
Enter time period (in years): 3
Simple Interest is: 150.0

```
#include <stdio.h>
void simple_interest(int,int,int);
int main(){
    //SI = P×R×T/100
    int P,R,T;
    printf("\nEnter principal amount: ");
    scanf("%d",&P);
    printf("\nEnter rate of interest: ");
    scanf("%d",&R);
    printf("\nEnter time period (in years): ");
    scanf("%d",&T);

    simple_interest(P,R,T);
```

```
    return 0;

}
void simple_interest(int P,int R,int T){

    float SI = P*R*T/100;

    printf("\nSimple Interest is: %.1f",SI);

}
```

14.

```
//Concept pointer

#include <stdio.h>
int main(){
    int a;
    int *p;
    p=&a;
    *p=20;
    printf("\na=%d\n",a);
    printf("Address of A=%p\n",&a);
    printf("Address of *p = %p\n",&p);
    printf("*p=%p\n",p);
    return 0;
}
```

OUTPUT :

a=20

Address of A=0061FF1C

Address of *p = 0061FF18

*p=0061FF1C

15.

```
/*
int *p;
char *p;
float *p;
double *p;
*/
#include <stdio.h>
int main()
```

```
{
    int a=10;
    printf("001 a=%d\n",a);
    int *pa = &a;
    a=*pa +5; //dereferencing variable
    printf("002 a=%d",a);
    return 0;
}
```

16.

```
/*
int *p;
char *p;
float *p;
double *p;
*/
#include <stdio.h>
int main()
{
    int count=10,x;
    int *pCount = &count;
    x = *pCount;
    printf("\nCount = %d, x = %d",count,x);

    return 0;
}
```

17.

```
//Exercise
/*
int *p;
char *p;
float *p;
double *p;
*/
#include <stdio.h>
int main()
{
    char A = 100;
    printf("Address of A=%p\n",&A);
    char *pA = &A;
    char pPA = *pA;
    printf("\nData obtained from read operation on the pointer =  %d",pPA);
    *pA = 65;
    printf("\nA = %d",A);
```

```
    return 0;
}
```

18.

```
/*
int *p;
char *p;
float *p;
double *p;
*/
#include <stdio.h>
int main(void)
{
    int number = 0;
    int *pNumber = NULL;
    number = 10;
    printf("\nAddress of the number : %p",&number);
    printf("\nNumber's value = %d",number);

    pNumber = &number;

    printf("\nAddress of the pNumber : %p",(void*)&pNumber);
    printf("\nSize of the pNumber : %zd bytes",sizeof(pNumber));
    printf("\npNumber's value : %p",pNumber);
    printf("\nValue pointed to :  %d",pNumber);


    return 0;
}
```

OUTPUT :

Address of the number : 0061FF1C

Number's value = 10

Address of the pNumber : 0061FF18

Size of the pNumber : 4 bytes

pNumber's value : 0061FF1C

19.

```
/*
Write a C program that swaps the values of two integers using pointers.
*/
#include <stdio.h>
int main()
{
```

```
    int A = 100,B=200;
    printf("Before Swapping : \n");
    printf("A= %d ; B = %d\n",A,B);
    int *pA = &A;
    int *pB = &B;
    int temp = *pA;
    *pA = *pB;
    *pB = temp;
    printf("\nAfter Swapping : \n A = %d ; B = %d",A,B);


    return 0;
}
```

OUTPUT :

Before Swapping :

A= 100 ; B = 200

After Swapping :

A = 200 ; B = 100

20.

```
//Example
#include <stdio.h>
int main(){
    long num1 = 0;
    long num2 = 0;
    long *pNum = NULL;
    pNum = &num1;
    *pNum = 2;
    ++num2;
    num2 += *pNum;
    pNum = &num2;
    ++*pNum;
   printf("num1: %ld, num2: %ld, *pNum: %ld, *pNum + num2 = %ld\n" , num1,
num2, *pNum, *pNum + num2 );


    return 0 ;
}
```

OUTPUT :

num1: 2, num2: 4, *pNum: 4, *pNum + num2 = 8

21.

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

```c
int main()
{
    int a;
    int *p = &a;
    if(p!=0){
        *p = 5;
    }
    printf("a=%d",a);
}
```

OUTPUT :

a=5