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);

int main(){
	int a = 10, b = 20;
	printf("001a = %p\n",&a);
	printf("001b = %p\n",&b);
	add_num(a,b);
	printf("The values of a and b is %d , %d",a,b);
	return 0;
}

void add_num(int a, int b){
	a = 40;
	b = 50;
```

```
        printf("002a = %p\n",&a);
        printf("002b = %p\n",&b);
        int sum =0;
        sum = a + b;
        printf("Sum = %d \n",sum);
}
```

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.

```c
#include <stdio.h>

void increment(int x)
{
    x = x + 1;
    printf("x = %d\n", x);
}
int main()
{
    int num = 5;
    printf("Before function call, num = %d\n", num);
    increment(num);
    printf("After function call, num = %d\n", num);
    return 0;
}
```

input

```
Before function call, num = 5
x = 6
After function call, num = 5


...Program finished with exit code 0
```

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.

```c
#include <stdio.h>

void swap(int x, int y)
{
    int temp = x;
    x = y;
    y = temp;
    printf("x = %d, y = %d\n", x, y);
}

int main()
{
    int a = 10, b = 20;
    printf("Before function call, a = %d, b = %d\n", a, b);
    swap(a, b);
    printf("After function call, a = %d, b = %d\n", a, b);
    return 0;
}
```

input

```
Before function call, a = 10, b = 20
x = 20, y = 10
After function call, a = 10, b = 20


...Program finished with exit code 0
```

3. Develop a C program that calculates the factorial of a number using call by value

```c
#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 a number to calculate its factorial: ");
    scanf("%d", &num);
    if (num < 0) {
        printf("Factorial is not defined for negative numbers.\n");
    }
    else
    {
        int result = factorial(num);
        printf("Factorial of %d is %d\n", num, result);
    }
    return 0;
}
```

```
Enter a number to calculate its factorial: 7
Factorial of 7 is 5040


...Program finished with exit code 0
```

4. Create a C program that defines a function to find the maximum of two numbers using call by value.

```c
#include <stdio.h>

int max(int x, int y)
{
    return (x > y) ? x : y;
}

int main()
{
    int a = 15, b = 20;
    int maximum = max(a, b);
    printf("The maximum of %d and %d is %d\n", a, b, maximum);
    return 0;
}
```

```
The maximum of 15 and 20 is 20


...Program finished with exit code 0
```

Problem Statement 5: 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
```

Program:

```
#include<stdio.h>
int addition(int x,int y)
{
    printf("addition: %d\n", x+y);
    return 0;
}
int subtraction(int x,int y)
{
    printf("subtraction: %d\n", x-y);
    return 0;
}
int multiplication(int x,int y)
{
    printf("multiplication: %d\n", x*y);
    return 0;
}
int division(int x,int y)
{
    printf("division: %d\n", x/y);
    return 0;
}
int main()
{
    int num1,num2;
    printf("enter the numbers: ");
    scanf("%d %d",&num1,&num2);
    int add=addition(num1,num2);
    int subtract=subtraction(num1,num2);
    int multiply=multiplication(num1,num2);
    int divide=division(num1,num2);
    return 0;
}
```

```
enter the numbers: 20 10
addition: 30
subtraction: 10
multiplication: 200
division: 2


...Program finished with exit code 0
```

Problem Statement 6: 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

Program:

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

```
float celsiusToFahrenheit(float celsius)
{
   return (celsius * 9 / 5) + 32;
}

float fahrenheitToCelsius(float fahrenheit)
{
   return (fahrenheit - 32) * 5 / 9;
}

int main()
{
   float tempC, tempF;
   printf("Enter temperature in Celsius: ");
   scanf("%f", &tempC);
   printf("Temperature in Fahrenheit: %.2f\n", celsiusToFahrenheit(tempC));
   printf("Enter temperature in Fahrenheit: ");
   scanf("%f", &tempF);
   printf("Temperature in Celsius: %.2f\n", fahrenheitToCelsius(tempF));

   return 0;
}
```

```
Enter temperature in Celsius: 30
Temperature in Fahrenheit: 86.00
Enter temperature in Fahrenheit: 95
Temperature in Celsius: 35.00


...Program finished with exit code 0
```

Problem Statement 7: 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

Program:

```
#include <stdio.h>
float SimpleInterest(float principal, float rate, float time)
{
   return (principal * rate * time) / 100;
}
int main()
{
   float p, r, t;
   printf("Enter principal amount: ");
   scanf("%f", &p);
   printf("Enter rate of interest: ");
   scanf("%f", &r);
   printf("Enter time period (in years): ");
   scanf("%f", &t);
   float interest = SimpleInterest(p, r, t);
   printf("Simple Interest is: %.2f\n", interest);
   return 0;
}
```

```
Enter principal amount: 3000
Enter rate of interest: 8
Enter time period (in years): 2
Simple Interest is: 480.00


...Program finished with exit code 0
```

8. Pointer - Exercise

1) Create a char type variable and initialize it to value 100

2) Print the address of the above variable.

3) Create a pointer variable and store the address of the above variable

4) Perform read operation on the pointer variable to fetch 1 byte of data from the pointera

5) Print the data obtained from the read operation on the pointer.

6) Perform write operation on the pointer to store the value 65

7) Print the value of the variable defined in step 1.

```c
#include<stdio.h>
int main()
{
    char a=100;
    printf("address: %p\n",(void *)&a);
    char *p=&a;
    char read=*p;
    printf("data read from pointer: %d\n",read);
    *p=65;
    printf("value of a: %d",a);
    return 0;
}
```

input

```
address: 0x7fff088d557e
data read from pointer: 100
value of a: 65

...Program finished with exit code 0
```

9. Another Example:

```c
#include<stdio.h>
int main(void)
{
    int number=0;
    int*pnumber=NULL;
    number=10;
    printf("number's address: %p\n",&number);
    printf("number's value: %d\n\n",number);
    pnumber=&number;
    printf("pnumber's address: %p\n",(void*)&pnumber);
    printf("pnumber's size: %zd bytes\n",sizeof(pnumber));
    printf("pnumber's value: %p\n",pnumber);
    printf("value pointed to: %d\n",*pnumber);
    return 0;
}
```

input

```
pnumber's size: 8 bytes
pnumber's value: 0x7fff09847eec
value pointed to: 10


...Program finished with exit code 0
```

10. write a c program that swaps the value of 2 integers using pointers

```c
#include <stdio.h>

void swap(int *x, int *y)
{
    int temp = *x;
    *x = *y;
    *y = temp;
}

int main()
{
    int a, b;
    printf("Enter the first integer: ");
    scanf("%d", &a);
    printf("Enter the second integer: ");
    scanf("%d", &b);
    printf("Before swapping: a = %d, b = %d\n", a, b);
    swap(&a, &b);
    printf("After swapping: a = %d, b = %d\n", a, b);

    return 0;
}
```

input

```
Enter the first integer: 10
Enter the second integer: 15
Before swapping: a = 10, b = 15
After swapping: a = 15, b = 10
```

11. write a c program to swap the number using swap function and follow the pass by reference method

```c
#include <stdio.h>

void swap(int *x, int *y)
{
    int temp = *x;
    *x = *y;
    *y = temp;
}

int main()
{
    int a, b;
    printf("Enter the first number: ");
    scanf("%d", &a);
    printf("Enter the second number: ");
    scanf("%d", &b);
    printf("Before swapping: a = %d, b = %d\n", a, b);
    swap(&a, &b);
    printf("After swapping: a = %d, b = %d\n", a, b);

    return 0;
}
```

input

```
Enter the first number: 7
Enter the second number: 3
Before swapping: a = 7, b = 3
After swapping: a = 3, b = 7
```

12. WAP for Finding the Cube of a Number Using Pass by Reference

```c
#include <stdio.h>

void findCube(int *num)
{
    *num = (*num) * (*num) * (*num);
}

int main()
{
    int number;
    printf("Enter a number: ");
    scanf("%d", &number);
    findCube(&number);
    printf("The cube of the number is: %d\n", number);

    return 0;
}
```

```
Enter a number: 7
The cube of the number is: 343


...Program finished with exit code 0
```

13. WAP to calculate the simple interest with the help of a function and pass call by reference method.

```c
#include <stdio.h>

void SimpleInterest(float *principal, float *rate, float *time, float *si)
{
    *si = (*principal) * (*rate) * (*time) / 100;
}

int main()
{
    float p, r, t, si;
    printf("Enter the principal amount: ");
    scanf("%f", &p);
    printf("Enter the rate of interest: ");
    scanf("%f", &r);
    printf("Enter the time: ");
    scanf("%f", &t);
    SimpleInterest(&p, &r, &t, &si);
    printf("The Simple Interest is: %.2f\n", si);

    return 0;
}
```

input

```
Enter the principal amount: 2000
Enter the rate of interest: 8
Enter the time: 2
The Simple Interest is: 320.00


...Program finished with exit code 0
```