

Function Implemmentation

=====

1. Function Prototype (Function Declaration)

which is defined before the main() function

Syntax: return_type function_name(If you are passing parameter, mention the datatypes of the parameter);

2. Function call

This is implemented inside the main() function

Syntax: function_name(pass the variable only the variable names);

3. Function Definition

This is implemented after the main() Function

Syntax:

return_type function_name(If you are passing parameter, 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);
```

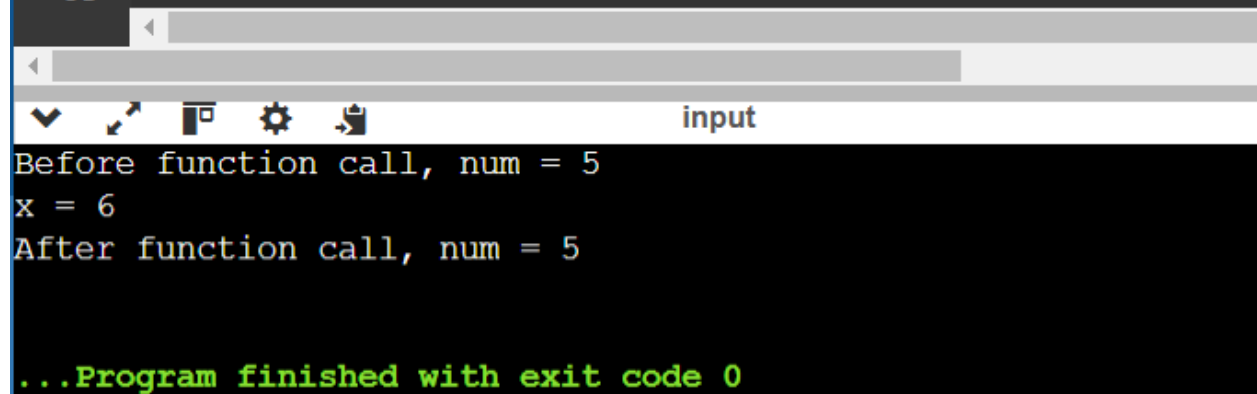
```
int main(){
    int a = 10, b = 20;
    printf("001a = %p\n",&a);
    printf("001b = %p\n",&b);
    add_num(a,b);
    printf("The values of a and b is %d , %d",a,b);
    return 0;
}
```

```
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.

```
47 #include <stdio.h>
48
49 void increment(int x)
50 {
51     x = x + 1;
52     printf("x = %d\n", x);
53 }
54 int main()
55 {
56     int num = 5;
57     printf("Before function call, num = %d\n", num);
58     increment(num);
59     printf("After function call, num = %d\n", num);
60     return 0;
61 }
62
63
```



```
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.

```
47 #include <stdio.h>
48
49 void swap(int x, int y)
50 {
51     int temp = x;
52     x = y;
53     y = temp;
54     printf("x = %d, y = %d\n", x, y);
55 }
56
57 int main()
58 {
59     int a = 10, b = 20;
60     printf("Before function call, a = %d, b = %d\n", a, b);
61     swap(a, b);
62     printf("After function call, a = %d, b = %d\n", a, b);
63     return 0;
64 }
65
66
```

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

```
#include <stdio.h>

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

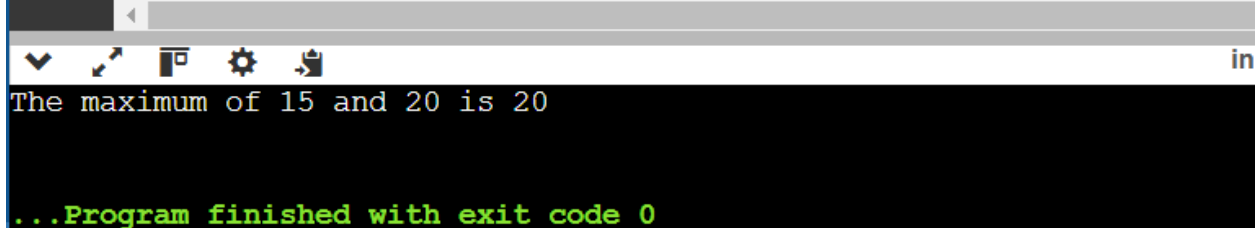
int main()
{
    int num;
    printf("Enter a number to calculate its factorial: ");
    scanf("%d", &num);
    if (num < 0) {
        printf("Factorial is not defined for negative numbers.\n");
    }
    else
    {
        int result = factorial(num);
        printf("Factorial of %d is %d\n", num, result);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

```
78  #include <stdio.h>
79
80  int max(int x, int y)
81  {
82      return (x > y) ? x : y;
83  }
84
85  int main()
86  {
87      int a = 15, b = 20;
88      int maximum = max(a, b);
89      printf("The maximum of %d and %d is %d\n", a, b, maximum);
90      return 0;
91  }
92
```



The screenshot shows a terminal window with a dark background. The top part displays the C code for finding the maximum of two numbers. The code defines a function `max` that takes two integers `x` and `y` and returns the larger one using a ternary operator. The `main` function calls `max` with `a = 15` and `b = 20`, and prints the result. The bottom part of the terminal shows the program's output: "The maximum of 15 and 20 is 20" followed by "...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.

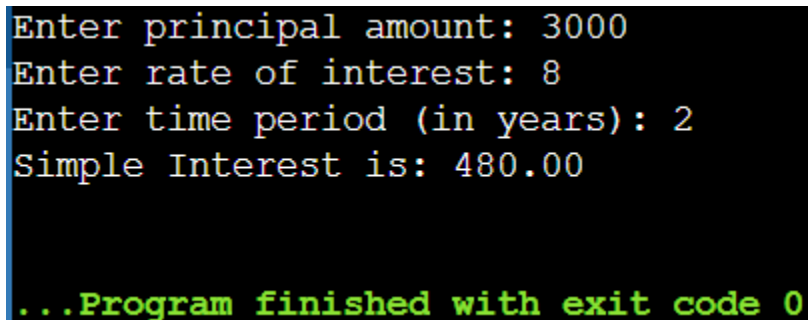
Use appropriate data types for financial calculations (e.g., float or double).
Utilize arithmetic operators to compute simple interest using the formula
 $SI = P \times R \times T / 100$

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;
}
```



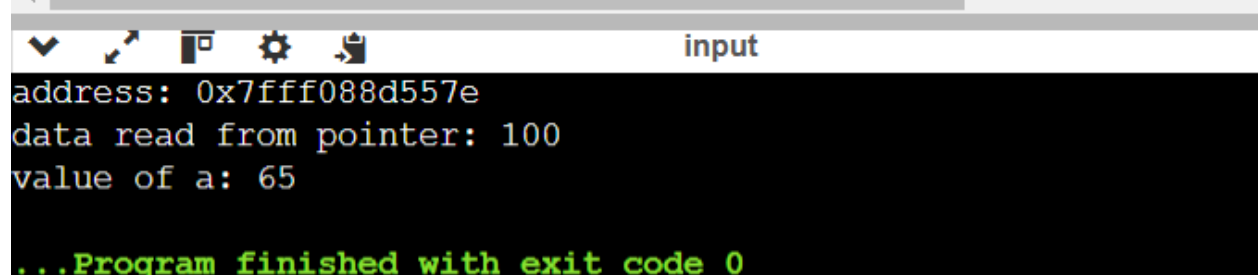
```
Enter principal amount: 3000
Enter rate of interest: 8
Enter time period (in years): 2
Simple Interest is: 480.00

...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 pointer
- 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.

```
224 #include<stdio.h>
225 int main()
226 {
227     char a=100;
228     printf("address: %p\n",(void *)&a);
229     char *p=&a;
230     char read=*p;
231     printf("data read from pointer: %d\n",read);
232     *p=65;
233     printf("value of a: %d",a);
234     return 0;
235 }
236
237
238
239
240
```



The screenshot shows a terminal window with a dark background. At the top, there is a toolbar with icons for a dropdown menu, a cursor, a terminal window, a gear (settings), and a clipboard. To the right of the toolbar is the word "input". Below the toolbar, the terminal output is displayed in a monospaced font. The output shows the memory address of variable 'a' as 0x7fff088d557e, the data read from the pointer as 100, and the value of 'a' after being updated to 65. At the bottom, a green message indicates the program finished with exit code 0.

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

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

9. Another Example:

```

224 #include<stdio.h>
225 int main(void)
226 {
227     int number=0;
228     int*pnumber=NULL;
229     number=10;
230     printf("number's address: %p\n",&number);
231     printf("number's value: %d\n\n",number);
232     pnumber=&number;
233     printf("pnumber's address: %p\n",(void*)&pnumber);
234     printf("pnumber's size: %zd bytes\n",sizeof(pnumber));
235     printf("pnumber's value: %p\n",pnumber);
236     printf("value pointed to: %d\n",*pnumber);
237     return 0;
238 }
239
240
241
242
243

```

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

```

241 #include <stdio.h>
242
243 void swap(int *x, int *y)
244 {
245     int temp = *x;
246     *x = *y;
247     *y = temp;
248 }
249
250 int main()
251 {
252     int a, b;
253     printf("Enter the first integer: ");
254     scanf("%d", &a);
255     printf("Enter the second integer: ");
256     scanf("%d", &b);
257     printf("Before swapping: a = %d, b = %d\n", a, b);
258     swap(&a, &b);
259     printf("After swapping: a = %d, b = %d\n", a, b);
260
261     return 0;
262 }
263
264

```

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

```

265 #include <stdio.h>
266
267 void swap(int *x, int *y)
268 {
269     int temp = *x;
270     *x = *y;
271     *y = temp;
272 }
273
274 int main()
275 {
276     int a, b;
277     printf("Enter the first number: ");
278     scanf("%d", &a);
279     printf("Enter the second number: ");
280     scanf("%d", &b);
281     printf("Before swapping: a = %d, b = %d\n", a, b);
282     swap(&a, &b);
283     printf("After swapping: a = %d, b = %d\n", a, b);
284
285     return 0;
286 }
287
288
289
290

```

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

```

292 #include <stdio.h>
293
294 void findCube(int *num)
295 {
296     *num = (*num) * (*num) * (*num);
297 }
298
299 int main()
300 {
301     int number;
302     printf("Enter a number: ");
303     scanf("%d", &number);
304     findCube(&number);
305     printf("The cube of the number is: %d\n", number);
306
307     return 0;
308 }
309
310
311
312

```

```

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.

```

311 #include <stdio.h>
312
313 void SimpleInterest(float *principal, float *rate, float *time, float *si)
314 {
315     *si = (*principal) * (*rate) * (*time) / 100;
316 }
317
318 int main()
319 {
320     float p, r, t, si;
321     printf("Enter the principal amount: ");
322     scanf("%f", &p);
323     printf("Enter the rate of interest: ");
324     scanf("%f", &r);
325     printf("Enter the time: ");
326     scanf("%f", &t);
327     SimpleInterest(&p, &r, &t, &si);
328     printf("The Simple Interest is: %.2f\n", si);
329
330     return 0;
331 }
332
333
334
335

```

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

```