

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      //function declarations
6      void MathematicalOperations(int, int, int *, int *, int *, int *, int *);
7
8      //variable declaration
9      int a;
10     int b;
11     int answer_sum;
12     int answer_difference;
13     int answer_product;
14     int answer_quotient;
15     int answer_remainder;
16
17     //code
18     printf("\n\n");
19     printf("Enter Value Of 'A' : ");
20     scanf("%d", &a);
21
22     printf("\n\n");
23     printf("Enter Value Of 'B' : ");
24     scanf("%d", &b);
25
26     // PASSING ADDRESSES TO FUNCTION ... FUNCTION WILL FILL THEM UP WITH
27     VALUES ... HENCE, THEY GO INTO THE FUNCTION AS ADDRESS PARAMETERS AND COME
28     OUT OF THE FUNCTION FILLED WITH VALID VALUES
29
30     // THUS, (&answer_sum, &answer_difference, &answer_product, &answer_quotient,
31     &answer_remainder) ARE CALLED "OUT PARAMETERS" OR "PARAMETERIZED RETURN
32     VALUES" ... RETURN VALUES OF FUNCTIONS COMING VIA PARAMETERS
33
34     // HENCE, ALTHOUGH EACH FUNCTION HAS ONLY ONE RETURN VALUE, USING THE CONCEPT
35     OF "PARAMETERIZED RETURN VALUES", OUR FUNCTION "MathematicalOperations()"
36     HAS GIVEN US 5 RETURN VALUES !!!
37
38     MathematicalOperations(a, b, &answer_sum, &answer_difference, &answer_product,
39     &answer_quotient, &answer_remainder);
40
41     printf("\n\n");
42     printf("***** RESULTS ***** : \n\n");
43     printf("Sum = %d\n\n", answer_sum);
44     printf("Difference = %d\n\n", answer_difference);
45     printf("Product = %d\n\n", answer_product);
46     printf("Quotient = %d\n\n", answer_quotient);
47     printf("Remainder = %d\n\n", answer_remainder);
48     return(0);
49 }
50
51 void MathematicalOperations(int x, int y, int *sum, int *difference, int *product,
52 int *quotient, int *remainder)
53 {
54     //code
```

```
45     *sum = x + y;           // Value at address 'sum' = (x + y)
46     *difference = x - y;    // Value at address 'difference' = (x - y)
47     *product = x * y;       // Value at address 'product' = (x * y)
48     *quotient = x / y;      // Value at address 'quotient' = (x / y)
49     *remainder = x % y;     // Value at address 'remainder' = (x % y)
50 }
51
```