```c
#include <stdio.h>

int main(void)
{
    //function declarations
    void MathematicalOperations(int, int, int *, int *, int *, int *, int *);

    //variable declaration
    int a;
    int b;
    int answer_sum;
    int answer_difference;
    int answer_product;
    int answer_quotient;
    int answer_remainder;

    //code
    printf("\n\n");
    printf("Enter Value Of 'A' : ");
    scanf("%d", &a);

    printf("\n\n");
    printf("Enter Value Of 'B' : ");
    scanf("%d", &b);

    // PASSING ADDRESSES TO FUNCTION ... FUNCTION WILL FILL THEM UP WITH VALUES ... HENCE, THEY GO INTO THE FUNCTION AS ADDRESS PARAMETERS AND COME OUT OF THE FUNCTION FILLED WITH VALID VALUES
    // THUS, (&answer_sum, &answer_difference, &answer_product, &answer_quotient, &answer_remainder) ARE CALLED "OUT PARAMETERS" OR "PARAMETERIZED RETURN VALUES" ... RETURN VALUES OF FUNCTIONS COMING VIA PARAMETERS
    // HENCE, ALTHOUGH EACH FUNCTION HAS ONLY ONE RETURN VALUE, USING THE CONCEPT OF "PARAMETERIZED RETURN VALUES", OUR FUNCTION "MathematicalOperations()" HAS GIVEN US 5 RETURN VALUES !!!

    MathematicalOperations(a, b, &answer_sum, &answer_difference, &answer_product, &answer_quotient, &answer_remainder);

    printf("\n\n");
    printf("****** RESULTS ****** : \n\n");
    printf("Sum = %d\n\n", answer_sum);
    printf("Difference = %d\n\n", answer_difference);
    printf("Product = %d\n\n", answer_product);
    printf("Quotient = %d\n\n", answer_quotient);
    printf("Remainder = %d\n\n", answer_remainder);
    return(0);
}

void MathematicalOperations(int x, int y, int *sum, int *difference, int *product, int *quotient, int *remainder)
{
    //code
    *sum = x + y;          // Value at address 'sum' = (x + y)
    *difference = x - y;   // Value at address 'difference' = (x - y)
    *product = x * y;      // Value at address 'product' = (x * y)
    *quotient = x / y;     // Value at address 'quotient' = (x / y)
    *remainder = x % y;    // Value at address 'remainder' = (x % y)
}

```