```c
#include <stdio.h>

enum
{
    NEGATIVE = -1,
    ZERO,
    POSITIVE
};

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

    //variable declaration
    int a;
    int b;
    int answer, ret;

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

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

    ret = Difference(a, b, &answer);

    printf("\n\n");
    printf("Difference Of %d And %d = %d\n\n", a, b, answer);

    if (ret == POSITIVE)
        printf("The Difference Of %d And %d Is Positive !!!\n\n", a, b);

    else if (ret == NEGATIVE)
        printf("The Difference Of %d And %d Is Negative !!!\n\n", a, b);

    else
        printf("The Difference Of %d And %d Is Zero !!!\n\n", a, b);

    return(0);
}

// WE WANT OUR FUNCTION Difference() TO PERFORM 2 JOBS ...
// ONE, IS TO SUBTRACT THE INPUT NUMBERS ('y' FROM 'x') AND THE SECOND, IS TO TELL WHETHER THE DIFFERENCE OF 'x' AND 'y' IS POSITIVE or NEGATIVE or ZERO ...
// BUT ANY FUNCTION HAS ONLY ONE VALID RETURN VALUE, THEN HOW CAN WE MANAGE TO RETURN TWO VALUES TO THE CALLING FUNCTION?
// THIS IS WHERE PARAMETERIZED RETURN VALUE COMES INTO THE PICTURE ...
// WE CAN RETURN THE ACTUAL DIFFERENCE OF 'x' AND 'y', THAT IS, THE ACTUAL ANSWER
```

```c
    VALUE, VIA OUT-PARAMETER / PARAMETERIZED RETURN VALUE
// AND WE CAN RETURN THE STATUS OF THE ANSWER (POSITIVE / NEGATIVE / ZERO) VIA THE ACTUAL RETURN VALUE OF THE FUNCTION ...

int Difference(int x, int y, int *diff)
{
    //code
    *diff = x - y;

    if (*diff > 0)
        return(POSITIVE);

    else if (*diff < 0)
        return(NEGATIVE);

    else
        return(ZERO);
}


```