```c
#include <stdio.h>

int main(void)
{
    //variable declarations
    int num = 5;
    const int* ptr = NULL; // Read this line from right to left => "ptr is a pointer (*) to integer (int) constant (const)."

    //code
    ptr = &num;
    printf("\n");
    printf("Current Value Of 'num' = %d\n", num);
    printf("Current 'ptr' (Address of 'num') = %p\n", ptr);

    // The following line does NOT give error ... as we are modifying the value of the variable individually
    num++;
    printf("\n\n");
    printf("After num++, value of 'num' = %d\n", num);


    // The following line gives error and is hence commented out.
    // We cannot alter the value stored in "A pointer to constant integer"
    // With respect to the pointer, the value it points to should be constant.
    // Uncomment it and see the error.

    //  (*ptr)++;

    // The following line does NOT give error
    // We do not get error because we are changing the pointer (address).
    // The pointer is not constant. The value to which the pointer points is constant.
    ptr++;

    printf("\n\n");
    printf("After ptr++, value of 'ptr' = %p\n", ptr);
    printf("Value at this new 'ptr' = %d\n", *ptr);
    printf("\n");
    return(0);
}

// CONCLUSION :
// As "ptr" is a "variable pointer to constant integer" - we can change the value of pointer "ptr".
// We can change the value of the variable (num) individually - whose address is contained in "ptr".
// So in short, we cannot change "the value at address of ptr" - we cannot change the value of "num" with respect to "ptr" => (*ptr)++ is NOT allowed.
// But, we can change the address 'ptr' itself => So, ptr++ is allowed.


```