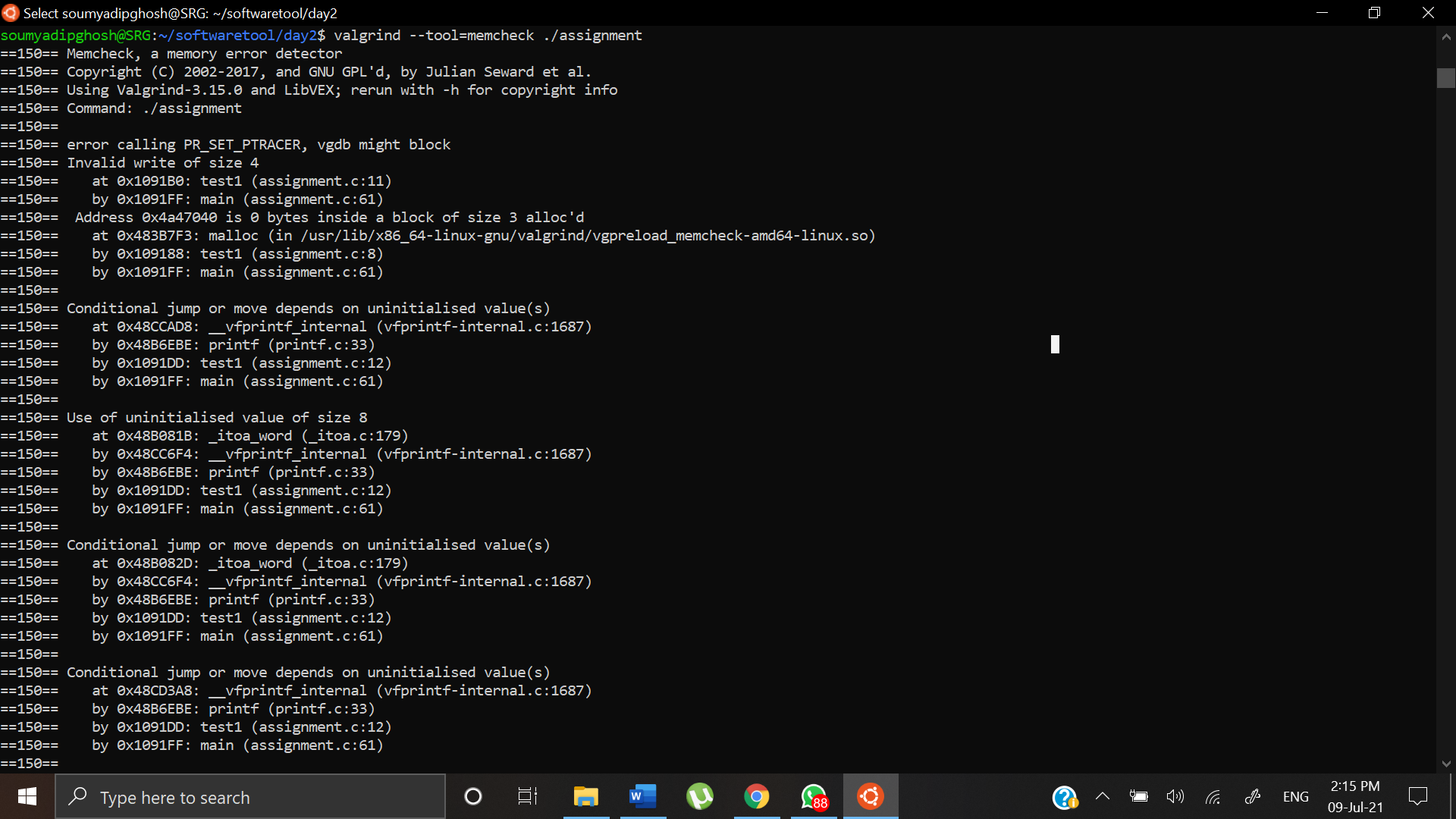
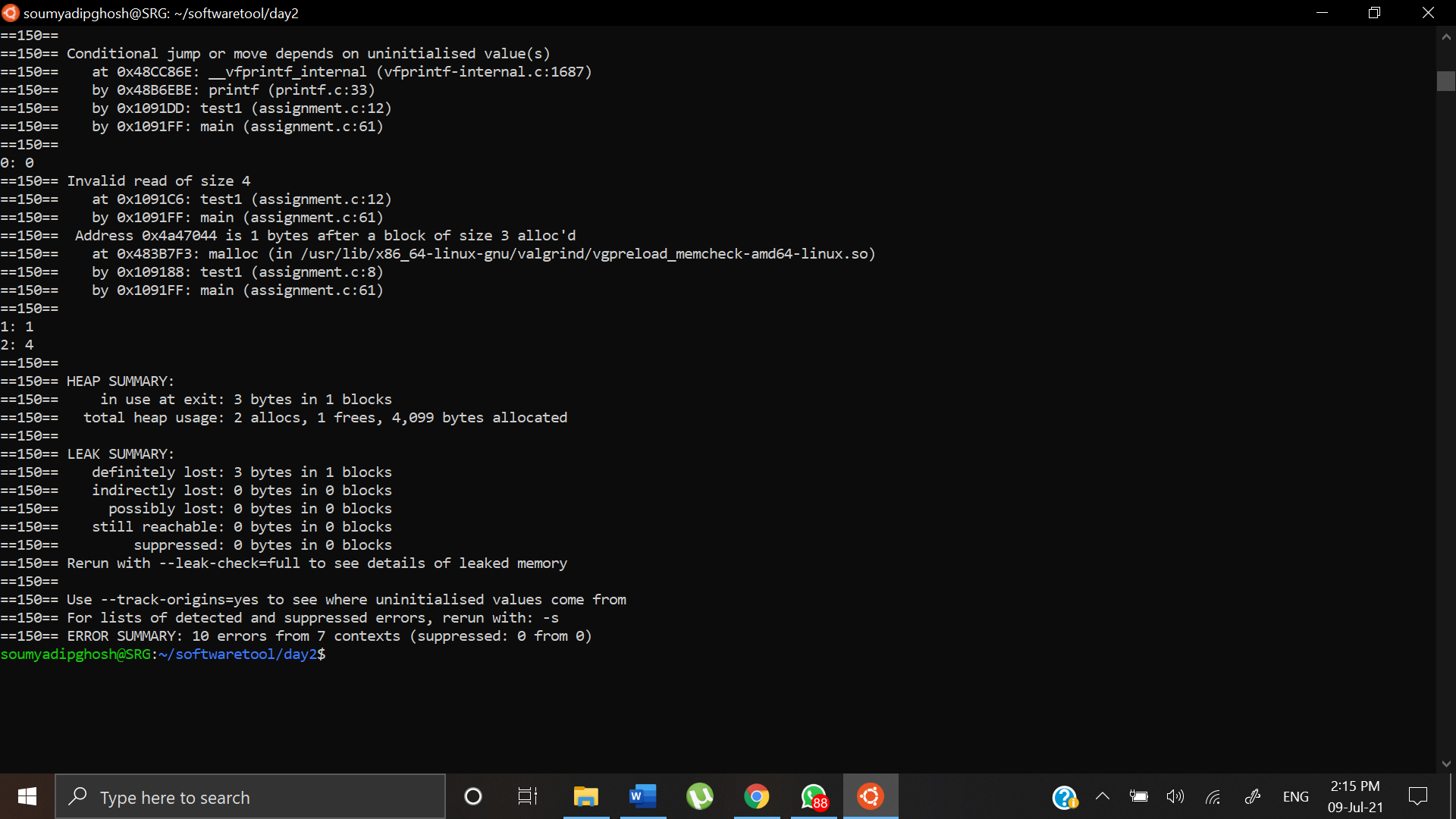
Assignment

Day-2

Subject: Software Tools

test1 function:





Here we can see there is a problem in line number 11 of the program inside the test function. The line number 11 is:

height[i]=i\*i;

The assignment is becoming beyond memory bounds for the array. The bug is that we have assigned only 3 bytes block of memory to the array: height and in line 11 we are assigning an integer of block memory 4 bytes to an array with block memory 3 bytes.

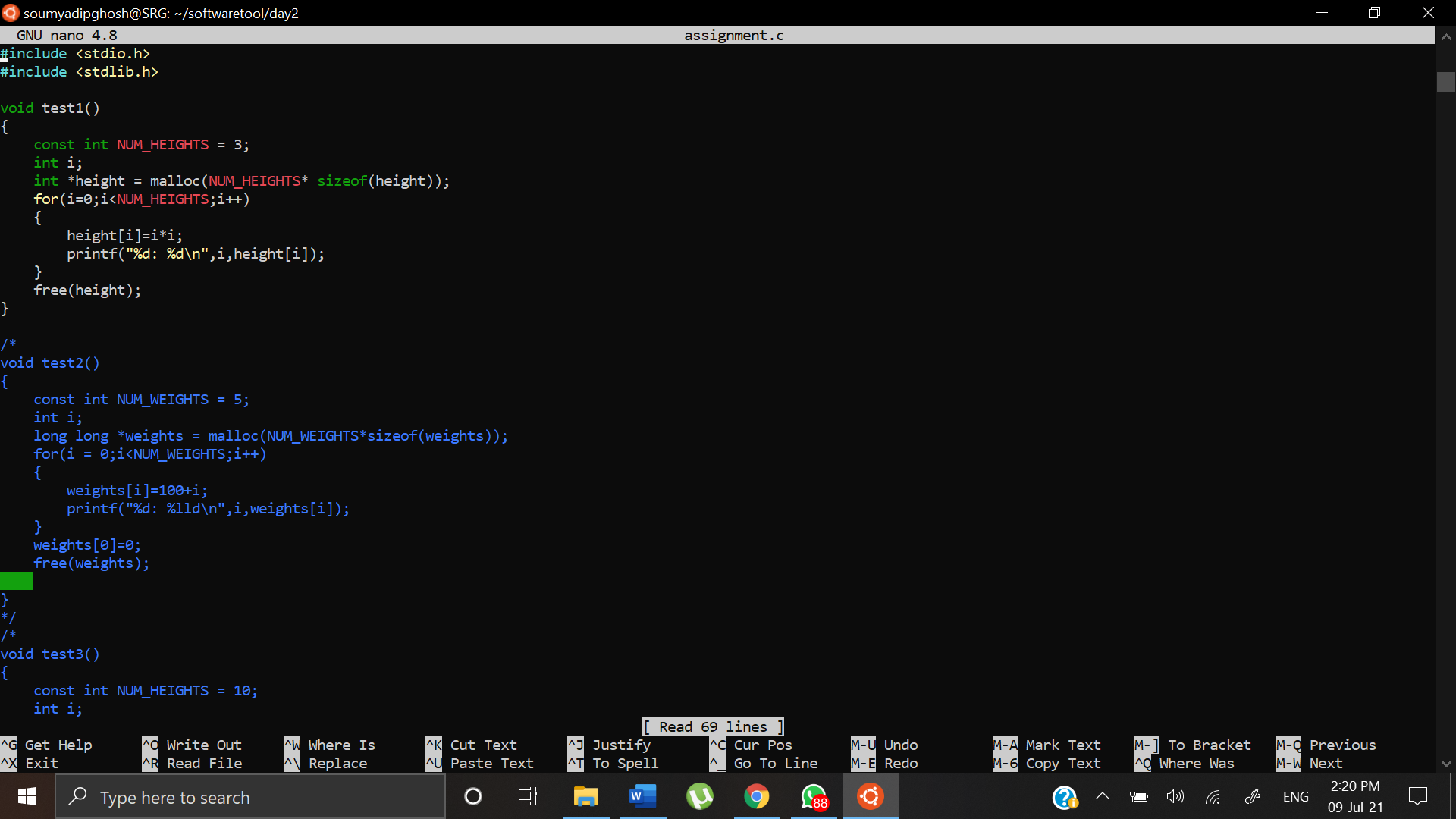
int \*height = malloc(NUM\_HEIGHTS);

We have not allocated memory properly for the array. Hence the printf function will fail to print arr[0] or arr[1] or arr[2] because they remain uninitialized.

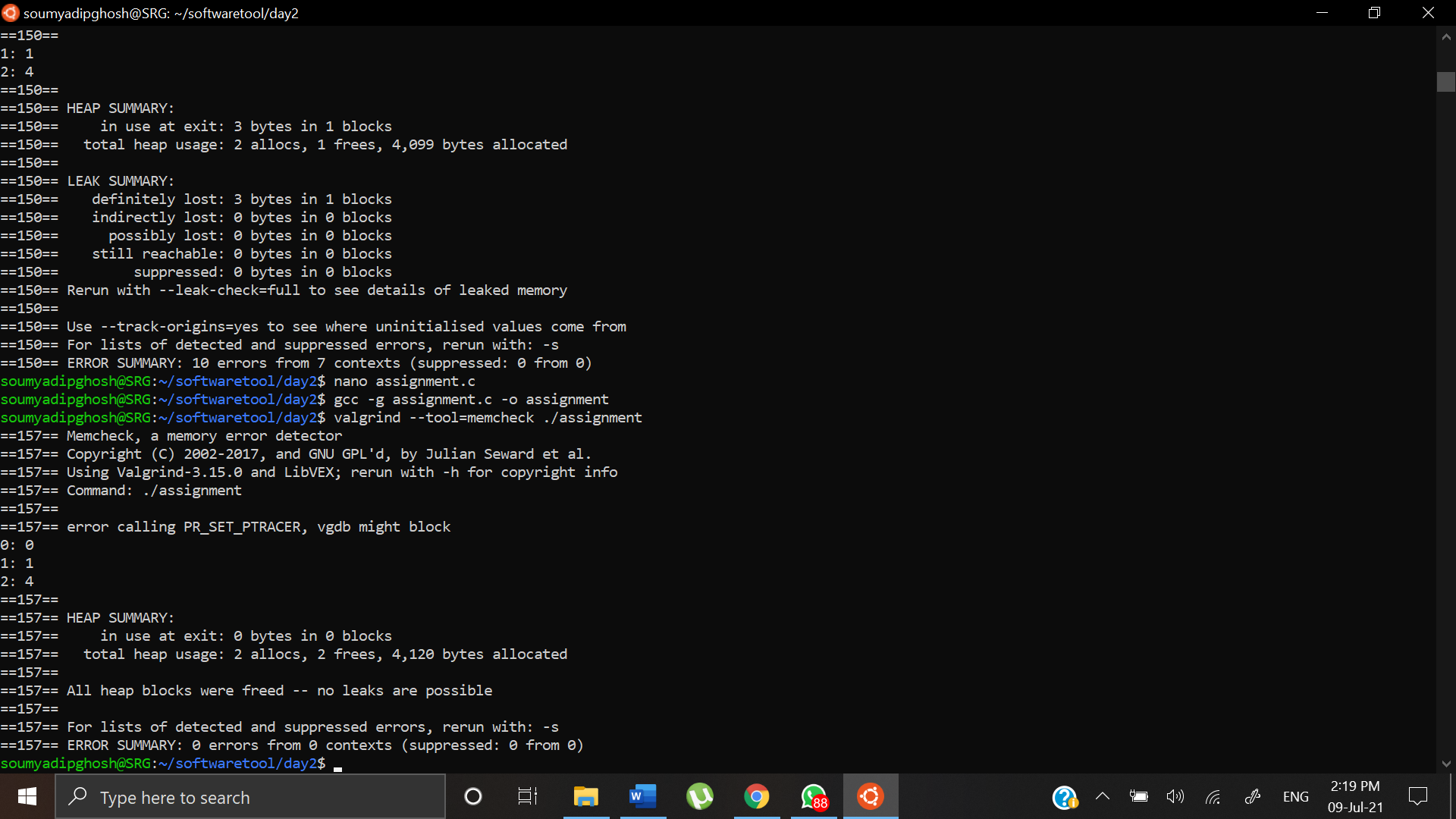
Also, in heap summary we can see that data leak is occurring because we have not freed the dynamically allocated memory.

After correction:

Code:

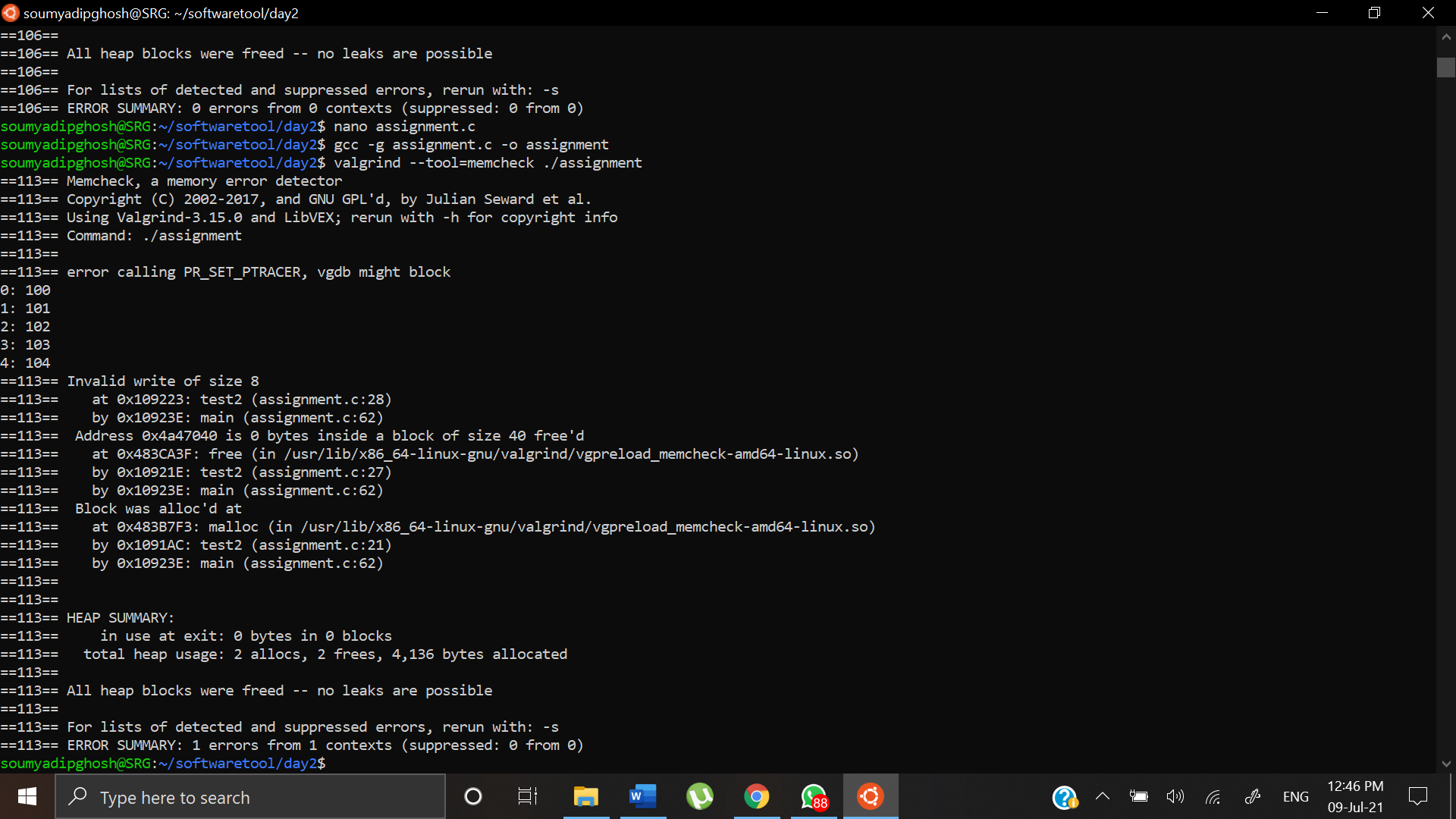


Valgrind Output:



Hence all the errors were freed.

test2 function:



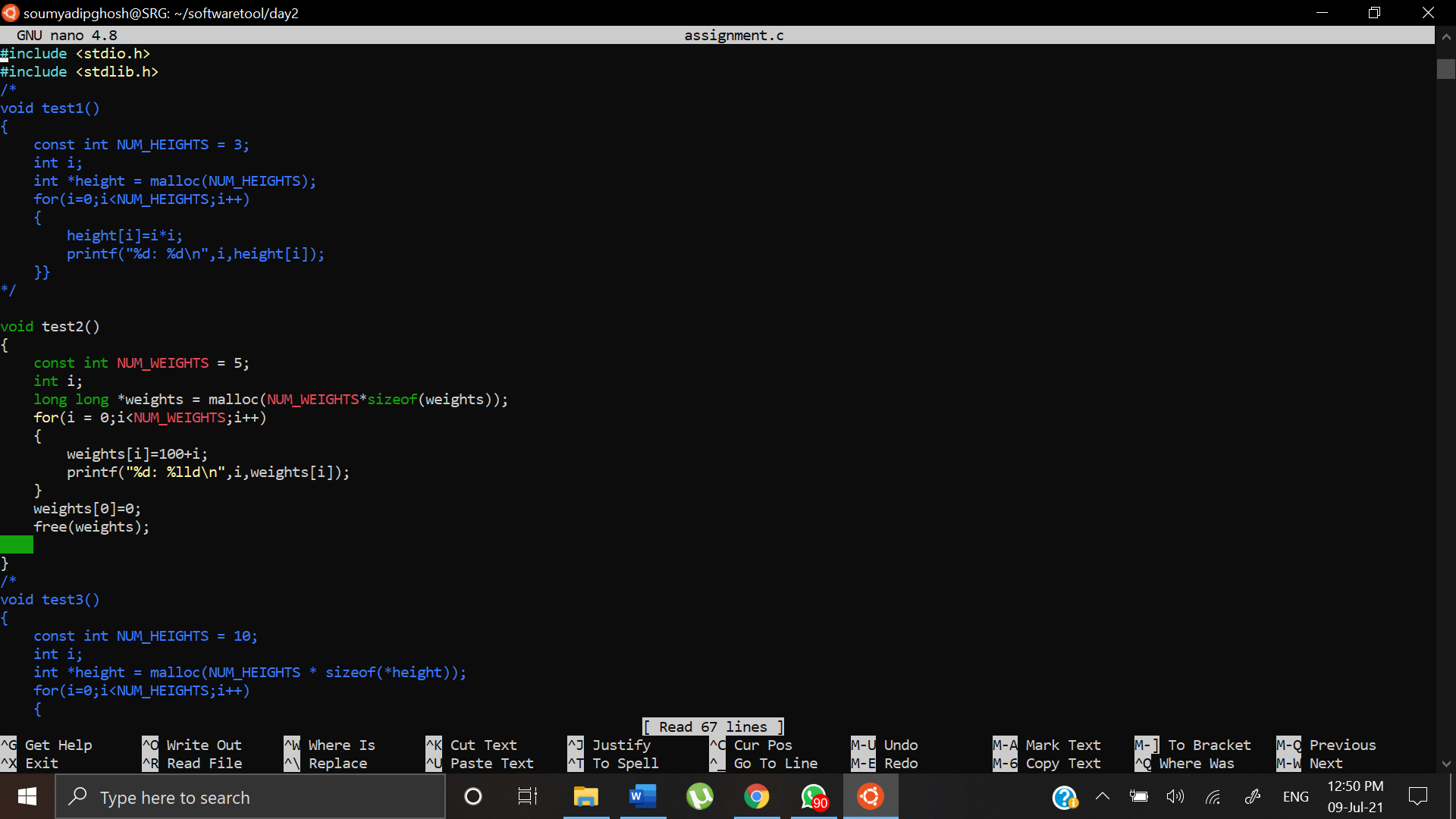
It tells us that there is a problem in line 28:

weights[0]= 0;

The error tells us that we are trying to allocate something into a 40 bytes memory block that was already freed. Hence by checking the code we can see that we are assigning 0 to the 0 bytes position of weight array that was freed.

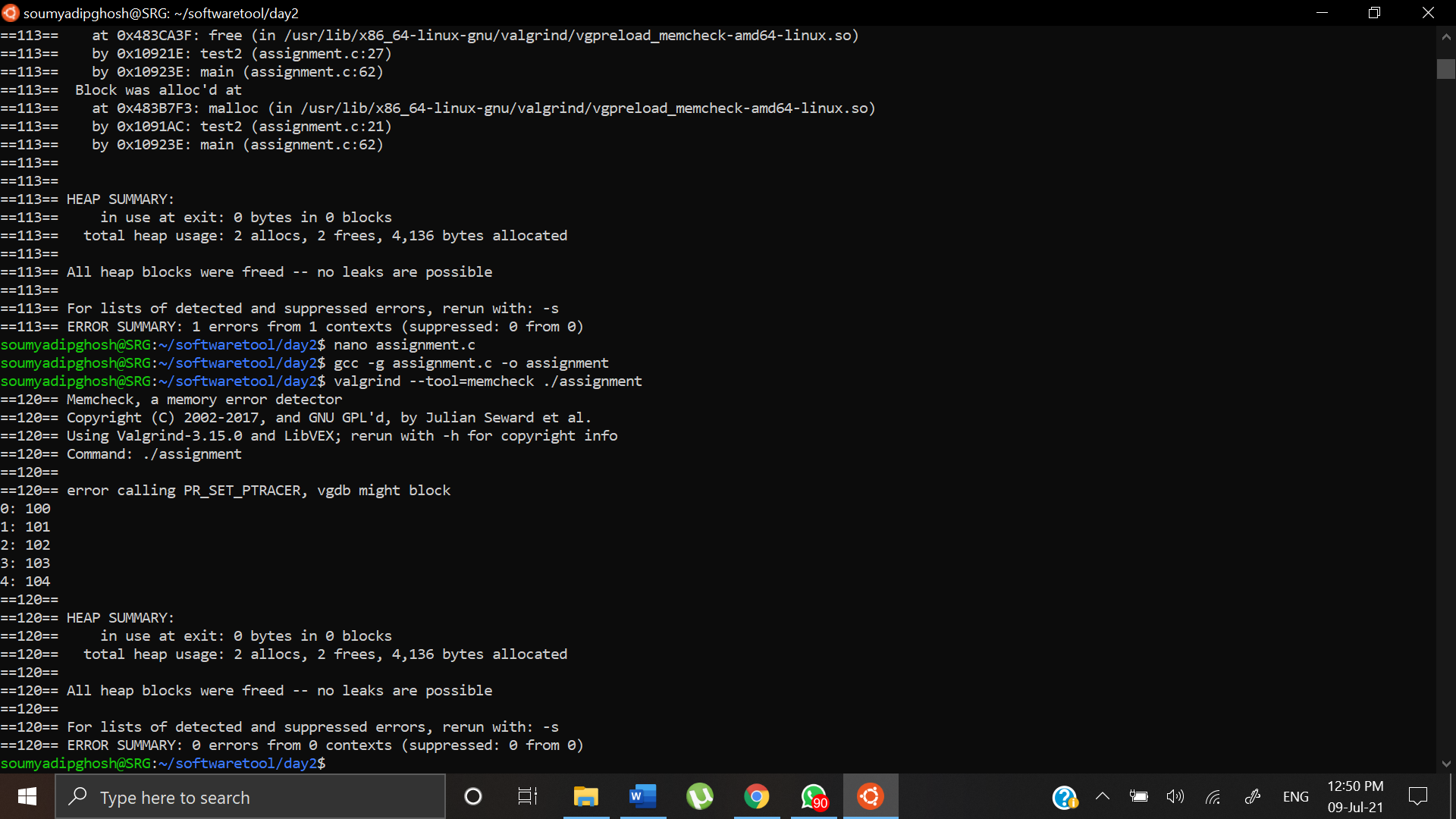
After Correction:

Code:



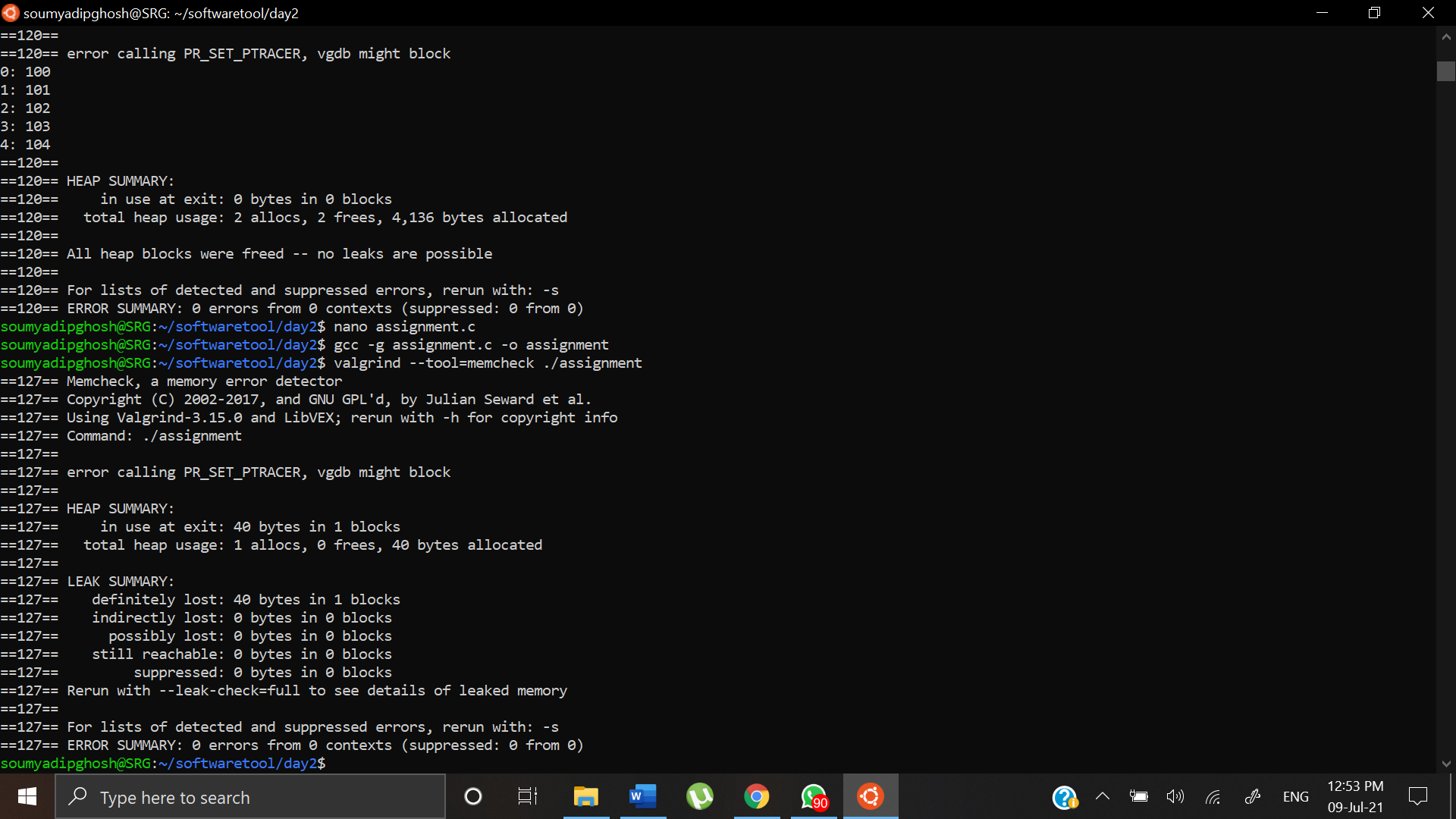
We just moved weights[0] to the line before we free(weights).

Valgrind Output:

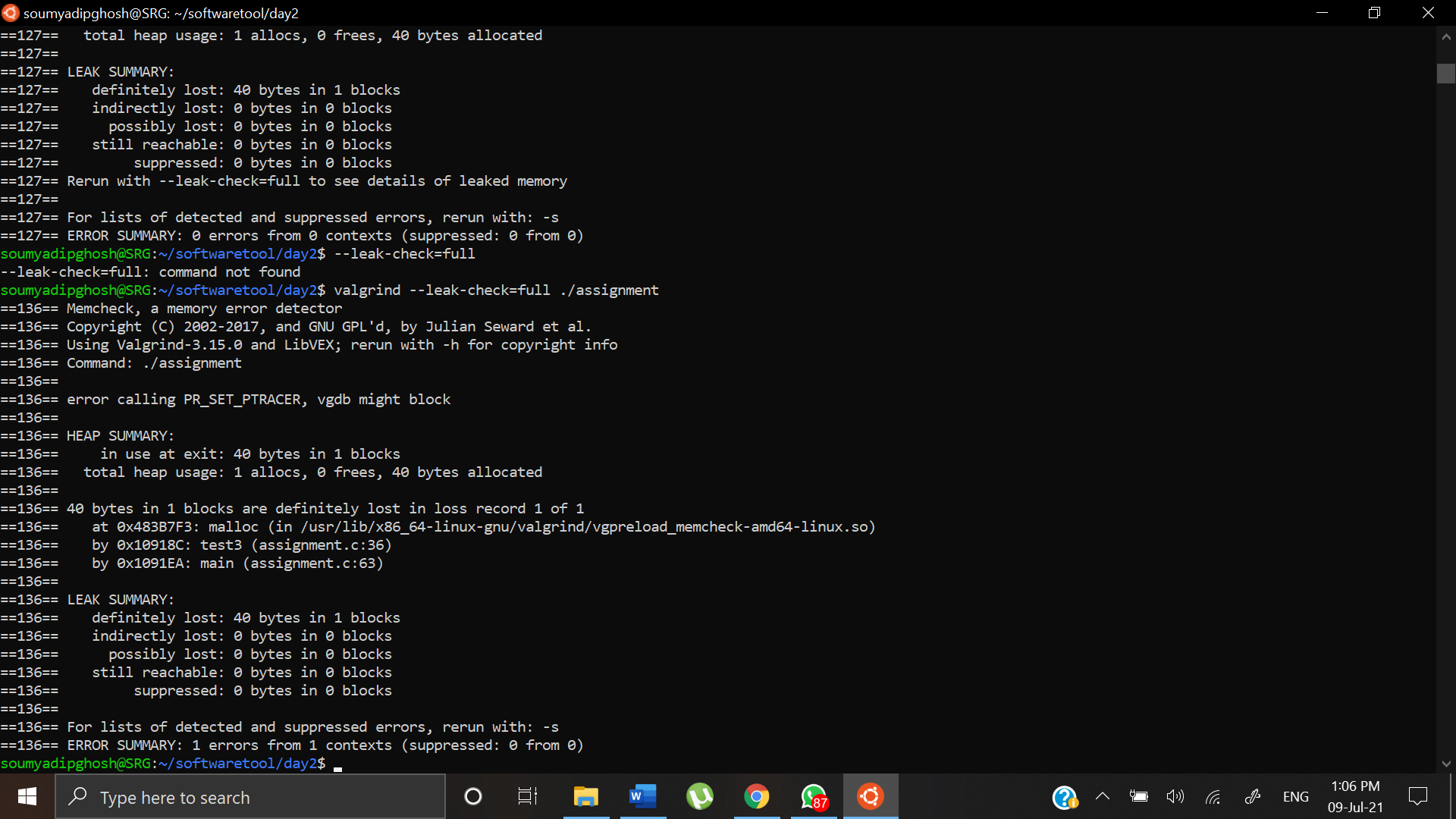


Hence test2 is error free.

test3 function:



This does not tell us much about where the error occurred. Hence we check the error again using valgrind --leak-check=full ./assignment as stated in the output screen above.



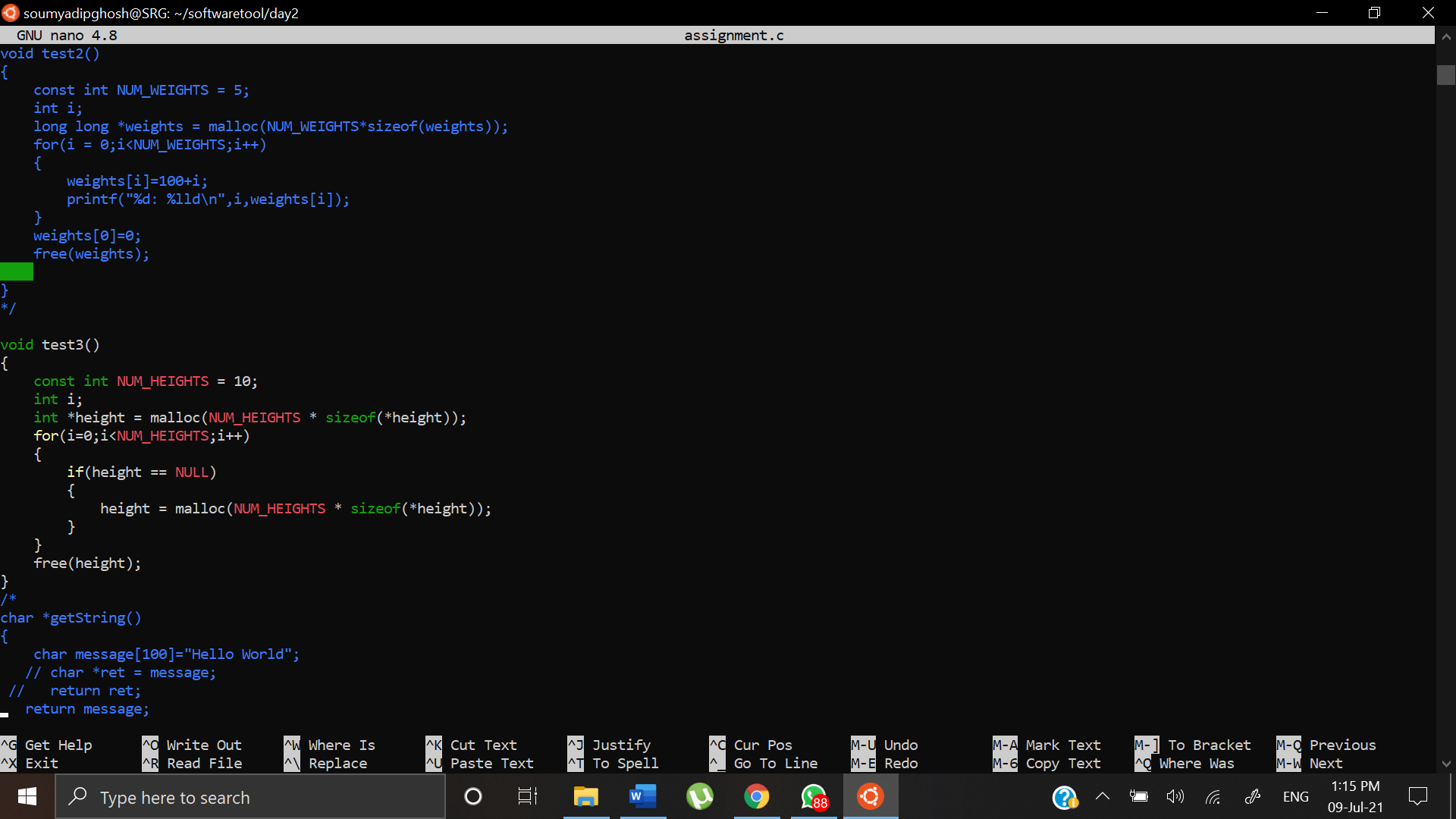
We can see there is an error in line 36.

if((height = NULL))

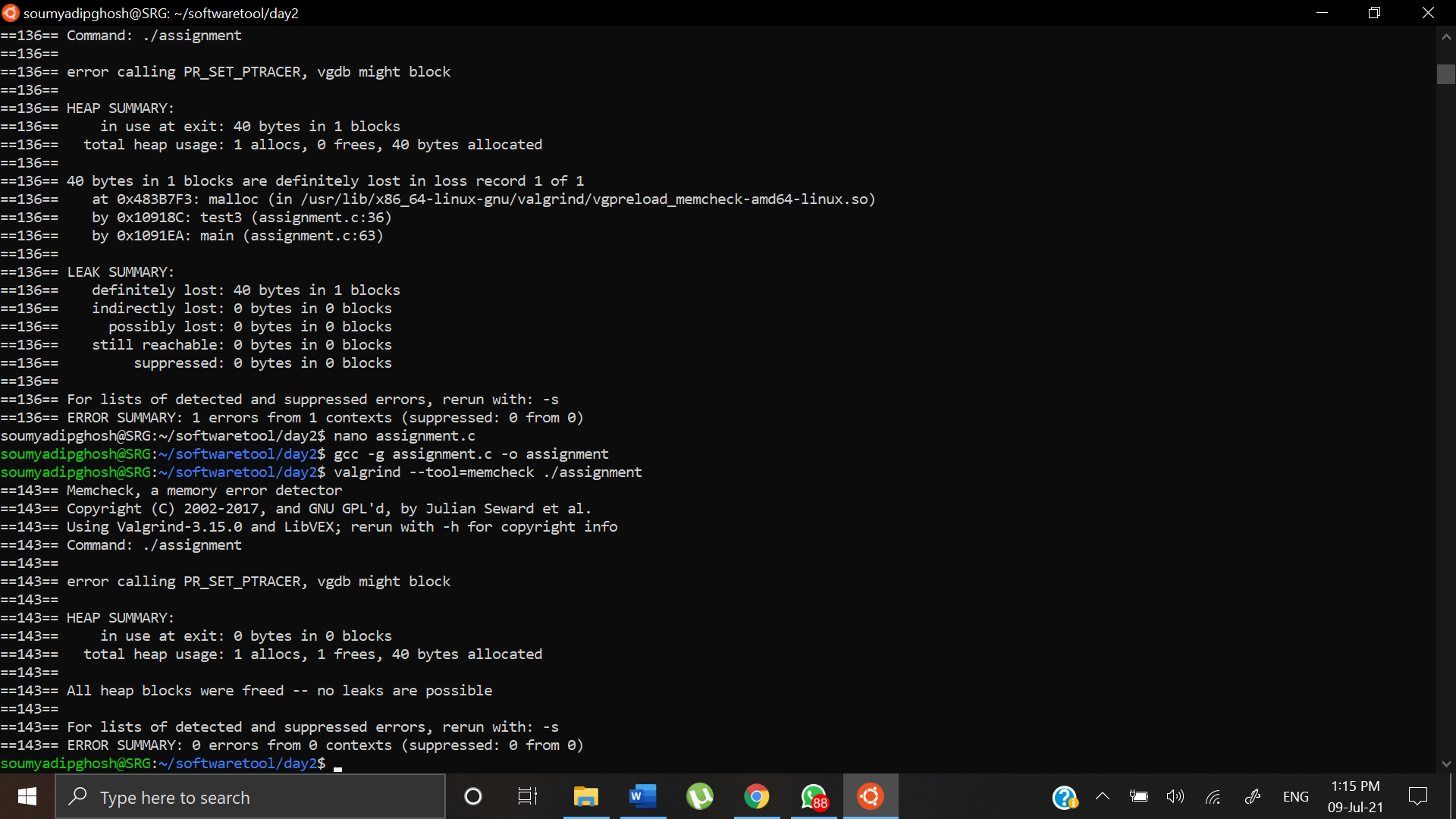
Here we are assigning height = NULL inside a if statement instead of checking height==NULL. That’s why the error shows that 40 bytes are lost.

After correction:

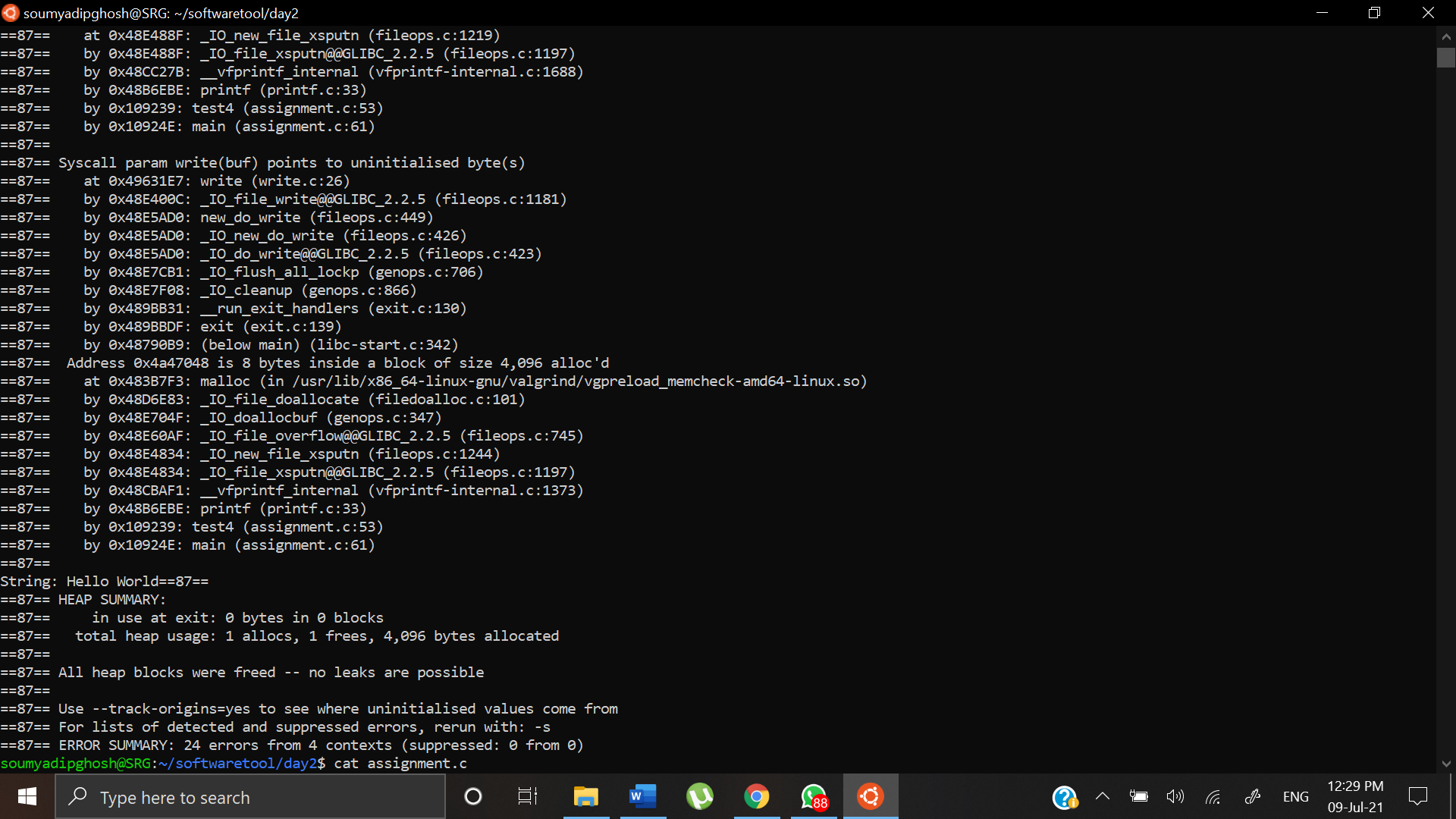
The code:



Valgrind Output:



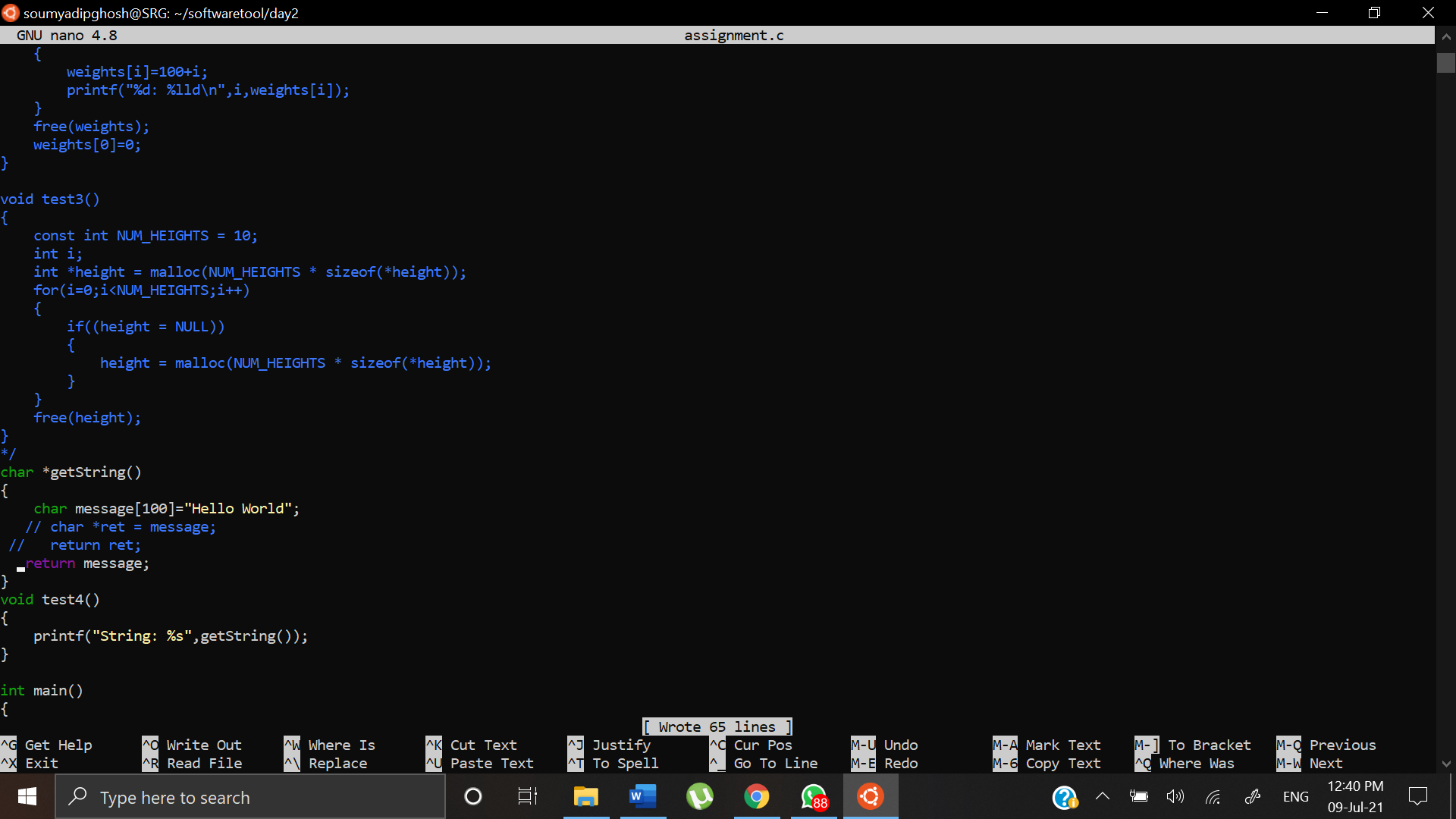
test4 function:



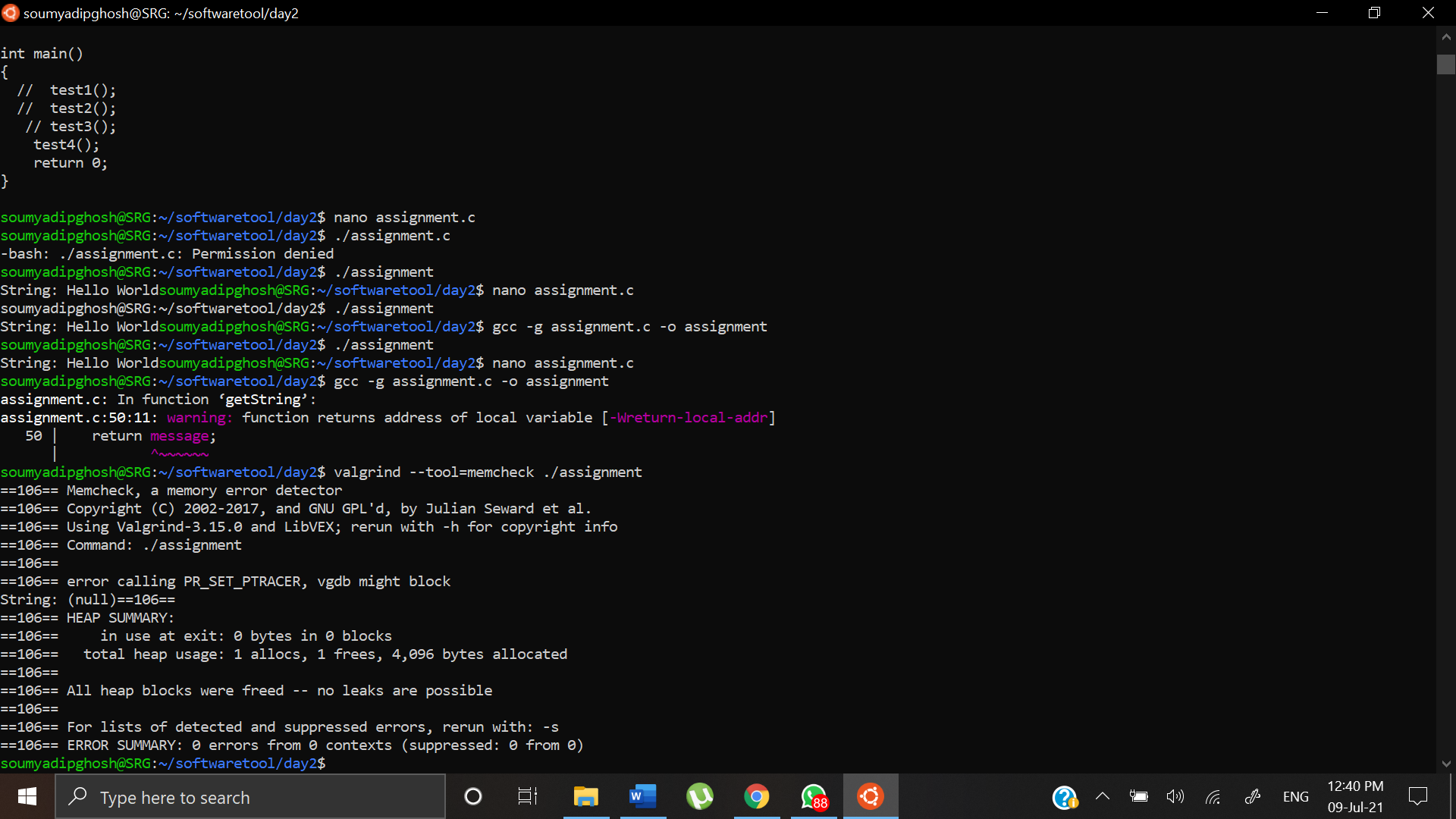
A possible error that might occur is due to the pointer declared locally which is pointing to the base address of the array. When the control comes out of the function block, stack memory is released and hence the pointer may get some garbage values. Hence, we can directly return message string, instead of assigning a pointer and returning. That would eliminate the error.

After Correction:

Corrected Code



Valgrind Output:



No such errors were found.