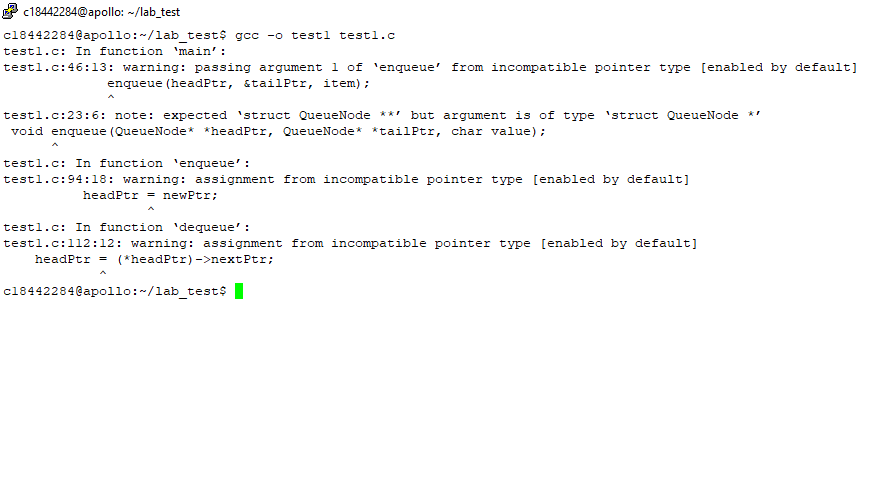
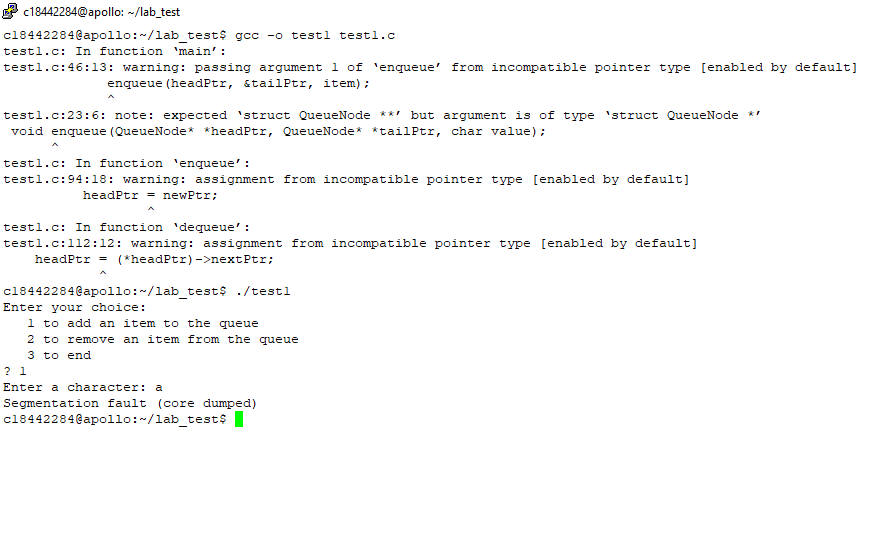
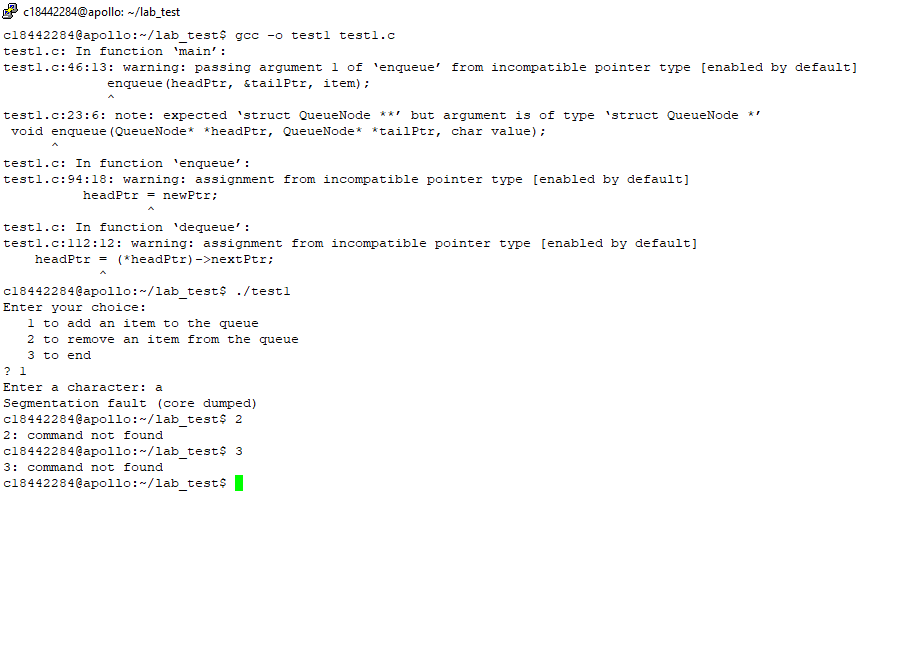
**NAME: AMIR AKBARI DT228/2 Labtest1 LabGroup A**

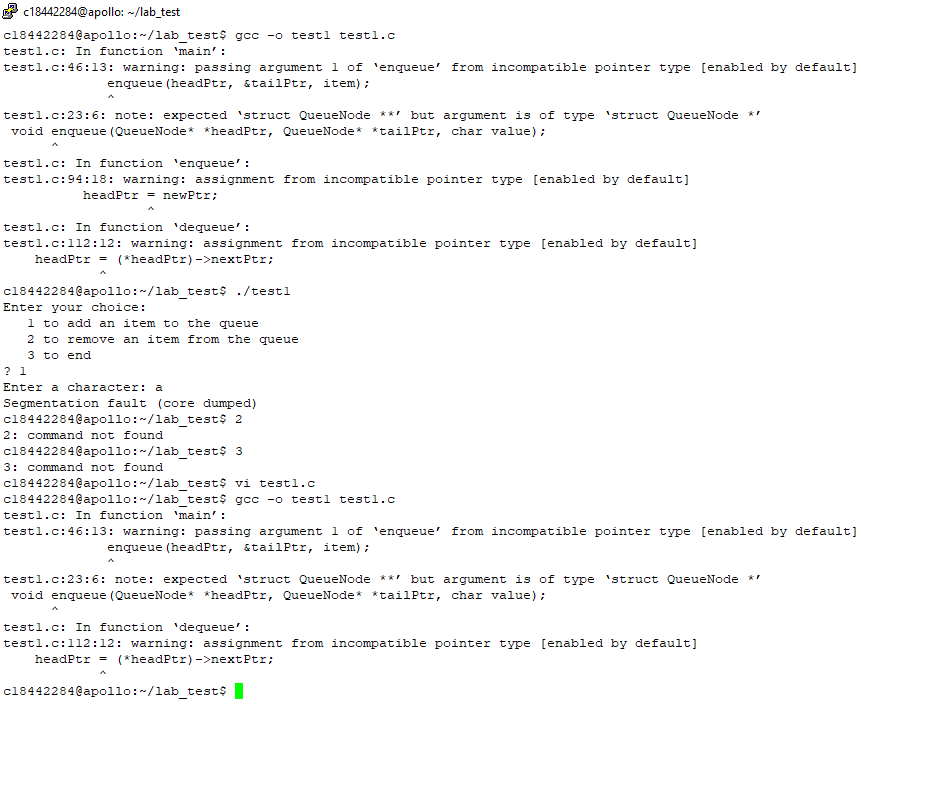
**Screen shots with the all warnings**





**Problem: headPtr = newPtr;**

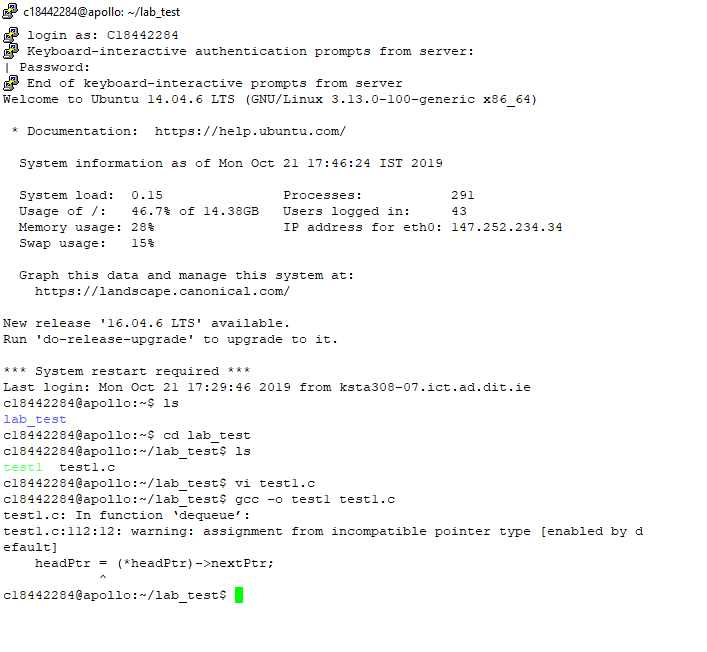
**First warning fixed: \* headPtr = newPtr**



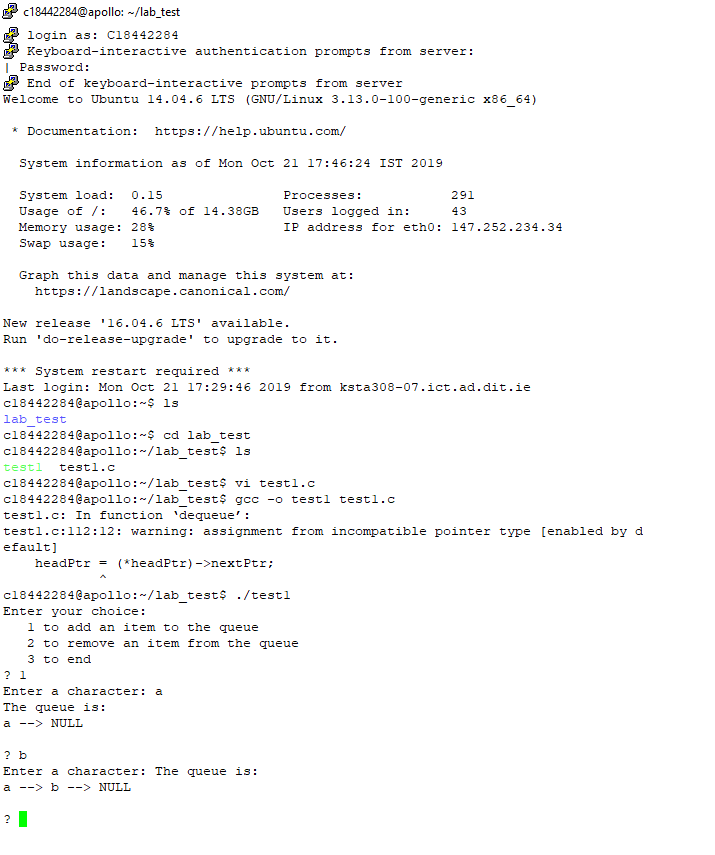
Program crashed after fixing the warning 1

**Problem: enqueue(headPtr, &tailPtr, item);**

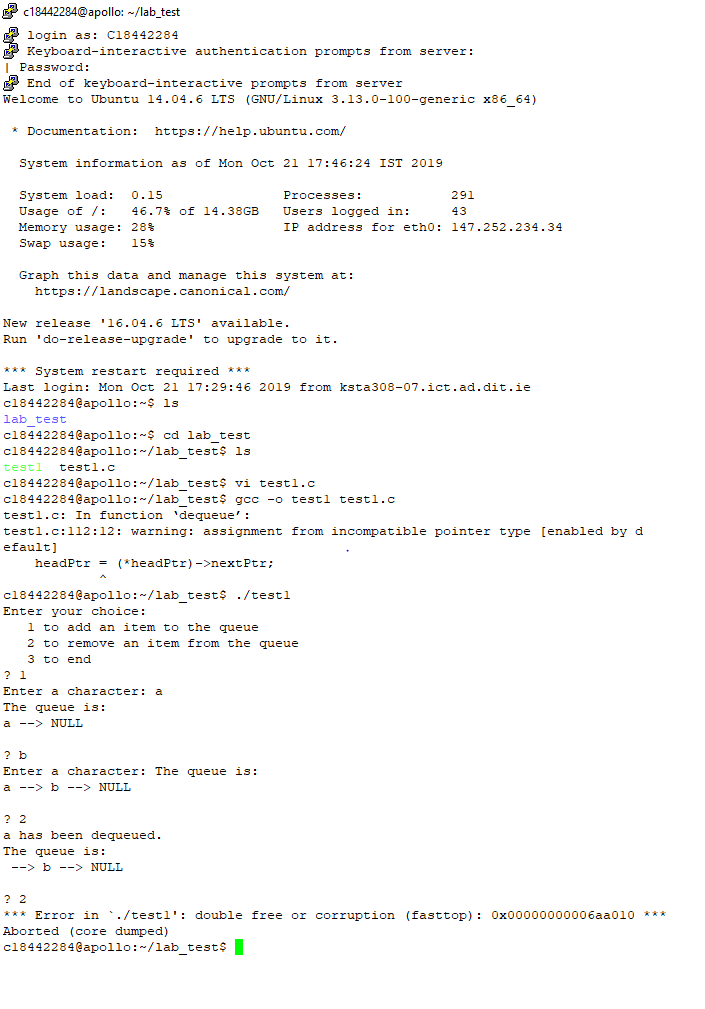
**Second warning fixed: enqueue(&headPtr, &tailPtr, item);**



Program start working(compiling) when I fixed the second warning and I can add to the queue

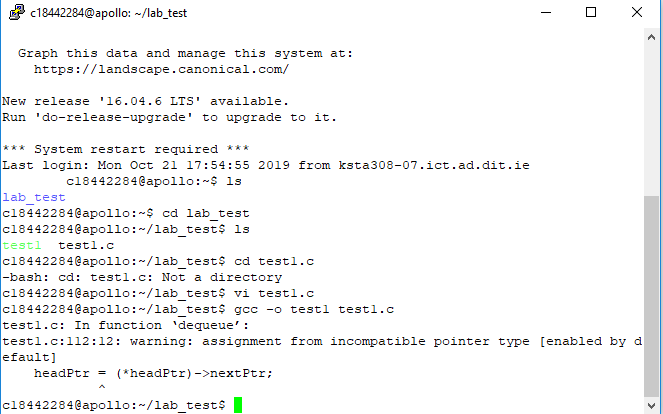


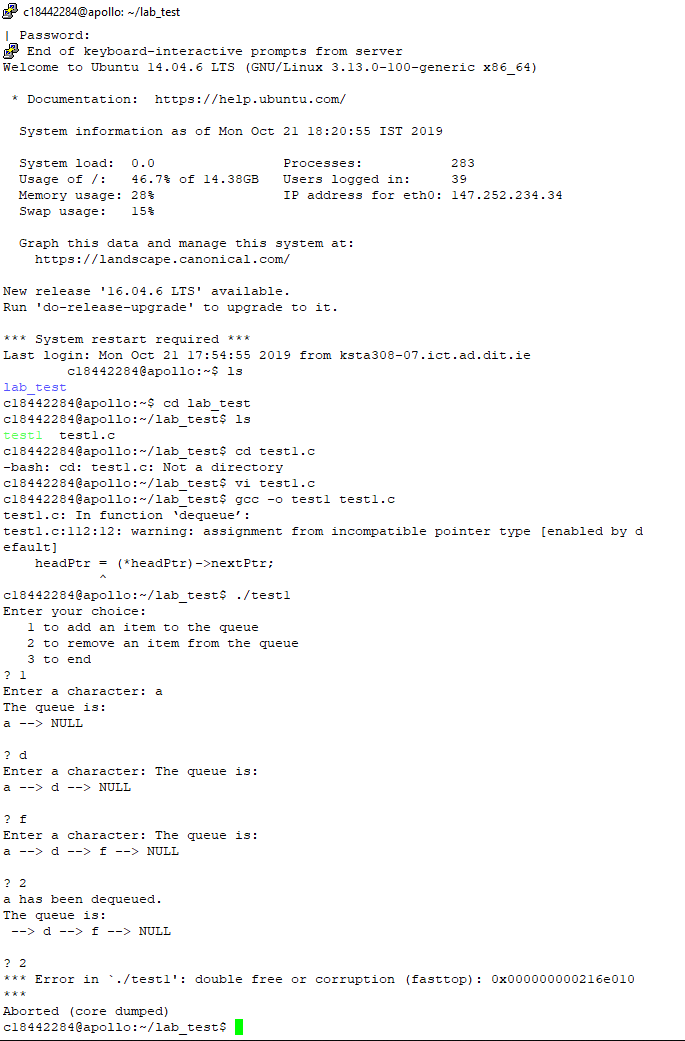
Program still crashes with for dequeue



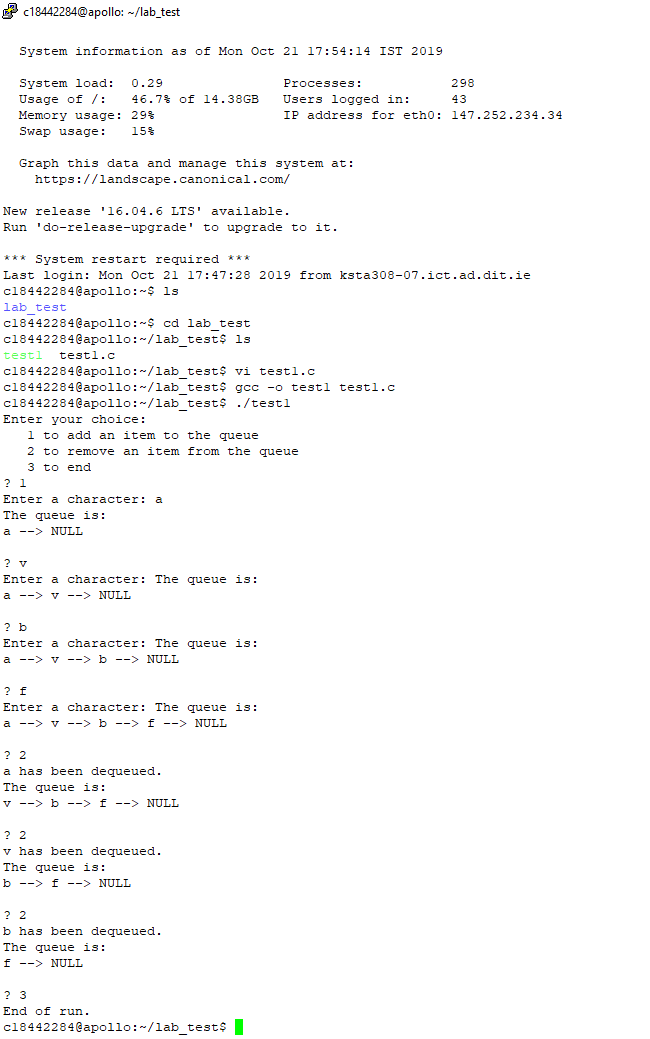
**Problem: headPtr = (\*headPtr) ->nextPtr;**

**Last waning fixed: \*headPtr = (\*headPtr) ->nextPtr in the dequeue function**





Program working perfectly now with no error and no warning



**Explaining:**

**1st warning:**

The first mistake headPtr is local to the function this headPtr is assigned to the address of the new node, but it should have been the heatPtr in the main which pointing to the new node.

Also we know the headPtr in the main contains null so that’s why the list is empty when we try to run it, to fix this problem I just changed headPtr to \*headPtr.

\*headPtr have to be assigned to point to the new node, by doing this now we can add elements to the queue.

Problem: headPtr = newPtr;

Fixed: \*headPtr = newPtr;

**2nd warning:**

The problem is we using the headPtr in the main which that means we not acutlly passing the address of the headPtr, so we have to pass the address of the headPtr which poiting to the null at the moment, by add the & before headPtr we can assign it to pointing to the first element and access to what in that address and then print it off.

Problem: enqueue(headPtr, &tailPtr, item);

Fixed: enqueue(&headPtr, &tailPtr, item);

**3rd warning:**

When we call the dequeue function it would assign the address of the second node to the headPtr, and then it will free TempPtr which means remove the first node, so when we call the dequeue function the program would crash, so by changing the headPtr to the \*headPtr this warning would be fixed and that means now the headPtr will point to the second node in the queue and we can remove the first one from the queue without crashing the program.

Problem: headPtr = (\*headPtr) ->nextPtr;

Fixed: \*headPtr = (\*headPtr) -> nextPtr;