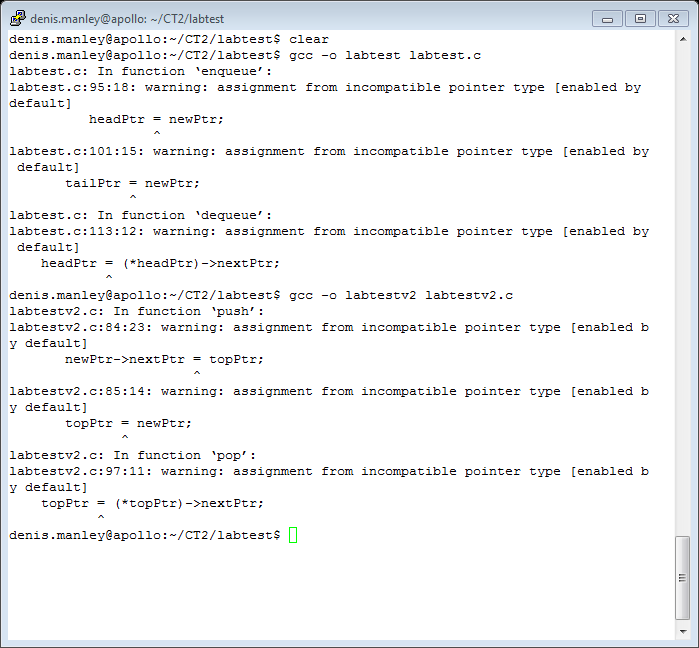
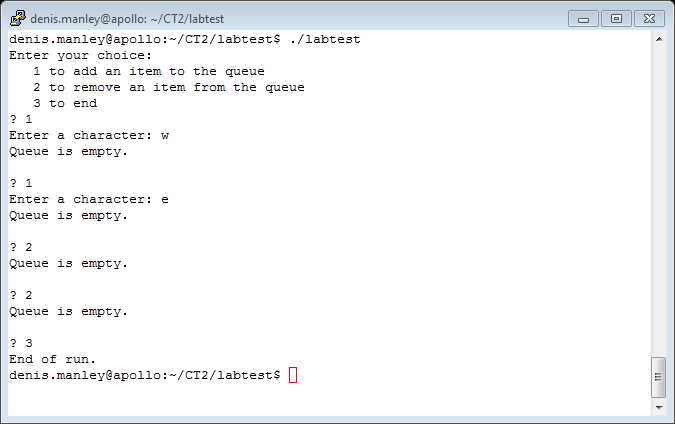
DT249/2 and DT255/2 Lab test 1 Solution

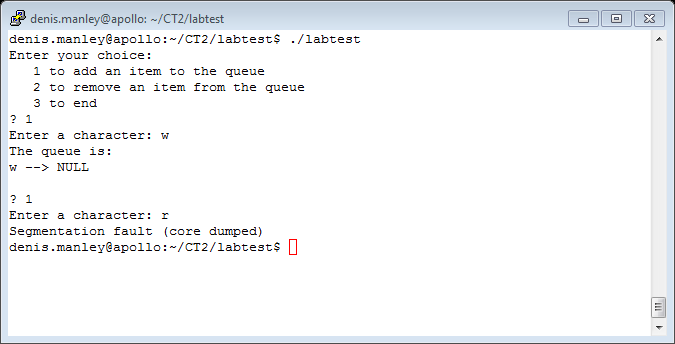
Screen shot with all the warnings





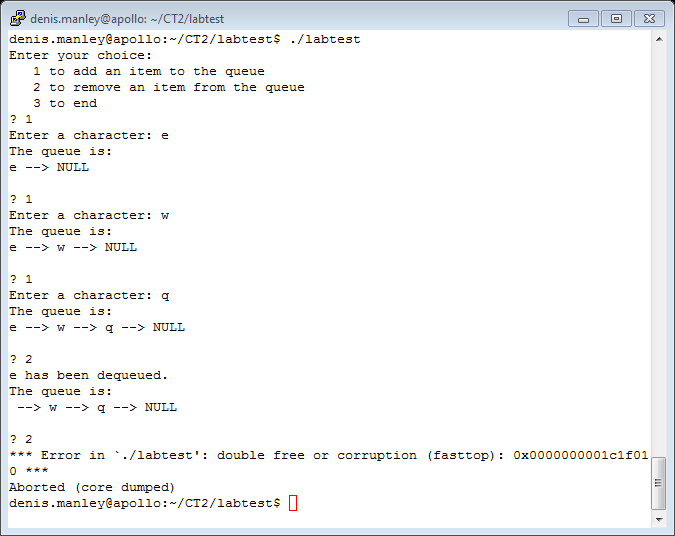
Screen shot with first warning fixed: line 95 \* headPtr = newPtr

Program has crashed:



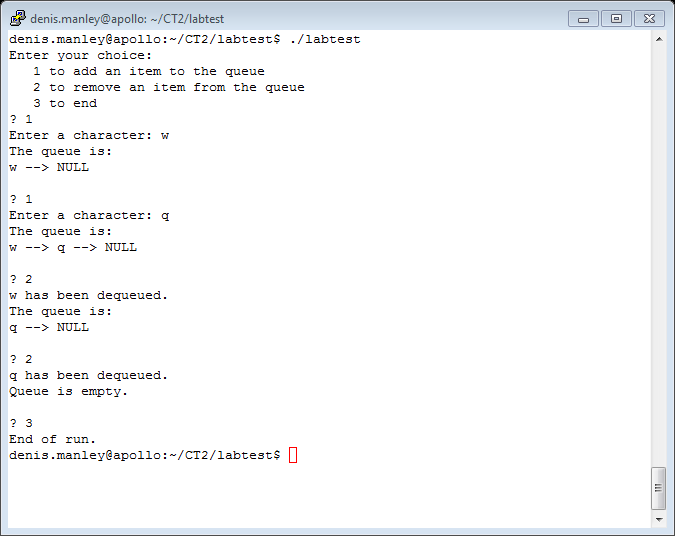
Output when the 2nd warning

Line 101 \*tailPtr = newPtr



Screen shot when all the warnings are fixed

113 \*headPtr = (\*headPtr) ->nextPtr in the *dequeue function*



What happended:

Warning 1:

headPtr = newPtr

here the headPtr (local to the function) is assigned the address of the new node.

However it should have been headPtr (in main) that was assigned to point to the new node

Therefore the headPtr (in main) still contains the value NULL and so the list will be displayed as empty (print function starts with the headPtr (in main) :

To fix the problem the headPtr (in main) - \*headPtr- must be assigned to point to a new node if it is to add an element to the queue.

So the problem is resolved by:

\*headPtr = newPtr;

(I will show illustration in the class)

Warning 2 after fixing warning 1:

tailPtr = newPtr (enqueue function)

So when we create and add the first node \*headPtr (headPtr in fumction main) points to it but the tailPtr (in main) or \*tailPtr should also point to it:

However **tailPtr = newPtr** only causes it to point to the local version of tailPtr (tailPtr in the main function still contains a Null value)

So when we try to a **second** node - call the enqueue function - - (\*tailPtr)->nextPtr = newPtr the program crashes as \*tailPtr (tailPtr in main) has a Null value and so *tailPtr -> nextPtr* will result in the program crashing,

(I will illustrate this later)

Finally the last warning

headPtr = (\*headPtr) ->nextPtr

the first time we call the dequeue function the R.H.S. of the expression will assign the address of the second node to headPtr (local to dequeue function).

Then free tempPtr removes the 1st node so the \*headPtr (head pointer in main is a stray pointer. So the next time we call the dequeue function it attempts to access the stray pointer and the program crashes giving the error show above in the screen shot.

When we fix the problem

\*headPtr = (\*headPtr) -> nextPtr

The correct variable is updated and so headPtr (in main) now points to the second node in the queue….

(I will illustrate this in class)