NUST
Defining futures

NCDC
NUST CHIP DESIGN CENTRE

Digital Design Verification

Lab # 07

Memory Management & Pointers

Submitted by:

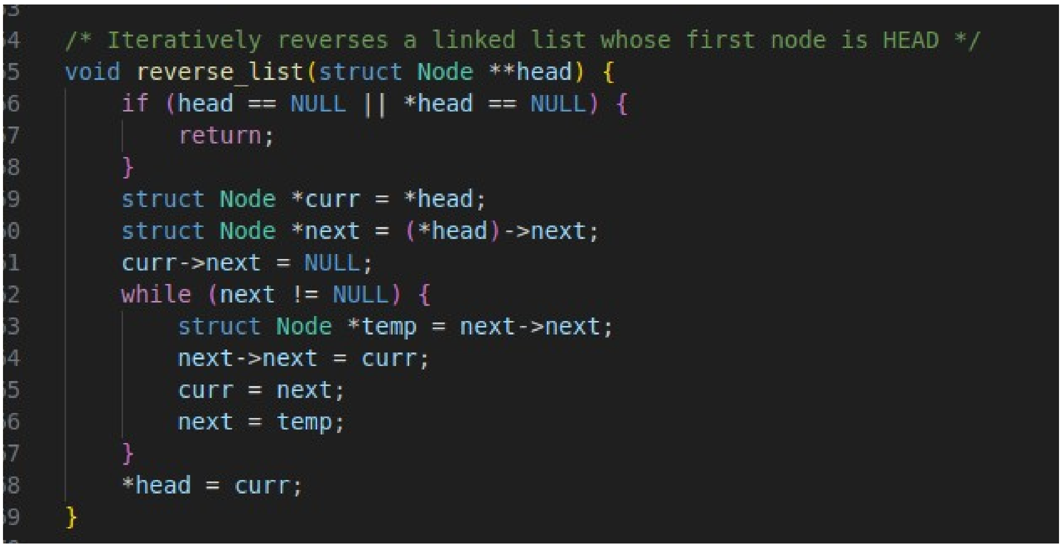
Name:	**Khalil Rehman**
Instructor:	**Hira Sohail**

Date:
July 24, 2025

NUST Chip Design Centre (NCDC), Islamabad, Pakistan

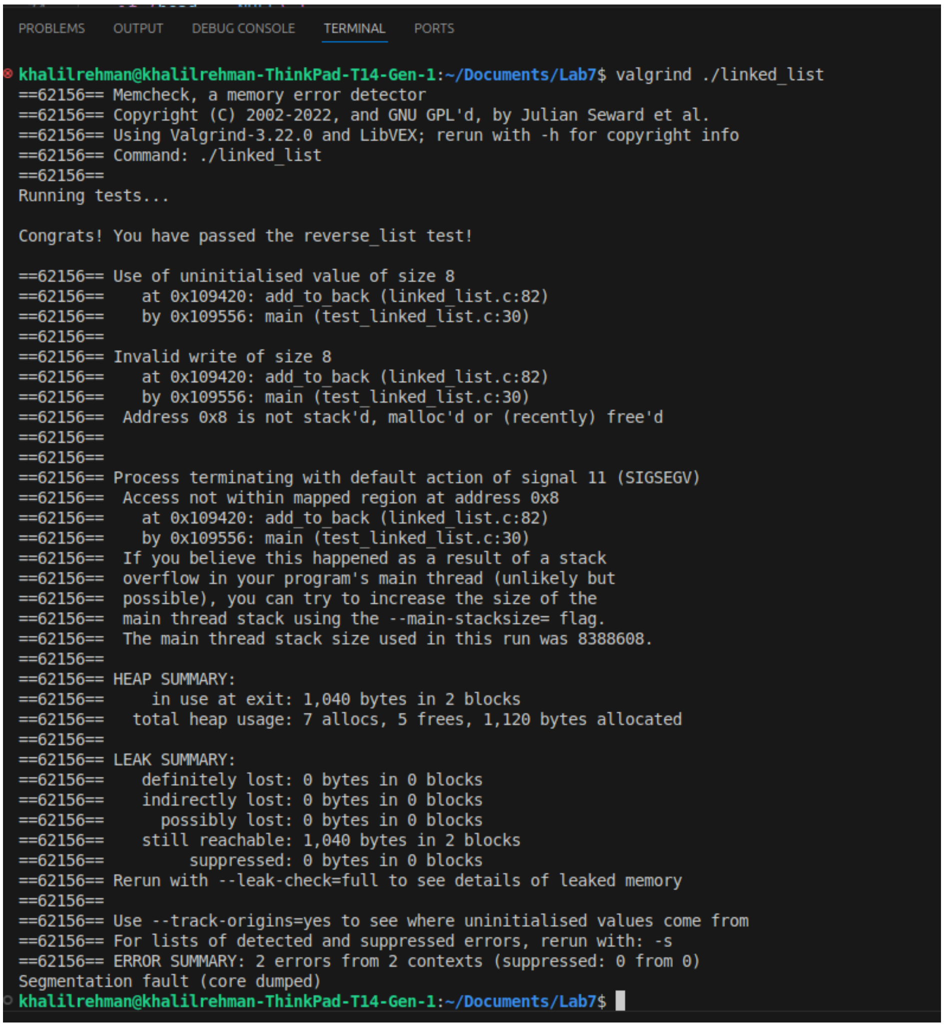
TASK # 01:

Code:



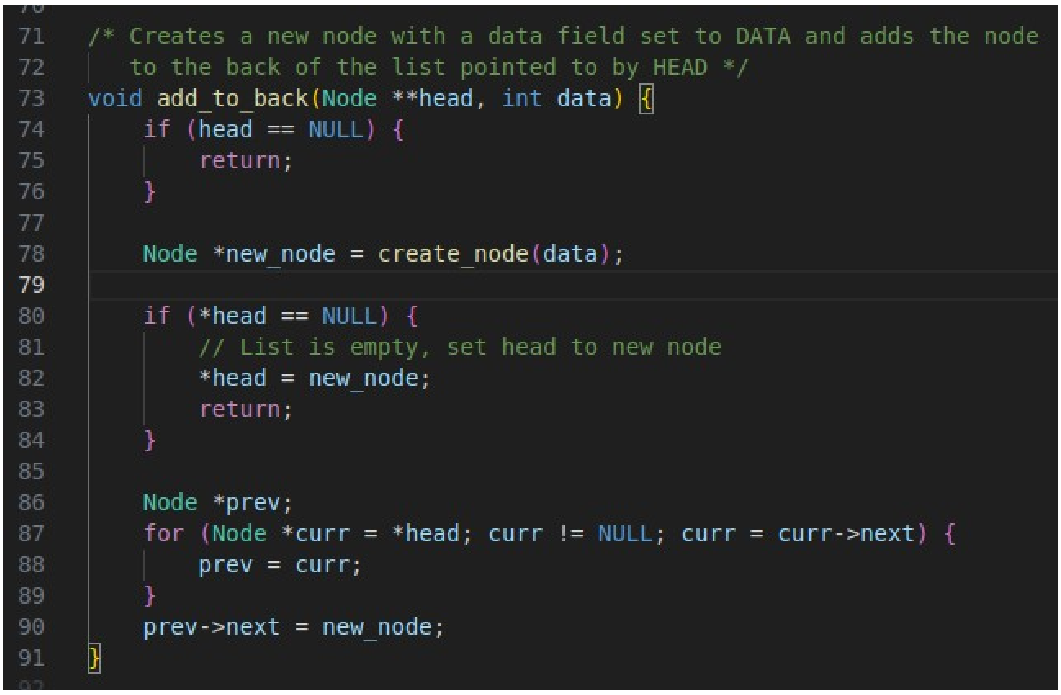
```c
/* Iteratively reverses a linked list whose first node is HEAD */
void reverse_list(struct Node **head) {
    if (head == NULL || *head == NULL) {
        return;
    }
    struct Node *curr = *head;
    struct Node *next = (*head)->next;
    curr->next = NULL;
    while (next != NULL) {
        struct Node *temp = next->next;
        next->next = curr;
        curr = next;
        next = temp;
    }
    *head = curr;
}
```

Terminal output:



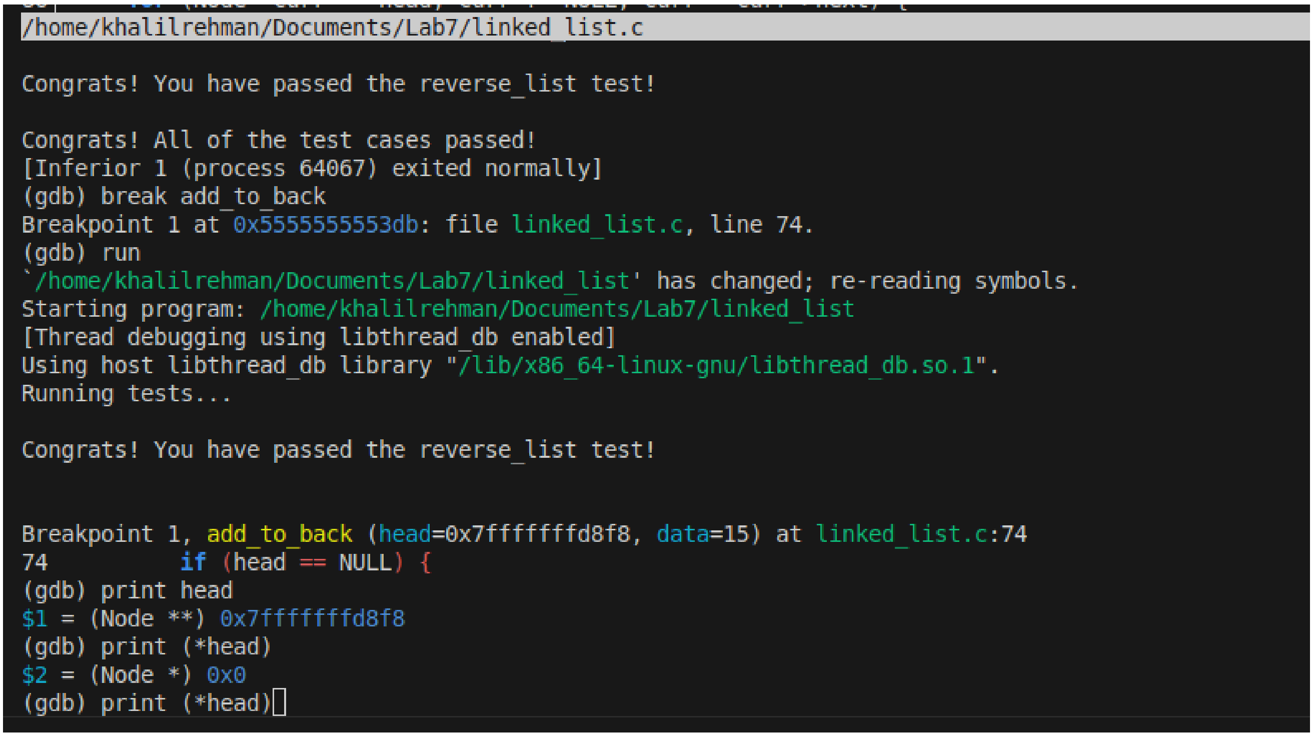
```
khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ valgrind ./linked_list
==62156== Memcheck, a memory error detector
==62156== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==62156== Using Valgrind-3.22.0 and LibVEX; rerun with -h for copyright info
==62156== Command: ./linked_list
==62156==
Running tests...

Congrats! You have passed the reverse_list test!

==62156== Use of uninitialised value of size 8
==62156==    at 0x109420: add_to_back (linked_list.c:82)
==62156==    by 0x109556: main (test_linked_list.c:30)
==62156==
==62156== Invalid write of size 8
==62156==    at 0x109420: add_to_back (linked_list.c:82)
==62156==    by 0x109556: main (test_linked_list.c:30)
==62156==  Address 0x8 is not stack'd, malloc'd or (recently) free'd
==62156==
==62156==
==62156== Process terminating with default action of signal 11 (SIGSEGV)
==62156==  Access not within mapped region at address 0x8
==62156==    at 0x109420: add_to_back (linked_list.c:82)
==62156==    by 0x109556: main (test_linked_list.c:30)
==62156==  If you believe this happened as a result of a stack
==62156==  overflow in your program's main thread (unlikely but
==62156==  possible), you can try to increase the size of the
==62156==  main thread stack using the --main-stacksize= flag.
==62156==  The main thread stack size used in this run was 8388608.
==62156==
==62156== HEAP SUMMARY:
==62156==     in use at exit: 1,040 bytes in 2 blocks
==62156==   total heap usage: 7 allocs, 5 frees, 1,120 bytes allocated
==62156==
==62156== LEAK SUMMARY:
==62156==    definitely lost: 0 bytes in 0 blocks
==62156==    indirectly lost: 0 bytes in 0 blocks
==62156==      possibly lost: 0 bytes in 0 blocks
==62156==    still reachable: 1,040 bytes in 2 blocks
==62156==         suppressed: 0 bytes in 0 blocks
==62156== Rerun with --leak-check=full to see details of leaked memory
==62156==
==62156== Use --track-origins=yes to see where uninitialised values come from
==62156== For lists of detected and suppressed errors, rerun with: -s
==62156== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)
Segmentation fault (core dumped)
khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ 
```

TASK # 02:

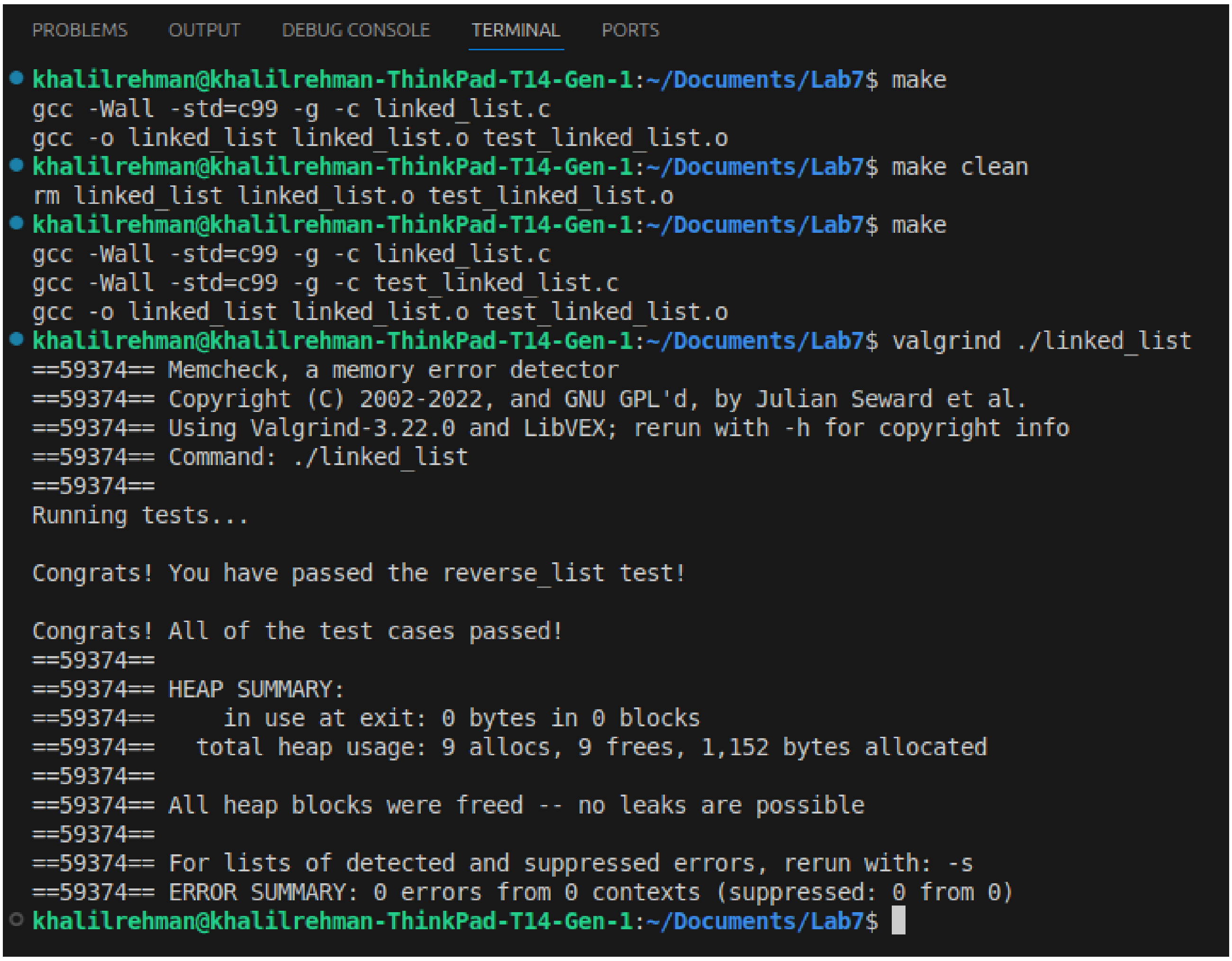


```c
/* Creates a new node with a data field set to DATA and adds the node
   to the back of the list pointed to by HEAD */
void add_to_back(Node **head, int data) {
    if (head == NULL) {
        return;
    }

    Node *new_node = create_node(data);

    if (*head == NULL) {
        // List is empty, set head to new node
        *head = new_node;
        return;
    }

    Node *prev;
    for (Node *curr = *head; curr != NULL; curr = curr->next) {
        prev = curr;
    }
    prev->next = new_node;
}
```

TERMINAL OUTPUT:



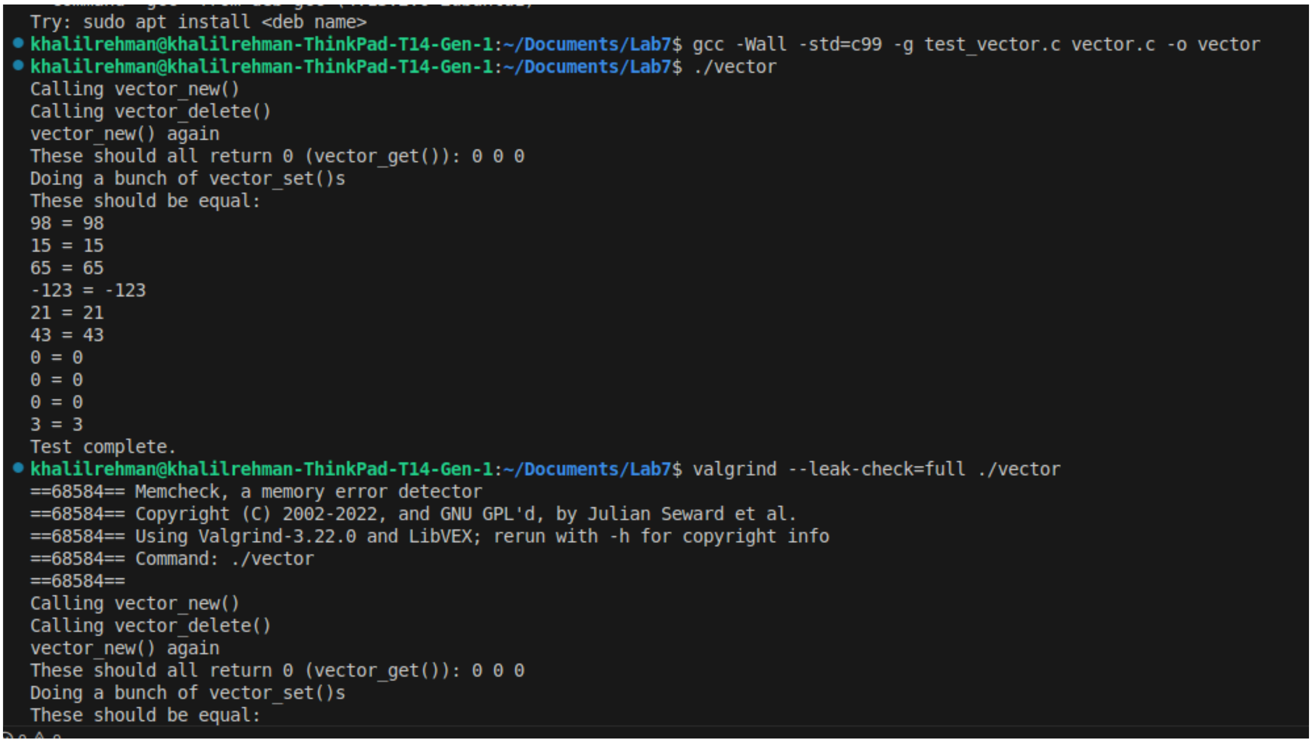
```
/home/khalilrehman/Documents/Lab7/linked_list.c

Congrats! You have passed the reverse_list test!

Congrats! All of the test cases passed!
[Inferior 1 (process 64067) exited normally]
(gdb) break add_to_back
Breakpoint 1 at 0x5555555553db: file linked_list.c, line 74.
(gdb) run
`/home/khalilrehman/Documents/Lab7/linked_list' has changed; re-reading symbols.
Starting program: /home/khalilrehman/Documents/Lab7/linked_list
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Running tests...

Congrats! You have passed the reverse_list test!


Breakpoint 1, add_to_back (head=0x7fffffffd8f8, data=15) at linked_list.c:74
74            if (head == NULL) {
(gdb) print head
$1 = (Node **) 0x7fffffffd8f8
(gdb) print (*head)
$2 = (Node *) 0x0
(gdb) print (*head)
```



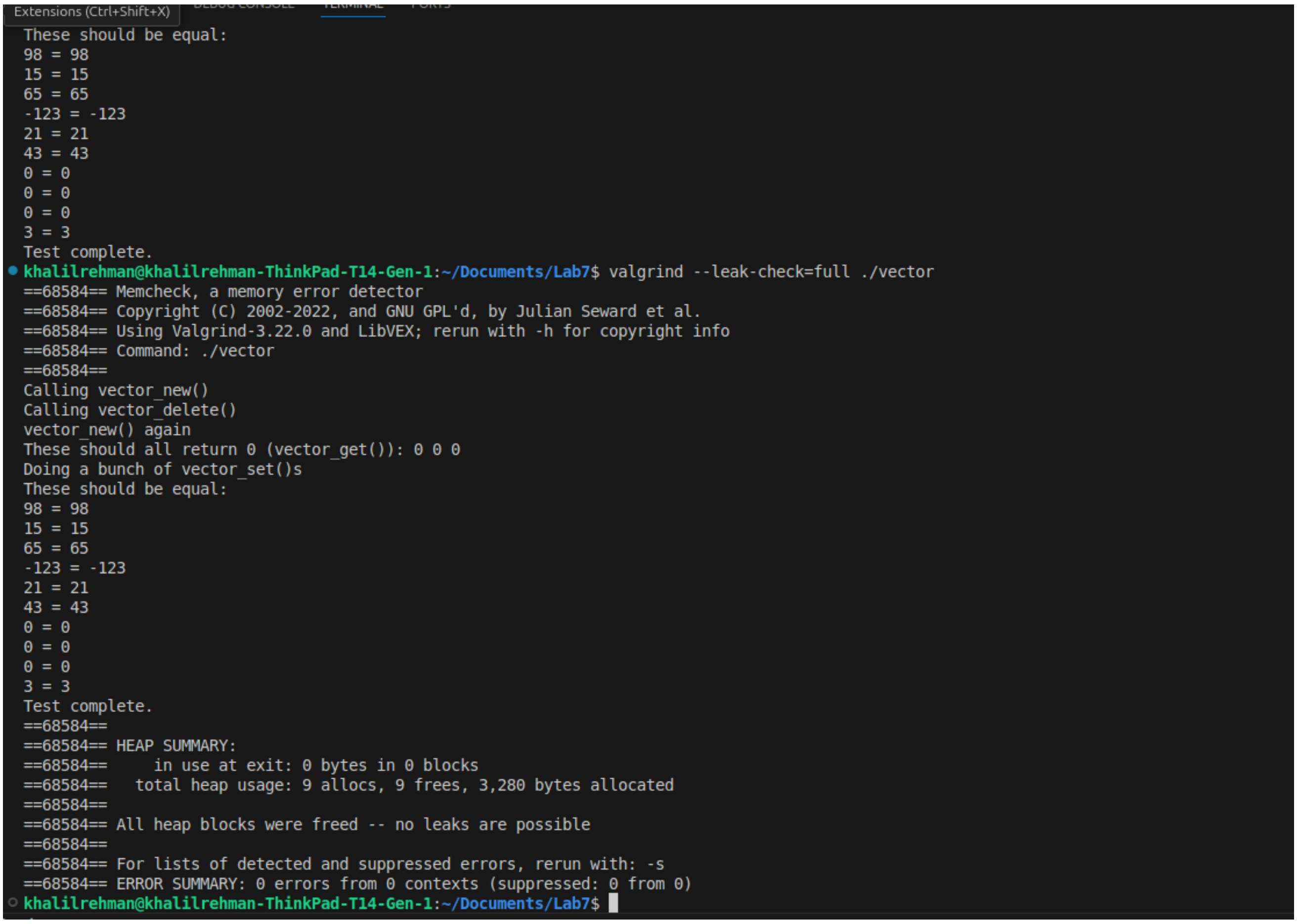
```
khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ make
gcc -Wall -std=c99 -g -c linked_list.c
gcc -o linked_list linked_list.o test_linked_list.o
khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ make clean
rm linked_list linked_list.o test_linked_list.o
khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ make
gcc -Wall -std=c99 -g -c linked_list.c
gcc -Wall -std=c99 -g -c test_linked_list.c
gcc -o linked_list linked_list.o test_linked_list.o
khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ valgrind ./linked_list
==59374== Memcheck, a memory error detector
==59374== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==59374== Using Valgrind-3.22.0 and LibVEX; rerun with -h for copyright info
==59374== Command: ./linked_list
==59374==
Running tests...

Congrats! You have passed the reverse_list test!

Congrats! All of the test cases passed!
==59374==
==59374== HEAP SUMMARY:
==59374==     in use at exit: 0 bytes in 0 blocks
==59374==   total heap usage: 9 allocs, 9 frees, 1,152 bytes allocated
==59374==
==59374== All heap blocks were freed -- no leaks are possible
==59374==
==59374== For lists of detected and suppressed errors, rerun with: -s
==59374== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ 
```

TASK # 03:

TERMINAL OUTPUT



```
  Try: sudo apt install <deb name>
● khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ gcc -Wall -std=c99 -g test_vector.c vector.c -o vector
● khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ ./vector
  Calling vector_new()
  Calling vector_delete()
  vector_new() again
  These should all return 0 (vector_get()): 0 0 0
  Doing a bunch of vector_set()s
  These should be equal:
  98 = 98
  15 = 15
  65 = 65
  -123 = -123
  21 = 21
  43 = 43
  0 = 0
  0 = 0
  0 = 0
  3 = 3
  Test complete.
● khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ valgrind --leak-check=full ./vector
  ==68584== Memcheck, a memory error detector
  ==68584== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
  ==68584== Using Valgrind-3.22.0 and LibVEX; rerun with -h for copyright info
  ==68584== Command: ./vector
  ==68584==
  Calling vector_new()
  Calling vector_delete()
  vector_new() again
  These should all return 0 (vector_get()): 0 0 0
  Doing a bunch of vector_set()s
  These should be equal:
```



```
  These should be equal:
  98 = 98
  15 = 15
  65 = 65
  -123 = -123
  21 = 21
  43 = 43
  0 = 0
  0 = 0
  0 = 0
  3 = 3
  Test complete.
● khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ valgrind --leak-check=full ./vector
  ==68584== Memcheck, a memory error detector
  ==68584== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
  ==68584== Using Valgrind-3.22.0 and LibVEX; rerun with -h for copyright info
  ==68584== Command: ./vector
  ==68584==
  Calling vector_new()
  Calling vector_delete()
  vector_new() again
  These should all return 0 (vector_get()): 0 0 0
  Doing a bunch of vector_set()s
  These should be equal:
  98 = 98
  15 = 15
  65 = 65
  -123 = -123
  21 = 21
  43 = 43
  0 = 0
  0 = 0
  0 = 0
  3 = 3
  Test complete.
  ==68584==
  ==68584== HEAP SUMMARY:
  ==68584==     in use at exit: 0 bytes in 0 blocks
  ==68584==   total heap usage: 9 allocs, 9 frees, 3,280 bytes allocated
  ==68584==
  ==68584== All heap blocks were freed -- no leaks are possible
  ==68584==
  ==68584== For lists of detected and suppressed errors, rerun with: -s
  ==68584== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
○ khalilrehman@khalilrehman-ThinkPad-T14-Gen-1:~/Documents/Lab7$ █
```