COP3530 – Data Structures and Algorithms I - Fall 2016

Project#3

*** Due 10/31/2016, Monday, 11:30 p.m.**

Instructions:

This is a project to be done and submitted individually. Please make sure your solutions are original, in your own words, expressing your own ideas. Any copying from others' home-works, text books or online sources will be penalized. I'll be checking the code for plagiarism, so, be advised.

*IMPORTANT: You can use any IDE for creating your programs. Create the programs and save them. Compile and run the programs. Save the screenshot/s of the successfully run programs and submit the screenshot/s with the corresponding source code files on the Dropbox. Do not submit any executable (.exe) files. Make sure that you zip everything required for the submission and **upload a single zip file**.*

Programs should be well-commented to make their understanding easy. Variable-names should be so chosen that it is easy to visualize what is going on, inside the program. Points would be taken off for poor documentation and readability.

It is highly recommended that students keep a soft copy of the assignment files with them. Any queries can be mailed to amishra@uwf.edu.

Any exceptions to this should be pre-approved by the Instructor; late submissions (only the ones pre-approved by the instructor before the deadline) are liable to be penalized.

Previous instructions about coding submissions stay as they are.

WHAT TO TURN IN:

Upload through the "Assignments" page on Dropbox, **a single zip file** containing all the files in your submission. Submit **each** of your C programs: one program for each of the above exercises. In addition, submit **all relevant screen captures of your successfully running programs** as jpeg files using the same name (for example, the screen captures for successfully running *sumOfCubes.c* should be named as *SumOfCubes1.jpg*, *SumOfCubes2.jpg* – if more than one relevant screen captures are included). Do not submit the object files or executables.

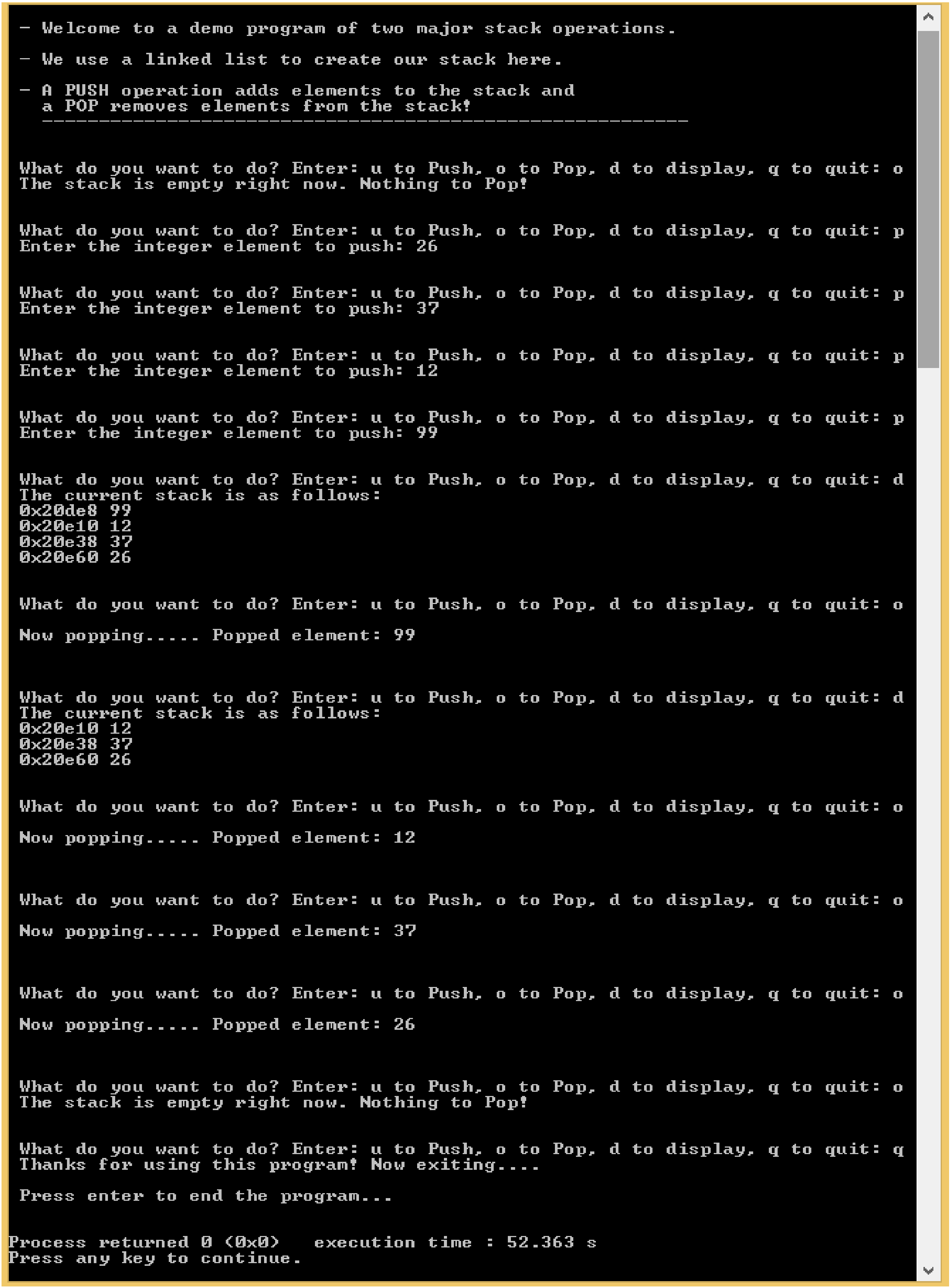
In order for your submission to be counted as on time, it should be submitted before the indicated deadline on the due date unless otherwise specified. The penalty for late submissions is 5% for each day late, and late submissions need to be preapproved by the instructor (they should NOT be automatically assumed as granted). The very latest that you may submit a late lab is one week after the submission was due.

The Problems

1. Write a complete C program with functions to create and display a stack of integers using a linked list.

a) Implement the functions push(int x), pop(), display(), and isEmpty(). **(10 points)**

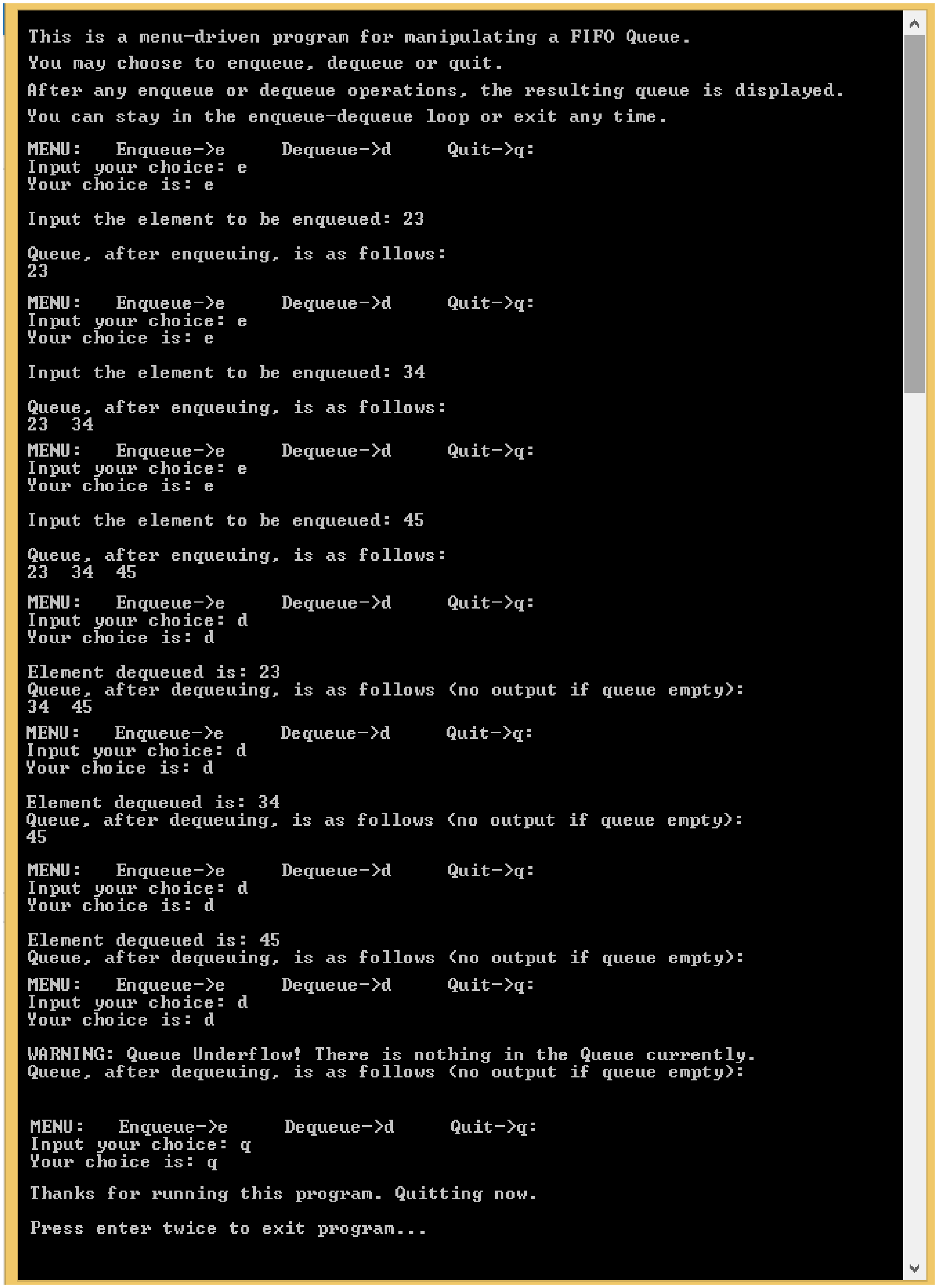
b) Write the test code in the main() function to test these functions. The test code should be menu driven, and should give users a choice to keep the program running in a loop, calling these functions or quitting the program. A representative sample output is given below. **(2 points**)


```
- Welcome to a demo program of two major stack operations.

- We use a linked list to create our stack here.

- A PUSH operation adds elements to the stack and
  a POP removes elements from the stack!
  ---------------------------------------------------------


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: o
The stack is empty right now. Nothing to Pop!


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: p
Enter the integer element to push: 26


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: p
Enter the integer element to push: 37


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: p
Enter the integer element to push: 12


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: p
Enter the integer element to push: 99


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: d
The current stack is as follows:
0x20de8 99
0x20e10 12
0x20e38 37
0x20e60 26


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: o

Now popping..... Popped element: 99


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: d
The current stack is as follows:
0x20e10 12
0x20e38 37
0x20e60 26


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: o

Now popping..... Popped element: 12


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: o

Now popping..... Popped element: 37


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: o

Now popping..... Popped element: 26


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: o
The stack is empty right now. Nothing to Pop!


What do you want to do? Enter: u to Push, o to Pop, d to display, q to quit: q
Thanks for using this program! Now exiting....

Press enter to end the program...


Process returned 0 (0x0)   execution time : 52.363 s
Press any key to continue.
```

2. Write a complete C program with functions to create and display a queue of integers using a linked list. You do not have to implement a circular queue.

a) Implement the functions enqueue(int x), dequeue(), display(), and isEmpty(). **(10 points)**

b) Write the test code in the main() function to test these functions. The test code should be menu driven, and give users a choice to keep the program running in a loop, calling these functions or quitting the program. A representative sample output is given below. **(3 points)**



```
This is a menu-driven program for manipulating a FIFO Queue.
You may choose to enqueue, dequeue or quit.
After any enqueue or dequeue operations, the resulting queue is displayed.
You can stay in the enqueue-dequeue loop or exit any time.

MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: e
Your choice is: e

Input the element to be enqueued: 23

Queue, after enqueuing, is as follows:
23

MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: e
Your choice is: e

Input the element to be enqueued: 34

Queue, after enqueuing, is as follows:
23  34
MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: e
Your choice is: e

Input the element to be enqueued: 45

Queue, after enqueuing, is as follows:
23  34  45

MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: d
Your choice is: d

Element dequeued is: 23
Queue, after dequeuing, is as follows (no output if queue empty):
34  45
MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: d
Your choice is: d

Element dequeued is: 34
Queue, after dequeuing, is as follows (no output if queue empty):
45

MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: d
Your choice is: d

Element dequeued is: 45
Queue, after dequeuing, is as follows (no output if queue empty):
MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: d
Your choice is: d

WARNING: Queue Underflow! There is nothing in the Queue currently.
Queue, after dequeuing, is as follows (no output if queue empty):


MENU:   Enqueue->e     Dequeue->d     Quit->q:
Input your choice: q
Your choice is: q

Thanks for running this program. Quitting now.

Press enter twice to exit program...
```

***