Assignment 4 Guidelines

You are allowed to use standard library functions. The assignment should be submitted as a tar/gz file containing the source code for each question as well as a readme file. There should also be a Makefile to compile the program.

Create your tar/gz file using the following command:
'$/> tar -zcvf myassignment.tar.gz Makefile readme.txt treecalc.c'

The tar/gz file should contain:
readme.txt
avltree.c
Makefile
The compiled program **MUST** be named '**avltree**'. Failure to name your binary program 'avltree' will result in a mark deduction.

For the Makefile, all programs should be compiled with -Wall and -ansi flags. The first line of the Makefile should be an all rule, as the programs will be compiled using make with no arguments.

Any warnings will result in a mark deduction appropriate to the severity of the warnings. There will be some marks allocated for style and documentation, but the majority will come from the correctness of the programs.

List all the limitations of your program and everything you want to share with TA who marks the assignment in readme.txt. Remember to include your name, and student ID in all files.

Requirements:

(1) Make your program display a welcome banner when started. This welcome banner MUST contain your full name and student ID. After that, your program MUST display a prompt when it is ready for new input. The prompt must the following string: "avl/> " Make sure to include a blank space right after the prompt.
(2) Option initialization (#1) should prompt for the filename on a separate line. It should display the following string 'filename: ' (without quote) and expect a valid file name to be entered by the user.
For example:

1. Initialization
2. Find
3. Insert
4. Remove
5. Check Height and Size
6. Find All (above a given frequency)
7. Exit
avl/> **1**
filename: **a4_data_file.txt**

(3) The second option is has a similar behavior as option one. When the user select this option, the program should prompt the user for the key using the prompt 'find key: '. After the key has been found, the program should display the result in such a way: 'key: flr795 frequency: 150'
For example:

1. Initialization
2. Find
3. Insert
4. Remove
5. Check Height and Size
6. Find All (above a given frequency)
7. Exit
avl/> **2**
find key: **flr795**
key: flr795 frequency: 150

or

1. Initialization
2. Find
3. Insert
4. Remove
5. Check Height and Size
6. Find All (above a given frequency)
7. Exit
avl/> **2**
find key: **unknown**
no_such_key

(4) Option 3 prompts for a key string to be inserted in your AVL tree. The program should display a 'insert key: ' prompt requesting the key to be entered by the user.
For example:

1. Initialization
2. Find
3. Insert
4. Remove
5. Check Height and Size
6. Find All (above a given frequency)
7. Exit
avl/> **3**
insert key: **flr795**

(5) Option 4 prompts for the key to be removed from your AVL tree. The program should display the 'remove key: ' prompt and expect the user to enter the key. If the key is not found, 'no_such_key' should be printed before returning to the prompt.
For example:

1. Initialization
2. Find
3. Insert
4. Remove
5. Check Height and Size
6. Find All (above a given frequency)
7. Exit
avl/> **4**
remove key: **flr795**

or

1. Initialization
2. Find
3. Insert
4. Remove
5. Check Height and Size
6. Find All (above a given frequency)
7. Exit
avl/> **4**
remove key: **unknown**
no_such_key

(6) Option 5 prints the height and size of your tree on a separate line. The format should be: 'height: 14 size: 22' where 14 is the actual tree height and 22 the real size of the tree.
(7) Option 6 display all key with a frequency above a given number. Your program should prompt for the frequency and output the keys in the same format as option 2.
For example:

1. Initialization
2. Find
3. Insert
4. Remove
5. Check Height and Size
6. Find All (above a given frequency)
7. Exit
avl/> **6**
frequency: **20**
key: flr795 frequency: 150
key: flr794 frequency: 254
key: flt123 frequency: 445
key: flt156 frequency: 265
...

You can traverse your tree in any order you wish (breadth-first or depth-first).

(8) Option 7: Terminates your program.