Project 3: AVL Tree

Due: Tue, Apr 3

In this assignment, you will implement an `AVL` class. This class will allow the insertion and removal of integers into an AVL tree.

Getting started

Clone the project stub into the `Projects` subdirectory of your SVN working directory.

```
cd ~/ID/Projects
svn export https://dev.cs.uakron.edu/svn/cs316sp17/shared/Projects/AVL/
```

Be sure to `svn add` the newly exported `AVL` directory and then commit.

Investigate the contents of that directory. You should have the following files:

- `CMakeLists.txt` — The build system for your homework. Read this file carefully.
- `main.cpp` — A small program that will utilize your AVL class. DO NOT MODIFY THIS FILE!
- `output.txt` — The expected output.

You can try building the project, but it won't work. The main.cpp file is written in terms of functions that *you* must provide.

Class requirements

The `AVL` class must have the following:

1. A default constructor.
2. A destructor — memory leaks will cost you points
3. An `insert` function that takes an integer and inserts it into the AVL tree. There will be no duplicates in the tree, if an integer is already present do not add another. This function return type is void.
4. A `clear` function that removes all nodes from the tree. This function return type is void.
5. A `serialize` function that returns a string representing the tree in a serialized manner.

Define the `AVL` class and define all associated functions in `AVL.hpp`.

Testing

Create a build directory in the root directory of the project. DO NOT ADD THE BUILD DIRECTORY TO SVN! From this directory you should run the following code:

```
cmake ..
```

From this point you can type `make` inside the build directory to compile your code and `./avl` to test the output of your code (once it compiles). The output of your program should match the `output.txt` file in the program directory. An easy way to check this is to run the following commands from the build directory.

```
make
./avl > ../my_output.txt
diff ../output.txt ../my_output.txt
```

Submission

1. Committing it to your SVN repository.

Grading basis

If your homework is not in subversion you will get a 0 on your assignment.

The total is out of 100 points.

- 100 You submitted code that compiles, and gives the correct output.
- You lose 10 points for having memory leaks. You can use valgrind to check for memory leaks on the knuth2 server.
- 200 You submitted code which received a 100% without keeping track of a nodes parent within the node (dont use a parent pointer).

NOTE: In order to receive extra credit you will have to research using pointer to a pointer (AVLNode**) and implement the entire AVL tree using then in your recursive calls. I will not answer any questions regarding how to do the extra credit in terms of how to do it.