Program #1 post coding written assessment

Coding write-up

My design, for this assignment I chose to have two controller type classes (hand and board). These two classes were responsible for either being what they need or having what they need. The hand also ‘uses’ the board quite heavily.

The classes used are: Board, Hand, Column Space, Row Space, Tile (this is a piece), and Tile Bag. The basic idea is that the Board will create the board data structure (doubly linked list of doubly linked lists). It will also have a Tile Bag object that is responsible for generating the new tiles for the game. Once the tiles are created, the Hand object can take tiles from it, and have/contain them while the user decides what to do. The tiles memory is not being changed or copied, instead pointers to them are being transferred. When the bag’s tile is given to the hand, the bag loses the pointer to it and the hand gains a pointer to it. I think this will end up making the program more efficient since only the memory address is being shared.

Eventually the user will want to place a tile on the board which is stored in the data structure. The data structure is comprised of the column spaces and the row spaces. The column spaces’ job is to contain the head pointers for the row spaces doubly linked list. The column spaces aren’t actually part of the game board and don’t hold tiles or display them. Instead they call the display & other functions of the row spaces which ARE actually the game board as the end user sees it. They hold tiles along with pointers to the list and can display themselves.

For program two I would like to create two base classes from these classes in addition to new potential ones. I think both the column and row spaces could have a base class that knows how to traverse forward and backwards, and contains pointers. It could also have a purely virtual display function. Secondly when I introduce a user player and an AI player in program two, they will both be descendants of the Hand class. The players are effectively their hands, with just different functions for taking actions. This will be useful for not needing too many functions and I’ll be able to use the playing a tile and showing hand functions over again.

It’s also possible I can have both the board and dictionary descend from the same collection base class since they are both the classes that contain the data structures.

Debugger write up:

The debuggers I used greatly assisted me in my development.

I used both GDB and Valgrind. With GDB it allowed me to step through my code and figure out exactly where my problems lie. In particular I used it a lot when creating my TileBag class. That class had a number of out of bounds exceptions and segmentation faults that would have been very hard to find. One tricky thing, what I had an out of bounds exception that didn’t occur when I was using GDB but did occur without it. I figured out that I was looping through the allocation one more time that I should have eventually, and Valgrind is what helped me discover that. It was telling me that I had memory that was accessed as an uninitialized value. This wasn’t causing a crash, or an exception, but did lead to incorrect logical behavior and Valgrind was invaluable in discovering that. Also Valgrind helped me to remember that I needed to delete the pointers I created in main (tricky since there’s no main destructor to remind me). I really want to be able to list all stack variables in gdb, and there’s probably a way to do it, so I’ll look that up for program #2. I suspect I will use both valgrind and gdb quite a bit in program #2.

One last word, I use a mac, and both gdb and valgrind are no longer supported by default on mac. I had to use homebrew to install them.