COMP 2355
Winter 2012
Homework #2
Due at midnight, February 14, 2012

For this assignment you need to modify the code from Homework #1. You can use either board for this homework. An official solution to Homework #1 will be posted on the course web page after the deadline for Homework #1. The following changes should be made:

1. The state array in homework #1 used 8 bits per state to store the depth. Modify the state array to use only four bits per state. As there is no native 4-bit type, you will need to use a larger type and bit manipulations to get and set the bits for each state. These four bits will be used for two purposes.
   a. Two of the four bits will represent 4 values: *Already seen*, *current depth*, *next depth*, and *unseen*. This will efficiently mark what states should be expanded in the current iteration and future iterations.
   b. The other bits will store the depth of each state modulo 4. This is sufficient information to extract the path from the board.
2. Write code to save and load the state array so that the results can be re-used.
3. Write an output operator to print the board.
4. Write one function for finding potential "reverse" moves and another function to "undo" a move. Use this to find and print the the solution after the BFS is complete.
5. Modify the functions in Homework #1 to pass states by reference, where appropriate. Additionally label variables as `const` where appropriate.

Submit your final assignment, including a text file containing the best move sequence, to your SVN repository in a folder entitled "HW #2".

Tips for using 4 bits per state:

- Create a simple API for allocating the memory, freeing the memory, reading a 2-bit value, writing a 2-bit value, saving to disk and loading from disk:

```
uint64_t *GetBitArray(uint64_t numEntries, int bitsPerEntry);
void FreeBitArray(uint64_t *array);
int ReadTwoBitValue(uint64_t *array, uint64_t index);
void WriteTwoBitValue(uint64_t *array, uint64_t index, int value);
void SaveArray(uint64_t *array, uint64_t numEntries, int bitsPerEntry, const char *filename);
void LoadArray(uint64_t *array, uint64_t numEntries, int bitsPerEntry, const char *filename);
```

- Store two separate arrays for the depth of each state and for the search status of each state.
- After you have handled all the states, you will need to scan through the array again to convert states which are marked *next depth* to *current depth*. You can convert states which are marked *current depth* to *already seen* during this scan, or when all the legal moves are computed.