# Gebze Technical University
# Department of Computer Engineering
# CSE 241/505
# Object Oriented Programming
# Fall 2019
# Homework # 2
# N-Puzzle Continued
# Due date Oct 22nd 2019

You will extend the C++ program that you wrote for the N-Puzzle problem. The new extensions are as follows

- Your program will accept one optional file name as a command line option. The file name is a text file that contains the shape and the initial configuration of the valid N-Puzzle board. This shape contains the possible positions of the tiles with tile numbers and impossible positions with '00's. The blank tile is marked with 'bb'. The following is a sample file. Note that your board does not have to be square anymore. There are 40 possible positions for the following board, which means tiles are numbered from 01 to 39. The solution always places the empty tile at the lower right corner.

```
00 00 00 00 01 02 04 05 00
00 00 00 06 08 09 10 12 13
00 00 00 17 19 14 15 18 07
00 00 00 20 21 23 25 24 29
22 26 28 35 34 31 33 38 39
bb 30 31 34 32 36 03 11 16
```

- Your board game will be more intelligent. Sometimes behaving randomly is rational and this strategy can beat other more "intelligent" behavior. As you have observed from your HW1, making intelligent moves all the time often gets you stuck. Random moves will get you out! You will mix random board moves with the "intelligent" moves that you implemented in HW1. For example, you may make a few random moves and then a few intelligent moves and continue until the solution is found. You may try different strategies such as more random moves at the beginning of the game and then less random moves.
- You will add the following Input/Action pairs to your table

| Input | Action |
|---|---|
| V | Solves the problem from the current configuration using the new intelligent algorithm. |
| T | Prints a report about how many moves have been done and if the solution is found |
| E | Asks a file name and saves the current board configuration as a loadable shape file. |
| L | Asks a file name and loads the current board configuration from the shape file. |

Notes:

- Do not use any functions from the standard C library (like **`printf`**), use `cout`, `cin`, etc.
- Your program should have only functions and no classes other than enums.
- Do not use any C++ features that we did not learn during the lectures.
- You will use all the object oriented techniques that we learned in the class including **`const, inline, declrtype, auto`** keywords.
- Do not use anything that we did not learn in the lectures.
- Do not forget to indent your code and provide comments.
- Check the validity of the user input.
- **Test your programs very carefully at least with 10 different runs and submit your result files for each.**
- You should submit your work to the moodle page and follow all the submission rules that will be posted.