Assignment 2

16 September 2012

Due Wednesday 19 September at 1am

Implement heuristic for 8-puzzle

Open any file within your `cs102` folder in gedit, and then choose the "Sync with git" option from the Tools » External Tools menu.

Now you should have an `a2` folder. Open `8puz.cpp`. This is an implementation of the 8-puzzle using the A* algorithm. Use Alt-F5 to run it. The output will end something like this:

```
GOAL!
1. up: 235:180:764
2. up: 235:184:760
3. right: 235:184:706
4. down: 235:104:786
5. left: 235:140:786
6. down: 230:145:786
7. right: 203:145:786
8. right: 023:145:786
9. up: 123:045:786
10. left: 123:405:786
11. left: 123:450:786
12. up: 123:456:780
Visited 1158 nodes.
```

It scrolled way off the screen, but the start state was `230:185:764`. The zero represents the empty space on the 3×3 grid. The names of the moves are based on which direction a tile moves *into* the empty space. For example, in step 1, "up" refers to moving the 5 from the middle-right slot up into the empty space at the upper-right. The result is `235:180:764` (so the zero moved *down*).

You can find how the initial board was created in the `puz_state::puz_state` constructor, around line 40. After setting up the tiles in their *solved* order, this constructor calls a series of moves that shuffle the tiles:

```
    move(down);
    move(right);
    move(right);
    move(down);
```

```
    move(left);
    move(left);
    move(up);
    move(right);
    move(up);
    move(left);
    move(down);
    move(down);
```

You can verify that these moves are precisely the opposite of the solution the program found, and in reverse order.

Now look at the `heuristic` method, near line 24. You can see that it currently returns zero for every state:

```
int puz_state::heuristic() const {
    int score = 0;
    // TO DO
    return score;
}
```

Since all states have the same heuristic, our A* algorithm isn't really doing any better than BFS. Your task is to implement a better heuristic, and see whether we can find the path by visiting fewer than 1158 nodes.

As described in the book, the heuristic should count the number of tiles that are *out of place*. The tiles are kept in an array (actually a C++ vector, but it works the same way) called `tiles`. This array holds 9 integers (in positions 0 through 8). The value 0 represents the empty space, and 1–8 represent the tiles.

To test your heuristic independently of the A* algorithm, you can temporarily comment out the body of `main` (around line 140) and replace it with something like this:

```
    puz_state ps;
    ps.move(puz_state::down);
    ps.move(puz_state::right);
    cout << ps.heuristic() << endl;
```

This sets up a puzzle state, makes some moves, and then prints out the heuristic score for the resulting board. In this particular case, your heuristic method should produce 2.

When you are satisfied with your heuristic (or really, any time you want to snapshot your changes to the server), just use "Sync with git."

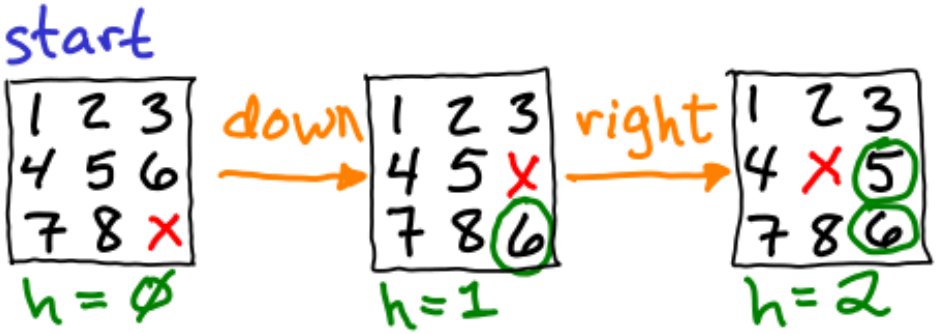

Figure 1: Moves and heuristics