Course C^{++}, Exercise Number 10

Deadline: 12.05.2015

The Fifteen Puzzle

The fifteen puzzle `http://en.wikipedia.org/wiki/15_puzzle`) was invented by Noyes Palmer Chapman in 1875. In the beginning of 1880, the puzzle became a craze, that lasted approximately half a year. (In 1981, the same effect was obtained by Rubik's cube.)

We will solve the 15-puzzle by the search algorithm that is described in the slides.

We implement the function F by an `unordered_map`, and the set U by a `priority_queue`.

1. Download the files in directory `fifteen` from the course homepage. Write the two constructors of `fifteen`, and `operator <<` . This last operator must be made **friend**.

   You can use `std::setw( )`, defined in `iomanip` to format the output.

2. Complete the missing members of class `fifteen`. These are **solvedposition( )**, **hashvalue( )**, **equals( )**, **makemove( )**, **issolved( )**, **distance( )** .

3. We want to implement F using an `unordered_map`. In order to do this, we need a hash object and a compare object. Just write in file **solve.cpp**:

   ```
   size_t hash( const fifteen& f )
   {
      return f. hashvalue( );
   }

   bool equals( const fifteen& f1, const fifteen& f2 )
   {
      return f1. equals( f2 );
   }
   ```

4. Next we can turn our attention to the priority queue U. Define

```
bool further( const fifteen& f1, const fifteen& f2 )
{
   return f1. distance( ) > f2. distance( );
}
```

When everything went well, you can now remove the `#if 0` around `solve`, and everything should compile without problems. If you find the code in `solve` ugly, you can define hash and equality objects instead of using pointers.

5. Finally, complete the function

```
std::list<move> printpath( const leveltable& levels,
                           fifteen f, size_t level );
```

which constructs a path towards `f`, assuming that `levels` contains `f` at level `level`.

P.S. Note that half of the states of the fifteen puzzle has no solution.