Programming Assignment #2 (due 11:59pm, 2014, 10/31)

Problem: Build an iRobot

In this programming assignment, you are asked to write a C++ program to perform shortest path search on a maze given the starting point and the trash location. An iRobot can move to up, down, left and right four directions. The objective is to find the shortest path from the starting point to the trash location.
Hint: Breadth-first search

Provided files: (1)main.cpp, (2)iRobot.cpp, (3)iRobot.h, (4)example

- **Main.cpp –** it checks the modified maze of **findTheTrash()** and can be changed if necessary for you to debug.
- **iRobot.cpp & iRobot.h –** These are the program files you need to implement. Thesr two files include the targeted function **findTheTrash()** in this homework, which returns the result of modified maze.
  These two files also include two finished functions called **buildMaze()** and **printMaze()**. **buildMaze()** reads the initial maze from a input file and **printMaze()** prints the current maze.

Input (example)

A maze example is shown below:

First line indicates the height of this maze.

Second line indicates the width of this maze.

The rest lines describe the maze with the following notations.

‘0’ represents the path that iRobot can walk.

‘2’ represents the block that iRobot cannot go through.

‘S’ represents the starting point of iRobot.

‘T’ represents the location of the trash.

```
6
8
2 2 2 2 2 2 2 2
2 S 0 0 0 0 0 2
2 0 2 2 0 2 0 2
2 0 2 0 2 0 0 2
2 0 0 0 2 T 0 2
2 2 2 2 2 2 2 2
```

Output

The output maze records the shortest path iRobot walk through.

‘1’ represents the path of iRobot.

```
2 2 2 2 2 2 2 2
2 S 1 1 1 1 1 2
2 0 2 2 0 2 1 2
2 0 2 0 2 1 1 2
2 0 0 0 2 T 0 2
2 2 2 2 2 2 2 2
```

Language

C or C++.

Platform

You may develop your software on UNIX/Linux.

Compile: $ g++ main.cpp iRobot.cpp

Execution: $ ./a.out

Submission

Please update the following materials to **online judge system** on the course website by the deadline, specifying your account and check the leader board.

(1) iRobot.h

(2) iRobot.cpp

Grading

(1) example correct: 40%

(2) 100% accuracy: 20%

(3) leader board ranking: 20% (ranking priority: accuracy > run time)

(4) hidden cases ranking: 20% (ranking priority: accuracy > run time)

Appendix:

Breadth-first search pseudo code:

```
1 procedure BFS(Graph,source):
2         create a queue Q
3         enqueue source onto Q
4         mark source
5         while Q is not empty:
6                 dequeue an item from Q into v
7                 for each edge e incident on v in Graph:
8                         let w be the other end of e
9                         if w is not marked:
10                                mark w
11                                enqueue w onto Q
```