**Computer Science I: Quiz 1 (Programs 4, 6)**

**Last: _______________________ , First: ________________________**

1) (10 pts) In program #4 (maze), the input was a grid of size R x C, indicating a starting square, blocked squares and squares to travel through. The goal of the assignment was to determine the shortest number of moves to get to the edge of the grid (in the assignment marked by the '~' character.) In terms of R and C, with proof, determine the run time of the posted solution. Please clearly justify your answer.

Run time = _________________

2) (5 pts) In the posted solution to program #4 (maze), the queue stored each grid location (ordered pair) as a single integer. When necessary, a single integer that was dequeued was "converted" into two separate values, x and y, representing the row and column of the desired square in the grid. Show that you understand this conversion by doing it for the following case: Let R = 20, the number of rows in the grid and C = 30, the number of columns in the grid. What is the row and column of the grid square designated by the integer 379? (Yes, you have to do some arithmetic on your own…) Note: the very first row is row 0 and very first column is column 0.

Row = ___ , Column = ___

3) (5 pts) In the posted solution for program #4 (maze), a queue is used. Each item in the queue represents a reachable grid square from the starting point. For each of these grid squares, the solution calculates the shortest distance from the starting point to the grid square. At any given point in time while the code is running, what is the largest difference between the distance from the starting point to two different grid squares that are simultaneously in the queue? Give proof of your answer.

Largest difference = ________

4) (5 pts) The function below is in the extra credit version of the posted solution to program #6 (boggle). Comments have been removed and the name of the function has been changed. What purpose does this function serve and why was it necessary in the context of this extra credit solution?

```
void function(trie* dictionary) {
    if (dictionary != NULL) {
        dictionary->inGrid = 0;
        int i;
        for (i=0; i<26; i++)
            function(dictionary->nextLetter[i]);
    }
}
```

5) (5 pts) Consider the following solveIt function written below. It has been modified from the solution posted for program #6 (boggle). (Note: in addition, the name of the trie was modified from dictionary to words so lines didn't wrap in the Word doc of this quiz.) If we were to use this function instead of the one in the posted solution, how does it assume the rules of the game have changed?

```
void solveIt(char prefix[], int curX, int curY, char board[][SIZE],
trie* words) {

  int curLen = strlen(prefix);
  if (isWord(prefix, words))printf("%s\n", prefix);
  if (!isPrefix(prefix, words)) return;

  int i;
  for (i=0; i<NUMDIR; i++) {
    if (inbounds(curX+DX[i], curY+DY[i])) {
      prefix[curLen] = board[curX+DX[i]][curY+DY[i]];
      prefix[curLen+1] = '\0';
      solveIt(prefix, curX+DX[i], curY+DY[i], board, words);
      prefix[curLen] = '\0';
    }
  }
}
```

6) (10 pts) If we were to remove the line

```
  if (!isPrefix(prefix, words)) return;
```

from the solveIt function in the posted solution to program #6 (boggle), then the run time of it would be prohibitive. Clearly explain why this line of code greatly speeds up the run time of the algorithm. Will it speed up the algorithm equally for all input grids? (Assume the input dictionary is the same as the one posted with the solution.) Why or why not?