Stat 598W: Homework 4

Due Saturday April 4th (sveinno@purdue.edu). Include a word/pdf document with your code and relevant output.

Problem 1: Given a sequence of integers a_i, $1 \leq i \leq n$, implement a $O(n^2)$ algorithm for finding all unique triplets (a_i, a_j, a_k) such that $a_i + a_j + a_k = 0$. Your functions should read a sequence of numbers from standard input and print all the triplets satisfying the condition.

Problem 2: Given an integer N, implement an algorithm to find whether a three-dimensional array exists such that the product of the dimensions is equal to N. The maximum index of each dimension should be greater than or equal to 2. For example, if $N = 6$ or $N = 10$ the function returns false, but if $N = 18$ it returns true because $2 * 2 * 3 = 18$. Your function should also print out the triplet found.

Problem 3: Write code that reads in pairs of numbers from the user, one by one. The first number b is between 2 and 16 and indicates the base of the second number. For example, 2 is binary and 16 hexadecimal. The second number then contains digits between 0 and $b - 1$, and if $b \geq 10$ it may also contain characters 'A', 'B', 'C', 'D', 'E', 'F' to represent 10, 11, 12, 13, 14, 15. Your function should find the base 10 representation of the second number.

Problem 4:

(a) Use BFS to print the nodes of a binary tree level by level, left to right.

(b) Use DFS to print the nodes of a binary tree in the order that they are visited.

(c) Implement the function node* = mirrorCopy(node* root) that returns a binary tree that is the mirror image of the binary tree passed as a parameter. For example, the mirror image of a tree with root node 1, left child 2, right child 3, is a tree with root 1, left child 3 and right child 2.

Create simple binary trees to make sure your functions work as intended.

Problem 5: Write a function that finds the shortest path through a three-dimensional maze. The maze contains h levels on top of each other, each split into m by n areas. Some areas have columns that support ceilings, others are free, and you can only move to free areas. You can jump up/down between levels as long as the area above/below you is free. Your initial location is at the top level, in area (0,0). Your goal is to reach area (m-1,n-1) at the bottom level. Your function should find the SHORTEST path, print out it's length and the direction chosen at each step. If no path exists, the function should say so.

The maze is a three-dimensional char-vector, your location is marked with '1', the destination is marked with '2', free areas are marked with '.', and columns are marked with 'o'.

```
enum Direction{EAST, WEST, SOUTH, NORTH, UP, DOWN};
typedef vector<vector<vector<char> > > array3D;
void solveMaze(array3d maze);

int main(){
    array3D maze;
    int h = 3;
    int m = 3;
    int n = 3;
    maze.resize(h);
    for(int i = 0; i < h; ++i){
        maze[i].resize(m);
        for(int j = 0; j < m; ++j)
            maze[i][j].resize(n);
    }

    //create maze manually
    maze[0][0][0] = '1';
    maze[h-1][m-1][n-1] = '2';
    ...
    //or read from a file
```

```
    ...

    solveMaze(maze);
    return 0;
}
```

Test your function using the maze below that has three levels, and requires 11 steps. The optimal path (or one of them) is EAST→EAST→SOUTH→SOUTH→WEST→WEST→DOWN→NORTH→EAST→DOWN→SOUTH→EAST.

```
1 . .
o o .
. . .
-----
o o o
. . o
. o o
-----
o o o
o . .
o . 2
```