Homework 3 of CSC 220 Algorithms, Fall 2020

given November 2, 2020, due November 23

It is the aim of this project to find a shortest path by BFS between two vertices in a subgraph of a two-dimensional integer grid graph of size SIZE by SIZE. The grid graph consists of points with integer coordinates in the range 0 to SIZE -1. Points are potential neighbors if they differ only in one coordinate, and in that coordinate by one; but some edges might be blocked. The graph is given as a three-dimensional array; the first two coordinates give the point, and the third coordinate selects the edge. In the two-dimensional grid, each point has only four neighbors, we number them clockwise starting with the edge going right. So `graph[i][j][1]` refers to the edge from point `(i,j)` up to the point `(i,j+1)`. The edge can have two values, FREE, or BLOCKED. We search a path that uses only FREE edges. SIZE, BLOCKED, and FREE are symbolic constants defined in the test file.

You write a function

```
void find_path(int *graph, int sx, int sy, int tx, int ty)
```

that finds the shortest path in the graph from vertex (sx,sy) to vertex (tx,ty) using the BFS algorithm. Each time you put an edge from (ax,ay) to (bx,by) in the queue, you call the function

```
void used_edge(int ax, int ay, int bx, int by)
```

which will color this edge green in the display. For each edge on the shortest path, it calls the function

```
void path_edge(int ax, int ay, int bx, int by)
```

which will color the edge red. If there is no shortest path from start to target point, you call only the function `used_edge`, and don't call the function `path_edge`. The functions `used_edge` and `path_edge` are provided with the test code. The test code uses the X Window system to visualize the graph and the edges chosen by your function.