

## Lab 3.1 writeup

Chen Hao Cheng

Adam Siefkas

Charles Davies

Hartland Brown

The problem with computing the distances between every possible pair of nodes and storing them is one of cost. There are 16 nodes and hence there are  $16^2$  connections between all of those nodes that would have to be stored. The implementation in the lab solves this problem by using an array's indices to represent each coordinate on the map, since the grid is indexed from 0-15 we can start at zero, and search above by adding the length of a row(4), below by subtracting the length, and left and right by adding or subtracting one. We can search (in the case of our code) breadthwise through all neighboring nodes and their children until we encounter the desired path.

```
int indexFromCoords(float xPos, float yPos)
{
    int xBlock = floor(xPos/blockSizeX); // ranges from 0-3
    int yBlock = floor(yPos/blockSizeY); // ranges from 0-3
    int blockIndex = 4*yBlock + xBlock; // ranges from 0-15
    return blockIndex;
}
```

float coordsFromIndex(int i, char xy)//since you cant return two values, we are going to carry a flag of which value to return.

```
{
    int xI = i%4;
    int yI = i/4;
```

```
float xPos = (xI+0.5)*blockSizeX;
float yPos = (yI+0.5)*blockSizeY;
if (xy == 'x'){return xPos;}
if (xy == 'y'){return yPos;}
}
```

```
int BFS(int startIndex, int finishIndex)
{
    QueueArray <int> queue;
    int checked[16];
    for (int a = 0; a < 16; a++) checked[a] = 0;
    int parents[16];
    for (int a = 0; a < 16; a++) parents[a] = 0;
    queue.enqueue(startIndex);
    checked[startIndex]=1;
    while (!queue.isEmpty()){
        int cur = queue.dequeue();
```

```
// sparki.clearLCD();
// sparki.println(cur);
```

```

if (cur == finishIndex){break;}
if (blockedSquares[cur/4][cur%4] != 0){

    int c1 = cur+1;
    int c2 = cur-1;
    int c3 = cur+4;
    int c4 = cur-4;

    if((c1>=0) && (c1<=15) && (checked[c1]==0)){
//    sparki.println(c1);
        Serial.print(c1);
        checked[c1] = 1;
        parents[c1] = cur;
        queue.enqueue(c1);
    }
    if((c2>=0) && (c2<=15) && (checked[c2]==0)){
//    sparki.println(c2);
        Serial.print(c2);
        checked[c2] = 1;
        parents[c2] = cur;
        queue.enqueue(c2);
    }
    if((c3>=0) && (c3<=15) && (checked[c3]==0)){
//    sparki.println(c3);
        Serial.print(c3);
        checked[c3] = 1;
        parents[c3] = cur;
        queue.enqueue(c3);
    }
    if((c4>=0) && (c4<=15) && (checked[c4]==0)){
//    sparki.println(c4);
        Serial.print(c4);
        checked[c4] = 1;
        parents[c4] = cur;
        queue.enqueue(c4);
    }
//    sparki.updateLCD();
}
}

int step1 = parents[finishIndex];
int stepCount = 1;
while(step1 != startIndex){
    stepCount += 1;
    step1 = parents[step1];
}
return stepCount;
}

```

