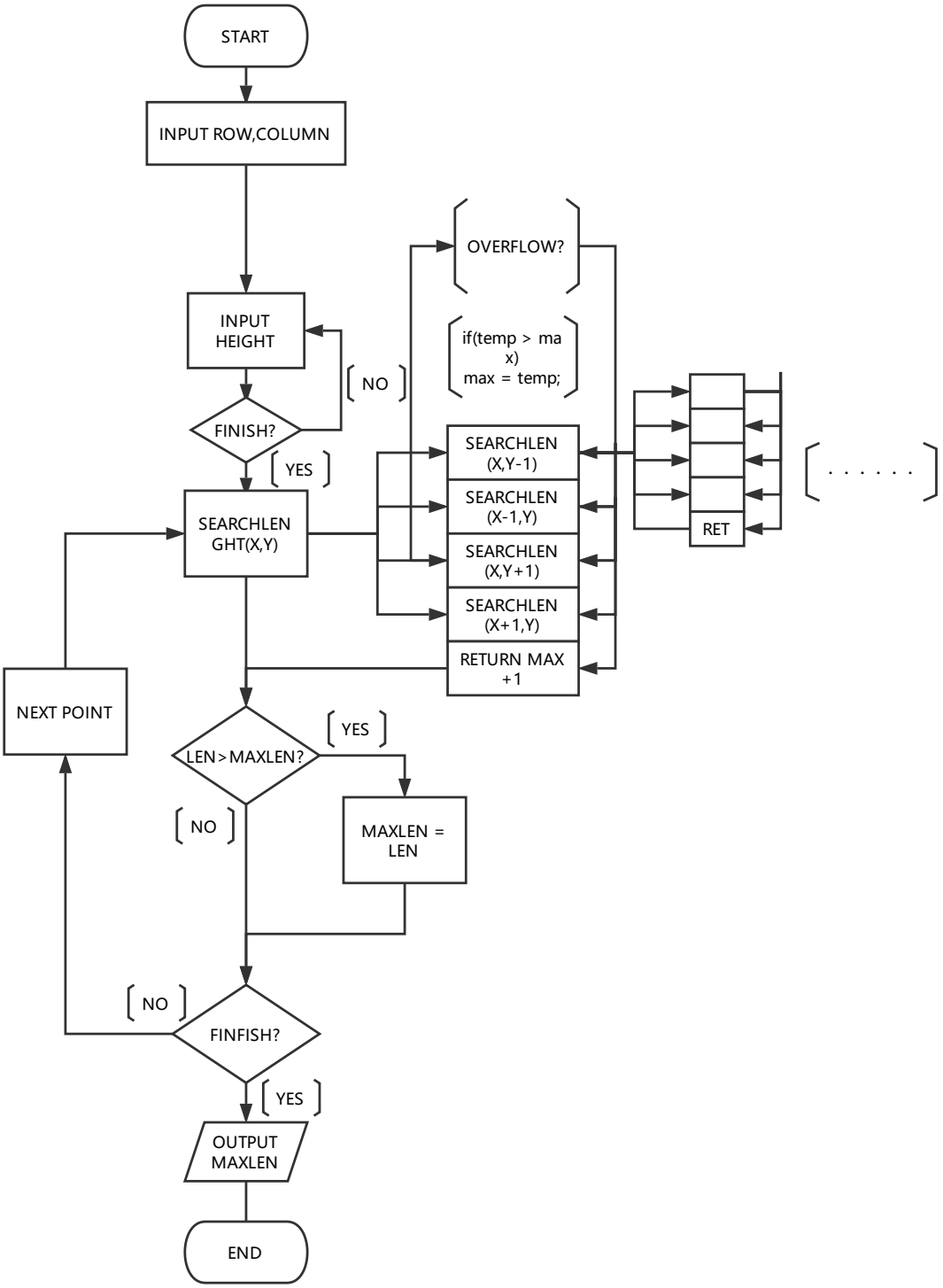
The 4th lab report

(Due on Jul.17th)

1 .Program algorithm:



2.Brief Explanations

The function of this program is actually to recursively find the shortest path in a two-dimensional array.

Before recursion starts, we need to do two things. First, calculate the top and bottom of the contiguous storage area of the two-dimensional array, and then initialize the R7 to the address of the recursive exit.

Then we traverse each point in the two-dimensional array, recursively seeking the shortest path for each point. The main process is as follows.

First, push the position of first in the array (represented by R5) point and return address into stack. recursive attempts are made at the next point (East, north, South, West) of each direction starting from this point. If it works, then go to the next point in that direction and push return address and next point's position in the array (represented by R5) into the stack. If the recursive attempt fails, then skip to the next direction and continue trying; if four directions fail, then the recursion is at its end and enters the recursive return mode.

(This involves the question of boundary judgment, especially when the direction is west and east. My solution is to start with bottom (top), list all overflow possibilities and compare them with the existing addresses)

In the recursive return mode, we need to do the following things: first, calculate the stack pointer offset (compared with the bottom) of the stack, which is the number of points (path length) after two times. if the LEN value in a certain area is smaller than this result, it will be stores in LEN area. Then, pop up the corresponding array position and return address of the point from stack, and then return to the upper level.

When all possible paths from one point are tested, the LEN and the existing MAXLEN are compared, and a larger one is stored in the MAXLEN. Then empty LEN, return to the beginning of the program, and continue to traverse the next point. When the value of R5 exceeds BOTTOM, the end of the traversal, at this point, the MAXLENGTH value is two times the desired result. Divided by two (using subtraction and loop Implementation) and stored in R2, the final result is obtained.

3.Source Code (in appendix)