Assignment Report 4

1. Solution

At first, I used a loop to calculate Row*Column and store it. Then, I set the stack of main function. **(PS. Each function or subroutine has 4 positions in the stack, from the bottom to the top they are RETURN VALUE, RETURN ADDRESS, FRAME POINTER, and CURRENT NUMBER (from R*C down to 1))**

Then I used memorized search to improve the program. If the current point is not 0 then it is already the max number, then just skip the procedure, pop the stack, and return, or search the four directions to find the max numbers of the path by recursion.

The four directions have 4 different ways to judge if the current number has crossed the boundaries. **I minus the current number repeatedly** to see if it is at **the last column**, which is similar to how I see if it is at the first column. **I minus the start address of data** to see if it is at **the first line**, which is similar to how I see if it is at the last line.

If the current number' s next move is in the matrix, then I compare the data value of it with the current number' s. If the next move is smaller, then push the stack and get into the subroutine.

After each number of data has found its max number of skiing length, I compare it to R2, which stores the max number. If it' s bigger then renew R2.

Finally, the max number needs plus 1 because the last point had not been counted.

3. Source Code

```
.ORIG x3000
STACK .BLKW 1
LDI R0,ROW
LDI R1,COL
LD R4,DATA
LEA R3,RES
LEA R6,STACK
AND R2,R2,#0
ADD R2,R2,R1
ADD R2,R2,xFFFF
MULTI ADD R0,R0,R1
ADD R2,R2,xFFFF
BRz MULTIDONE
BRnzp MULTI
MULTIDONE ST R0,TOTAL
;R0 = ROW*COL AND STORE IT
AND R2,R2,#0
ADD R6,R6,#-3
STR R0,R6,#0
;SET THE MAIN'S STACK
LOOP JSR FIND
;FIND THE CURRENT
;NUMBER'S MAX VALUE
LDR R1,R6,#0
NOT R1,R1
ADD R1,R1,#1
ADD R1,R1,R2
;COMPARE WITH R2
BRzp NO
LDR R2,R6,#0
NO ADD R0,R0,#-1
ADD R6,R6,#1
STR R0,R6,#0
ADD R0,R0,#0
;PUSH STACK TO NEXT NUMBER
BRz END
BRnzp LOOP
END ADD R2,R2,#1
HALT
;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;SUNBROUTINE

FIND ADD R6,R6,#-2
STR R7,R6,#0
ADD R6,R6,#-1
STR R5,R6,#0
ADD R5,R6,#-1
ADD R6,R6,#-1
;PUSH THE STACK
;AND STORE R7 AND R5
LDR R1,R6,#4
LEA R3,RES
ADD R3,R3,R1
ADD R3,R3,#-1
;R3 IS THE CURRENT POSITION
LDR R1,R3,#0
BRnp DONE
;IF IT IS NOT 0 THEN DONE
;MEMORIZED RESEARCH
```

```
61   LDR R1,R6,#4
62   LDI R3,COL
63   NOT R3,R3
64   ADD R3,R3,#1
65   LOOP1 ADD R1,R1,R3
66   BRz NEXT_1
67   BRn INSIDE1
68   BRp LOOP1
69   ;IF IT IS IN THE MATRIX
70   INSIDE1 LDR R1,R6,#4
71   LD R4,DATA
72   ADD R4,R4,R1
73   ADD R4,R4,#-1
74   ST R5,SR5
75   LDR R5,R4,#0
76   LDR R1,R4,#1
77   NOT R1,R1
78   ADD R1,R1,#1
79   ADD R1,R1,R5
80   LD R5,SR5
81   ADD R1,R1,#0
82   ;IF NEXT MOVE IS SMALLER
83   ;THEN GO INTO RECURSION
84   BRnz NEXT_1
85   LDR R1,R6,#4
86   ADD R1,R1,#1
87   STR R1,R6,#0
88   JSR FIND
89   LDR R1,R6,#0
90   ADD R6,R6,#1
91   ADD R1,R1,#1
92   ST R2,TEMP1
93   LDR R2,R6,#4
94   LEA R3,RES
95   ADD R3,R3,R2
96   LD R2,TEMP1
97   ADD R3,R3,#-1
98   LDR R4,R3,#0
99   NOT R4,R4
100  ADD R4,R4,#1
101  ADD R4,R1,R4
102  ;COMPARE AND RENEW THE
103  ;RESULT OF EACH NUMBER
104  BRnz NEXT_1
105  STR R1,R3,#0
106  ;
```

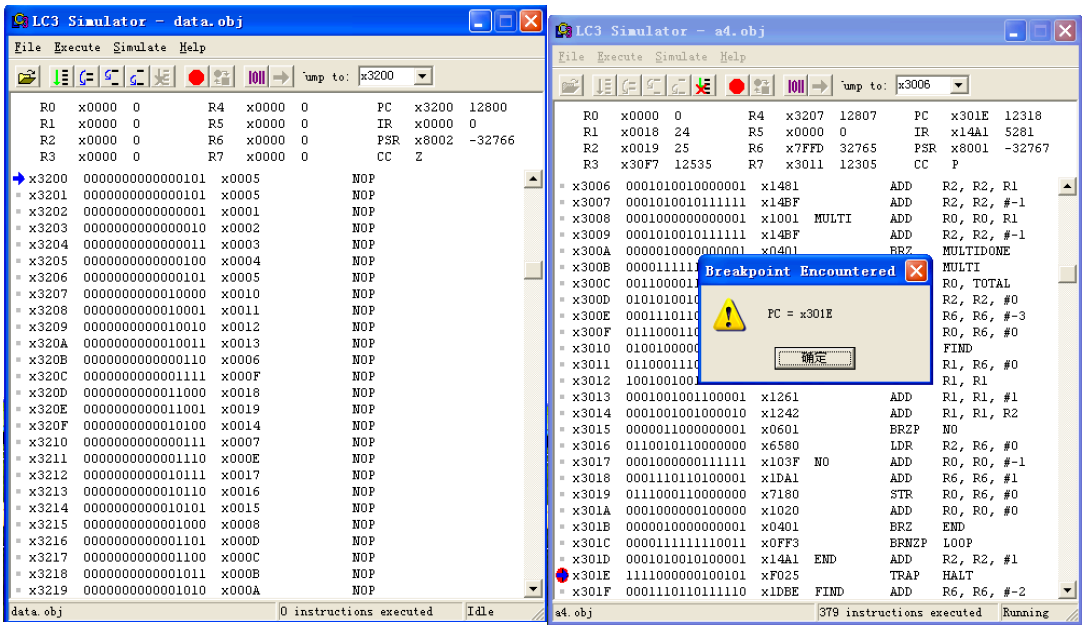
```
107  NEXT_1 LDR R1,R6,#4
108  LDI R3,COL
109  ADD R1,R1,R3
110  ADD R1,R1,#-1
111  NOT R3,R3
112  ADD R3,R3,#1
113  LOOP2 ADD R1,R1,R3
114  BRz NEXT_2
115  BRn INSIDE2
116  BRp LOOP2
117  ;IF IT IS IN THE MATRIX
118  INSIDE2 LDR R1,R6,#4
119  LD R4,DATA
120  ADD R4,R4,R1
121  ADD R4,R4,#-1
122  ST R5,SR5
123  LDR R5,R4,#0
124  LDR R1,R4,#-1
125  NOT R1,R1
126  ADD R1,R1,#1
127  ADD R1,R1,R5
128  LD R5,SR5
129  ADD R1,R1,#0
130  ;IF NEXT MOVE IS SMALLER
131  ;THEN GO INTO RECURSION
132  BRnz NEXT_2
133  LDR R1,R6,#4
134  ADD R1,R1,#-1
135  STR R1,R6,#0
136  JSR FIND
137  LDR R1,R6,#0
138  ADD R6,R6,#1
139  ADD R1,R1,#1
140  ST R2,TEMP2
141  LDR R2,R6,#4
142  LEA R3,RES
143  ADD R3,R3,R2
144  LD R2,TEMP2
145  ADD R3,R3,#-1
146  LDR R4,R3,#0
147  NOT R4,R4
148  ADD R4,R4,#1
149  ADD R4,R1,R4
150  ;COMPARE AND RENEW THE
151  ;RESULT OF EACH NUMBER
152  BRnz NEXT_2
153  STR R1,R3,#0
```

```
156   NEXT_2 LDR R1,R6,#4
157   LDI R3,COL
158   ADD R1,R1,R3
159   LD R3,TOTAL
160   NOT R3,R3
161   ADD R3,R3,#1
162   ADD R3,R1,R3
163   ;IF IT IS IN THE MATRIX
164   BRp NEXT_3
165   LDR R1,R6,#4
166   LD R4,DATA
167   ADD R4,R4,R1
168   ADD R4,R4,#-1
169   ST R5,SR5
170   ST R6,SR6
171   LDI R6,COL
172   LDR R5,R4,#0
173   ADD R4,R4,R6
174   LDR R1,R4,#0
175   NOT R1,R1
176   ADD R1,R1,#1
177   ADD R1,R1,R5
178   LD R5,SR5
179   LD R6,SR6
180   ADD R1,R1,#0
181   ;IF NEXT MOVE IS SMALLER
182   ;THEN GO INTO RECURSION
183   BRnz NEXT_3
184   LDR R1,R6,#4
185   LDI R3,COL
186   ADD R1,R1,R3
187   STR R1,R6,#0
188   JSR FIND
189   LDR R1,R6,#0
190   ADD R6,R6,#1
191   ADD R1,R1,#1
192   ST R2,TEMP3
193   LDR R2,R6,#4
194   LEA R3,RES
195   ADD R3,R3,R2
196   LD R2,TEMP3
197   ADD R3,R3,#-1
198   LDR R4,R3,#0
199   NOT R4,R4
200   ADD R4,R4,#1
201   ADD R4,R1,R4
202   ;COMPARE AND RENEW THE
203   ;RESULT OF EACH NUMBER
204   BRnz NEXT_3
205   STR R1,R3,#0
```

```
208   NEXT_3 LDR R1,R6,#4
209   LDI R3,COL
210   NOT R3,R3
211   ADD R3,R3,#1
212   ADD R3,R1,R3
213   ;IF IT IS IN THE MATRIX
214   BRnz DONE
215   LDR R1,R6,#4
216   LD R4,DATA
217   ADD R4,R4,R1
218   ADD R4,R4,#-1
219   ST R5,SR5
220   ST R6,SR6
221   LDI R6,COL
222   NOT R6,R6
223   ADD R6,R6,#1
224   LDR R5,R4,#0
225   ADD R4,R4,R6
226   LDR R1,R4,#0
227   NOT R1,R1
228   ADD R1,R1,#1
229   ADD R1,R1,R5
230   LD R5,SR5
231   LD R6,SR6
232   ADD R1,R1,#0
233   ;IF NEXT MOVE IS SMALLER
234   ;THEN GO INTO RECURSION
235   BRnz DONE
236   LDR R1,R6,#4
237   LDI R3,COL
238   NOT R3,R3
239   ADD R3,R3,#1
240   ADD R1,R1,R3
241   STR R1,R6,#0
242   JSR FIND
243   LDR R1,R6,#0
244   ADD R6,R6,#1
245   ADD R1,R1,#1
246   ST R2,TEMP4
247   LDR R2,R6,#4
248   LEA R3,RES
249   ADD R3,R3,R2
250   LD R2,TEMP4
251   ADD R3,R3,#-1
252   LDR R4,R3,#0
253   NOT R4,R4
254   ADD R4,R4,#1
255   ADD R4,R1,R4
```

```
256   ;COMPARE AND RENEW THE
257   ;RESULT OF EACH NUMBER
258   BRnz DONE
259   STR R1,R3,#0
260   ;RETURN VALUE
261   DONE LDR R1,R6,#4
262   LEA R3,RES
263   ADD R3,R3,R1
264   ADD R3,R3,#-1
265   LDR R1,R3,#0
266   STR R1,R6,#3
267   ADD R6,R5,#1
268   LDR R5,R6,#0
269   ADD R6,R6,#1
270   LDR R7,R6,#0
271   ADD R6,R6,#1
272   RET
273   ;;;;;;;;;;;;;;;;;;;;;;;;
274
275   ROW .FILL x3200
276   COL .FILL x3201
277   DATA .FILL x3202
278   SR0 .BLKW 1
279   SR1 .BLKW 1
280   SR2 .BLKW 1
281   SR3 .BLKW 1
282   SR4 .BLKW 1
283   SR5 .BLKW 1
284   SR6 .BLKW 1
285   TOTAL .BLKW 1
286   TEMP1 .BLKW 1
287   TEMP2 .BLKW 1
288   TEMP3 .BLKW 1
289   TEMP4 .BLKW 1
290   RES .BLKW 100
291   .END
```

4. Snapshots



(Source code in txt if TA wants to test it)

```
.ORIG x3000
STACK .BLKW 1
LDI R0,ROW
LDI R1,COL
LD R4,DATA
LEA R3,RES
LEA R6,STACK
AND R2,R2,#0
ADD R2,R2,R1
ADD R2,R2,xFFFF
MULTI ADD R0,R0,R1
ADD R2,R2,xFFFF
BRz MULTIDONE
BRnzp MULTI
MULTIDONE ST R0,TOTAL
;R0 = ROW*COL AND STORE IT
AND R2,R2,#0
ADD R6,R6,#-3
STR R0,R6,#0
;SET THE MAIN'S STACK
LOOP JSR FIND
;FIND THE CURRENT
;NUMBER'S MAX VALUE
LDR R1,R6,#0
NOT R1,R1
ADD R1,R1,#1
ADD R1,R1,R2
;COMPARE WITH R2
BRzp NO
LDR R2,R6,#0
NO ADD R0,R0,#-1
ADD R6,R6,#1
STR R0,R6,#0
ADD R0,R0,#0
;PUSH STACK TO NEXT NUMBER
BRz END
BRnzp LOOP
END ADD R2,R2,#1
HALT
;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;SUNBROUTINE

FIND ADD R6,R6,#-2
STR R7,R6,#0
ADD R6,R6,#-1
STR R5,R6,#0
ADD R5,R6,#-1
ADD R6,R6,#-1
;PUSH THE STACK
;AND STORE R7 AND R5
LDR R1,R6,#4
LEA R3,RES
ADD R3,R3,R1
ADD R3,R3,#-1
;R3 IS THE CURRENT POSITION
LDR R1,R3,#0
BRnp DONE
;IF IT IS NOT 0 THEN DONE
;MEMORIZED RESEARCH

LDR R1,R6,#4
LDI R3,COL
NOT R3,R3
ADD R3,R3,#1
LOOP1 ADD R1,R1,R3
BRz NEXT_1
BRn INSIDE1
BRp LOOP1
;IF IT IS IN THE MATRIX
INSIDE1 LDR R1,R6,#4
LD R4,DATA
ADD R4,R4,R1
ADD R4,R4,#-1
ST R5,SR5
LDR R5,R4,#0
LDR R1,R4,#1
NOT R1,R1
ADD R1,R1,#1
ADD R1,R1,R5
LD R5,SR5
ADD R1,R1,#0
;IF NEXT MOVE IS SMALLER
;THEN GO INTO RECURSION
BRnz NEXT_1
```

```
LDR R1,R6,#4
ADD R1,R1,#1
STR R1,R6,#0
JSR FIND
LDR R1,R6,#0
ADD R6,R6,#1
ADD R1,R1,#1
ST R2,TEMP1
LDR R2,R6,#4
LEA R3,RES
ADD R3,R3,R2
LD R2,TEMP1
ADD R3,R3,#-1
LDR R4,R3,#0
NOT R4,R4
ADD R4,R4,#1
ADD R4,R1,R4
;COMPARE AND RENEW THE
;RESULT OF EACH NUMBER
BRnz NEXT_1
STR R1,R3,#0
;
NEXT_1 LDR R1,R6,#4
LDI R3,COL
ADD R1,R1,R3
ADD R1,R1,#-1
NOT R3,R3
ADD R3,R3,#1
LOOP2 ADD R1,R1,R3
BRz NEXT_2
BRn INSIDE2
BRp LOOP2
;IF IT IS IN THE MATRIX
INSIDE2 LDR R1,R6,#4
LD R4,DATA
ADD R4,R4,R1
ADD R4,R4,#-1
ST R5,SR5
LDR R5,R4,#0
LDR R1,R4,#-1
NOT R1,R1
ADD R1,R1,#1
ADD R1,R1,R5
LD R5,SR5
ADD R1,R1,#0
;IF NEXT MOVE IS SMALLER
;THEN GO INTO RECURSION
BRnz NEXT_2
LDR R1,R6,#4
ADD R1,R1,#-1
STR R1,R6,#0
JSR FIND
LDR R1,R6,#0
ADD R6,R6,#1
ADD R1,R1,#1
ST R2,TEMP2
LDR R2,R6,#4
LEA R3,RES
ADD R3,R3,R2
LD R2,TEMP2
ADD R3,R3,#-1
LDR R4,R3,#0
NOT R4,R4
ADD R4,R4,#1
ADD R4,R1,R4
;COMPARE AND RENEW THE
;RESULT OF EACH NUMBER
BRnz NEXT_2
STR R1,R3,#0
;

NEXT_2 LDR R1,R6,#4
LDI R3,COL
ADD R1,R1,R3
LD R3,TOTAL
NOT R3,R3
ADD R3,R3,#1
ADD R3,R1,R3
;IF IT IS IN THE MATRIX
BRp NEXT_3
LDR R1,R6,#4
LD R4,DATA
ADD R4,R4,R1
ADD R4,R4,#-1
ST R5,SR5
ST R6,SR6
LDI R6,COL
LDR R5,R4,#0
```

```
ADD R4,R4,R6
LDR R1,R4,#0
NOT R1,R1
ADD R1,R1,#1
ADD R1,R1,R5
LD R5,SR5
LD R6,SR6
ADD R1,R1,#0
;IF NEXT MOVE IS SMALLER
;THEN GO INTO RECURSION
BRnz NEXT_3
LDR R1,R6,#4
LDI R3,COL
ADD R1,R1,R3
STR R1,R6,#0
JSR FIND
LDR R1,R6,#0
ADD R6,R6,#1
ADD R1,R1,#1
ST R2,TEMP3
LDR R2,R6,#4
LEA R3,RES
ADD R3,R3,R2
LD R2,TEMP3
ADD R3,R3,#-1
LDR R4,R3,#0
NOT R4,R4
ADD R4,R4,#1
ADD R4,R1,R4
;COMPARE AND RENEW THE
;RESULT OF EACH NUMBER
BRnz NEXT_3
STR R1,R3,#0
;

NEXT_3 LDR R1,R6,#4
LDI R3,COL
NOT R3,R3
ADD R3,R3,#1
ADD R3,R1,R3
;IF IT IS IN THE MATRIX
BRnz DONE
LDR R1,R6,#4
LD R4,DATA
ADD R4,R4,R1
ADD R4,R4,#-1
ST R5,SR5
ST R6,SR6
LDI R6,COL
NOT R6,R6
ADD R6,R6,#1
LDR R5,R4,#0
ADD R4,R4,R6
LDR R1,R4,#0
NOT R1,R1
ADD R1,R1,#1
ADD R1,R1,R5
LD R5,SR5
LD R6,SR6
ADD R1,R1,#0
;IF NEXT MOVE IS SMALLER
;THEN GO INTO RECURSION
BRnz DONE
LDR R1,R6,#4
LDI R3,COL
NOT R3,R3
ADD R3,R3,#1
ADD R1,R1,R3
STR R1,R6,#0
JSR FIND
LDR R1,R6,#0
ADD R6,R6,#1
ADD R1,R1,#1
ST R2,TEMP4
LDR R2,R6,#4
LEA R3,RES
ADD R3,R3,R2
LD R2,TEMP4
ADD R3,R3,#-1
LDR R4,R3,#0
NOT R4,R4
ADD R4,R4,#1
ADD R4,R1,R4
;COMPARE AND RENEW THE
;RESULT OF EACH NUMBER
BRnz DONE
STR R1,R3,#0
;RETURN VALUE
```

```
DONE LDR R1,R6,#4
LEA R3,RES
ADD R3,R3,R1
ADD R3,R3,#-1
LDR R1,R3,#0
STR R1,R6,#3
ADD R6,R5,#1
LDR R5,R6,#0
ADD R6,R6,#1
LDR R7,R6,#0
ADD R6,R6,#1
RET
;;;;;;;;;;;;;;;;;;;;;;;;;;

ROW .FILL x3200
COL .FILL x3201
DATA .FILL x3202
SR0 .BLKW 1
SR1 .BLKW 1
SR2 .BLKW 1
SR3 .BLKW 1
SR4 .BLKW 1
SR5 .BLKW 1
SR6 .BLKW 1
TOTAL .BLKW 1
TEMP1 .BLKW 1
TEMP2 .BLKW 1
TEMP3 .BLKW 1
TEMP4 .BLKW 1
RES .BLKW 100
.END
```