# Lab 5: Skiing

Professor Patt likes skiing. When skiing, one would slide down from a higher place to a lower place to gain speed. But when one arrives at the bottom, he has to walk up or wait for others to pick him up. As a computer man, Patt always wants to choose the longest distance he can slide.

Your job is to write a program to help Professor Patt. Your program reads the map where he is skiing and tells the longest distance he can slide.

> This lab is associated with the following chapters and sections. If you have no idea how to do this lab, please review the corresponding parts of the book.
>
> - 8.2 The Stack
> - 8.3 Recursion, a Powerful Technique When Used Appropriately
> - 17 Recursion, especially 17.7 Escaping a Maze

## Implementation Details

- You are required to write in **LC-3 assembly language**.
- Your program should start at x3000, which means the first instruction of your program is located in position x3000.
- Your program **must** use recursive ways to solve this problem.

## The map

Suppose the skiing field is a rectangular area $N \times M$. A map is provided as a two-dimensional array to indicate the altitude of each point in the skiing field. For example, the following two-dimensional array illustrates a sample skiing field in $3 \times 4$.

```
89   88   86   83
79   73   90   80
60   69   73   77
```

The map is stored from memory location x4000. The above map is stored in the following way:

```
        .ORIG   x4000
        .FILL   #3  ; N
        .FILL   #4  ; M
        .FILL   #89 ; the map
        .FILL   #88
        .FILL   #86
        .FILL   #83
        .FILL   #79
        .FILL   #73
        .FILL   #90
        .FILL   #80
```

```
        .FILL   #60
        .FILL   #69
        .FILL   #73
        .FILL   #77
        .END
```

# The longest distance

One can slide from a point $A$ to another $B$, if and only if the altitude of $B$ is lower than the altitude of $A$. So in the above case, the longest path is:

```
89 -> 88 -> 86 -> 83 -> 80 -> 77 -> 73 -> 69 -> 60   (9)
```

Notice that if the start point is 90, the longest path is shorter than starting from point 89:

```
90 -> 86 -> 83 -> 80 -> 77 -> 73 -> 69 -> 60   (8)
```

Professor Patt can start from any point. So in this case, we say the longest distance is 9, starting from point 89.

- After your program executing, the result longest distance should be stored in R2.
- Your program does not need to tell where to start.

# Limitations

- The size of the map: $1 \leq N \times M \leq 50$.
- The attitude in the map $x$: $0 \leq x \leq 100$.
- Acceptable program length: less than 300 lines.
- The time complexity would not affect your score.

# Grading

Lab 5 takes 10% of the total score, consisting of Check part (50%) and Report part (50%).

## Check part (50%)

- Find a TA to check your code in person. TAs may ask you questions when grading your lab assignment. You will get 100%, 80% or 60% of the checking score according to your response.
- You can try again if you fail in checking, but there will be a penalty of -10% (of checking part) for each try.
- We suggest you to write enough comments in your code so that you will be aware of what's going on in your program and confident to answer TA's questions.

# Report part (50%)

- English report should be concise and carrying main ideas. Try to use the report to convince TAs that you complete the task by yourself.

- Your lab report should *at least* contains the following contents:
  - Your algorithm. To make it clear, you can use figures, tables or any other easy-to-understand appearance.
  - Essential parts of your code with sufficient comments. Please only select the most important code phases and explain them.
  - The questions that TA asked you, and answers.

- 2~4 A4 pages. No template provided. Be sure to make it readable.

# Penalty

- **Wrong Answer**: -10% of Check part each time.
- **Delay**: -20% of the corresponding part per day.
- **Cheating**: -100% of this lab. Additionally, -10% of the final score of this course.