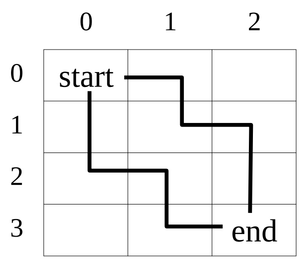
CS 218 – MIPS Assignment #5

Purpose:	Become familiar with the MIPS Instruction Set, and the MIPS procedure calling convention, and basic recursion.
Points:	100

Assignment:

Write a MIPS assembly language program to determine the number of possible paths through a two-dimensional grid. The only allowed moves are one step to the right or one step down. As such, the start coordinates must be < the end coordinates. The function will accept a start coordinate (row,col) and a final end coordinate (row,col). Moving from the starting location, (0,0) in this example, going to the end location, (3,2) in this example, can be performed in 10 different ways. Two different ways (out of the 10) are shown in the example above. The function must be recursive.



Using the provided main, write the following assembly language procedures:

- Write a MIPS assembly language function, ***readCoords()***. The number of steps must be between (0,0) and (COORD_MAX,COORD_MAX). Additionally, the end coordinates (both) must be greater than the start coordinates. If invalid input is provided, display an error message and re-prompt.

- Write a recursive MIPS assembly language function, ***countPaths(currRow, currCol, endRow, endCol)***, to determine the number of possible paths through a two-dimensional grid. Must be recursive for credit.

$$countPaths(r, c, er, ec) = \begin{cases} 0 & \text{if } r > er \text{ or } c > ec \\ 1 & \text{if } r = er \text{ or } c = ec \\ countPaths(r, c+1, er, ec) + \\ \quad countPaths(r+1, c, er, ec) & \text{otherwise} \end{cases}$$

- Write a MIPS assembly language function, ***prtResults(startRow, startCol, endRow, endCol, totalCount)***, to print the formatted results. Refer to the example output for formatting.

Refer to the example for formatting. You may define additional variables as required.

To understand what recursion is,

you must first understand recursion.

Example Output:

The following are some example sessions:

```
**********************************************

MIPS Assignment #5
Count Grid Paths Program
  Enter Start Coordinates ROW: 1
  Enter Start Coordinates COL: 1
  Enter End Coordinates ROW: 8
  Enter End Coordinates COL: 15


From a start coordinate of (1,1), to an end coordinate of (8,15),
there are 116280 ways traverse the grid.

You have reached recursive nirvana.
Program Terminated.
```

```
**********************************************

MIPS Assignment #5
Count Grid Paths Program
  Enter Start Coordinates ROW: 0
  Enter Start Coordinates COL: 0
  Enter End Coordinates ROW: 0
  Enter End Coordinates COL: 0

Error, end coordinates must be > then start coordinates.
Please re-enter.
  Enter Start Coordinates ROW: -1

Error, invalid coordinate value.  Please re-enter.
  Enter Start Coordinates ROW: 0
  Enter Start Coordinates COL: 0
  Enter End Coordinates ROW: 12
  Enter End Coordinates COL: -12

Error, invalid coordinate value.
Please re-enter.
  Enter End Coordinates COL: 12


From a start coordinate of (0,0), to an end coordinate of (12,12),
there are 2704156 ways traverse the grid.

You have reached recursive nirvana.
Program Terminated.
```

Submission:

- All source files must assemble and execute with QtSpim/SPIM MIPS simulator.
- Submit source file
  - Submit a copy of the program source file via the on-line submission
- Once you submit, the system will score the project and provide feedback.
  - If you do not get full score, you can (and should) correct and resubmit.
  - You can re-submit an unlimited number of times before the due date/time (at a maximum rate of 5 submissions per hour).
- Late submissions will be accepted for a period of 24 hours after the due date/time for any given assignment. Late submissions will be subject to a ~2% reduction in points per an hour late. If you submit 1 minute - 1 hour late -2%, 1-2 hours late -4%, … , 23-24 hours late -50%. This means after 24 hours late submissions will receive an automatic 0.

Program Header Block

All source files must include your name, section number, assignment, NSHE number, and program description. The required format is as follows:

```
#  Name: <your name>
#  NSHE ID: <your id>
#  Section: <section>
#  Assignment: <assignment number>
#  Description: <short description of program goes here>
```

Failure to include your name in this format will result in a reduction of points.

Scoring Rubric

Scoring will include functionality, code quality, and documentation. Below is a summary of the scoring rubric for this assignment.

Criteria	Weight	Summary
Assemble	-	Failure to assemble will result in a score of 0.
Program Header	3%	Must include header block in the required format (see above).
General Comments	7%	Must include an appropriate level of program documentation.
Program Functionality (and on-time)	90%	Program must meet the functional requirements as outlined in the assignment. Must be submitted on time for full score.