

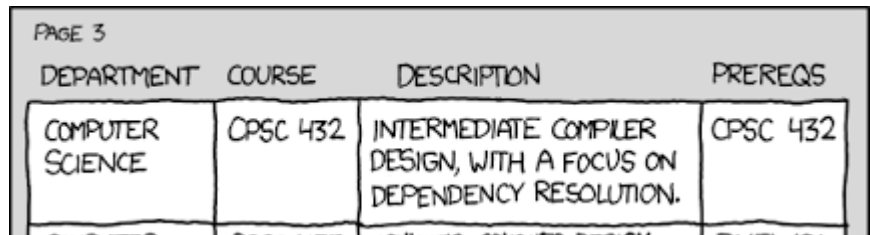
CS 218 – MIPS Assignment #5

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

Points: 80

Assignment

Write a recursive assembly language program to compute and display Pascal's Triangle¹. In Pascal's triangle each element is the sum of the two elements above it, except for edge elements, which are 1.



DEPARTMENT	COURSE	DESCRIPTION	PREREQS
COMPUTER SCIENCE	CPSC 432	INTERMEDIATE COMPILER DESIGN, WITH A FOCUS ON DEPENDENCY RESOLUTION.	CPSC 432

The main program should call the following MIPS functions:

- Value returning assembly language function, *readRows()*, to read a number from the user and ensure that the number is between 1 and 25 (inclusive). Includes prompting and error checking. The function should re-prompt for incorrect input. Returns result in *\$v0*.
- Value returning assembly language function, *pascal()*, to compute the k^{th} element of the n^{th} row of the Pascal triangle:

$$pascal(n, k) = \begin{cases} 1 & \text{if } k=0 \text{ or } k=n \\ pascal(n-1, k-1) + pascal(n-1, k) & \text{otherwise} \end{cases}$$

The Pascal function *must* be computed recursively. Returns result in *\$v0*.

- Void assembly language function, *displyPascalsTriangle()*, to display the pascal triangle. Routine must compute the Pascal number (via call), display appropriate row headers (see example), and then print the pascal numbers. This function must use a function to print the formatted pascal number. Since the *pascal()* function computes a single Pascal number, the *pascal()* function will need to be called in a nested loop (rows and columns). After display the "row #: " message (provided), the leading blanks can be determined by printing the *spc* string (3 blanks) times the maximum number of rows minus the current row.
- Void assembly language function, *prtPnum()*, print the formatted *Pascal(n,k)* number. In order to provide the correct triangle output the number will need to be printed in a formatted manner. This function must print an appropriate number of blanks based on the size of the number. Refer to the example output for formatting.
- Value returning assembly language function, *checkAgain()*, to see if the user wants to display another pascals triangle. The function should return TRUE for 'Y' or 'y' and FALSE for 'N' or 'n'. Note, QtSpim simulator requires single quotes for characters comparisons. The function should re-prompt for incorrect input. If the user enters 'N' or 'n', the routine should display a final message "Game Over." and "Thank you for playing.". You are expected to read all characters (ie., use read string) and check the first charatcer. Refer to example output for formatting. Returns result in *\$v0*.

¹ For more information, refer to: http://en.wikipedia.org/wiki/Pascals_triangle

Example Output

Below is an example of the program output.

```
MIPS Assignment #5
Pascal's Triangle Program

Enter number of rows in triangle (1-25): 11

row 0:
          1
row 1:
        1  1
row 2:
      1  2  1
row 3:
    1  3  3  1
row 4:
  1  4  6  4  1
row 5:
1  5 10 10  5  1
row 6:
 1  6 15 20 15  6  1
row 7:
 1  7 21 35 35 21  7  1
row 8:
 1  8 28 56 70 56 28  8  1
row 9:
 1  9 36 84 126 126 84 36  9  1
row 10:
 1 10 45 120 210 252 210 120 45 10  1

Another Game (y/Y/n/N)? n

Game Over.
Thank you for playing.
```

Note 1, in order to see the output for larger row values, you may need to expand the width of the console window.

Note 2, larger row values will take longer execution times, especially for rows > 20.

Submission

- All source files must assemble and execute with QtSpim/SPIM MIPS simulator.
- Submit source files
 - 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.
- 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 loss of up to 3%.

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.
Correct Sort Algorithm	-	Failure to use the provided sort algorithm 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.