# CS 400 HW 2: Pascal Triangle

## General Description

This homework question has two folds:

1. Create a vector filled with several rows from Pascal Triangle (<https://en.wikipedia.org/wiki/Pascal%27s_triangle>), where each row is represented by an int vector. Also print the first 8 rows of the triangle.
2. Check summations of each row against a given int vector, which contains alleged summation values. Print your check results.

## Requirement

1. Design a function that returns Pascal Triangle vector. The return type should be std::vector<std::vector<int>> type. The function takes an integer parameter that indicates number of rows to be created and returned. The parameter should be within the range 1 – 16 (inclusive). The function should return an empty vector for any values not in the range.
2. Design another function that takes two vectors as parameters and returns a vector of Boolean type. Function prototype: std::vector<bool> compare(const std::vector<std::vector<int>>& triangle, const std::vector<int> allegedSummations). The first parameter is a Pascal Triangle with proper number of rows (please figure out how many rows are needed), and the second parameter has alleged summation values of each row from a Pascal Triangle. The function will thus compare (calculated) real summations from the first vector against alleged values from the second vector and returns a Boolean vector as comparison results.
3. For requirement 2): you need to use standard accumulate function (<https://cplusplus.com/reference/numeric/accumulate/>) and vector iterators. Do not write your own accumulate logic.
4. (Optional) When printing a Pascal Triangle, please use blank spaces to align numbers.

## Start File

The start file (cs400\_hw2\_wsuID.cpp) contains the main() function that provides some test cases for your functions. You may choose to use them or develop your own.

## Submission

Filename format: cs400\_hw2\_{your WSU ID}.cpp

Comment: please have succinct comments in your source code.

Upload your solution file to the Blackboard.