**Due date**
Wednesday, May 7, 2014

**Delivery method**
This is a multiple file, two-part program that must be archived using the **tar** command. Deliver a single file named pa6.tar (all lowercase) as an **attachment** to an email that you send to the class account csc3320@orion.ucdenver.edu Put PA6 in the subject field, and your name in the body of the email.

**Program objectives**
An ability to use current techniques, skills, and tools necessary for computing practice (ABET i).

**Value**
This program is worth 17 points. The distribution of points will be as follows.

| Criterion | Value |
|---|---|
| C++ 11 | 3 |
| bin_coeff class | 3 |
| set class | 3 |
| Program style | 2 |
| Correct output with annotation | 6 |

**PART I**
**Background**
The *Binomial Theorem* can be used to compute the expansion of a binomial expression raised to an integer power. For example, given the expression $(a + b)^n$, we know its expansion is of the form $(a + b)\ (a + b)\ (a + b)\ldots.\ (a + b)$, where there are $n$ product terms.

However, the *Binomial Theorem* tells us that this product can also be computed as follows.

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k \text{, where } \binom{n}{k} = \frac{n!}{k!(n-k)!}\text{, and } n! = \prod_{j=1}^{n} j = n(n-1)(n-2)....2\times 1$$

The expression $\binom{n}{k}$ is called a *binomial coefficient* because it computes the coefficients of the terms in the expansion of $(a + b)^n$.

**Problem**
Write a class called bin_coeff that represents a binomial coefficient, then use this class to help compute and display the expansion of a binomial expression $(a + b)$ to an integer power $n$.

**Input**
An integer, $n$, that represents the exponent in the expression $(a + b)^n$. The maximum value of $n$ will be 10. This value will be input from the command line in argv[2].

**Output**
An $m$-tuple, where $m=n+1$, that represents all the binomial coefficients in the expansion of $(a + b)^n$ (See below for an output sample).

**Output sample**
Suppose you are expanding $(a + b)^3$. Then, the terms of the expansion are as follows.

$$(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$$

The 4-tuple that represents the coefficients is (1, 3, 3, 1).

**PART II**
**Problem**
Create a class called **set** which is a data structure that does not allow duplicates. Use this data structure to perform several set operations, including union, intersection, and difference. Test your set operations on data read from files as provided by the grader.

**Input**
Space delimited random-valued integers stored in a file. There will be 3 rows of space-delimited integers that represent the elements of 3 sets ($A$, $B$, and $C$). The file name will be in argv[1].

**Output**
The sets resulting from various combinations of the set operations, union, intersection and difference.

**Class set**
In addition to any members that you deem necessary to implement a set, you must have the following

1. Non-member friend operations of union ($\cup$), intersection ($\cap$) and difference (-) defined as follows for two sets, $A$ and $B$.
    a. $A \cup B = \{x \mid x \in A \vee x \in B\}$
    b. $A \cap B = \{x \mid x \in A \wedge x \in B\}$
    c. $A - B = \{x \mid x \in A \wedge x \notin B\}$

    Notice that these operations do not modify the operand sets.
2. Non-member friend operator ==, which is true if and only if the two sets being compared have the same elements.
3. Non-member friend operator << to display the contents of a set.

**Driver**
1. Read the name of the data file from the command line in argv[1].

2. Open the file and read the data into 3 sets $A$, $B$, and $C$.
3. Display the contents of sets $A$, $B$, and $C$ with an appropriate annotation.
4. Perform the following operations and display the results with appropriate annotations.
    a. $A \cap (B \cup C)$
    b. $(A \cup B) - C$
    c. $(A \cap B) \cup (B \cap C)$
    d. $(A - B) \cup (C - A)$
    e. $(A \cap B) == (B - C)$

**Notes**

You must use the following C++ 11 constructs and types

1. Range-based **for** loop.
2. std::tuple.
3. At least one lambda expression.

After finishing Part I, pause the screen with a prompt to enter a key to continue, then finish the program with Part II.