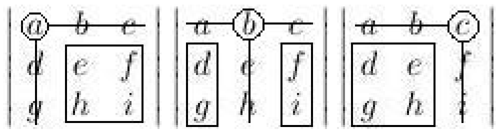
COSC 1437 (DL) - Fall 2016
Program Set #2

See 1437 Grading /Program Guide Sheet for directions and grading/submission information.

1. In mathematics, if you have a square matrix then its determinant can be found. A 2x2 matrix determinant can be found by:

$$\det(A) = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

Determinants of any matrix of size m × m, with m >2, are calculated by the following process. Start with the top-left corner and move across the top row. For each element, cross out the elements in the matrix in the same column and row and join the remaining elements together, as in



The remaining elements form a matrix of size m-1 × m-1. Multiply the determinant of this smaller matrix with the element in the top row of the original matrix. The determinant of the original matrix is formed by adding and subtracting these products, alternating operations for each term and beginning with subtraction between the first two terms. Thus, the determinant of a 3 × 3 matrix can be calculated as follows:

$$\begin{aligned}\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} &= a\begin{vmatrix} e & f \\ h & i \end{vmatrix} - b\begin{vmatrix} d & f \\ g & i \end{vmatrix} + c\begin{vmatrix} d & e \\ g & h \end{vmatrix} \\ &= (aei - afh) - (bdi - bfg) + (cdh - ceg)\end{aligned}$$

Likewise, the determinant of a 4 × 4 matrix can be calculated as follows:

$$\begin{vmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{vmatrix} = a\begin{vmatrix} f & g & h \\ j & k & l \\ n & o & p \end{vmatrix} - b\begin{vmatrix} e & g & h \\ i & k & l \\ m & o & p \end{vmatrix} + c\begin{vmatrix} e & f & h \\ i & j & l \\ m & n & p \end{vmatrix} - d\begin{vmatrix} e & f & g \\ i & j & k \\ m & n & o \end{vmatrix}$$

Determinants of larger matrices are calculated in a similar manner. Notice that if any of the elements on the top row is 0, then the associated determinant does not need to be calculated—0 times any number is still 0.

Write a C++ program that will find the determinant of square matrices from sizes 2-10. Let the user input the matrix size and randomly generate the elements in the matrix using values from -100 to 100. Error check. Output the matrix and the value of the determinant. Use a least one user-defined function in your program. Output should look similar to below.

Sample Runs:

```
Enter the size of the square matrix: 3
Here is the matrix:

1 2 3
2 3 4
5 6 5

The value of the determinant is: 2


Enter the size of the square matrix: 4
Here is the matrix:

1 2 3 4
2 3 4 4
5 6 5 2
8 9 9 4

The value of the determinant is: 2
```

Name the program: `MatrixDetermineXX.cpp`, where XX are your initials.

2. In this problem, write a C++ program that process some information base on a data file which contains the 2006 breed rankings based on the number of dog registrations, as well as the 2005 rankings, based on numbers from the American Kennel Club (AKC). The first part of the input file is shown here:

```
#Data obtained from AKC.org
#This file contains the 2006 and 2005 breed rank based on number of registrations.
#2006 Rank,Breed,2005 Rank
1,Retrievers (Labrador),1
2,Yorkshire Terriers,3
3,German Shepherd Dogs,4
4,Retrievers (Golden),2
5,Beagles,5
6,Dachshunds,6
7,Boxers,7
8,Poodles,8
9,Shih Tzu,9
10,Miniature Schnauzers,10
11,Chihuahuas,11
12,Bulldogs,13
```

The number one ranked dog in the AKC in 2006 was the "Retrievers (Labrador)", its ranking in 2005 was also first; whereas, the "Yorkshire Terrier" was 2nd in 2006 and 3rd in 2005. Output the total number of breeds in the file. Also, report the dog breed and rank number for those breeds that had the same rank in 2006 and 2005. Finally, show the total same-ranked dog breeds (2005 & 2006) at the end with a percentage as well. Let the user enter the file name from the keyboard. The data file (`.csv`) will be provided by your instructor. Output should look similar to below.

Sample Run:

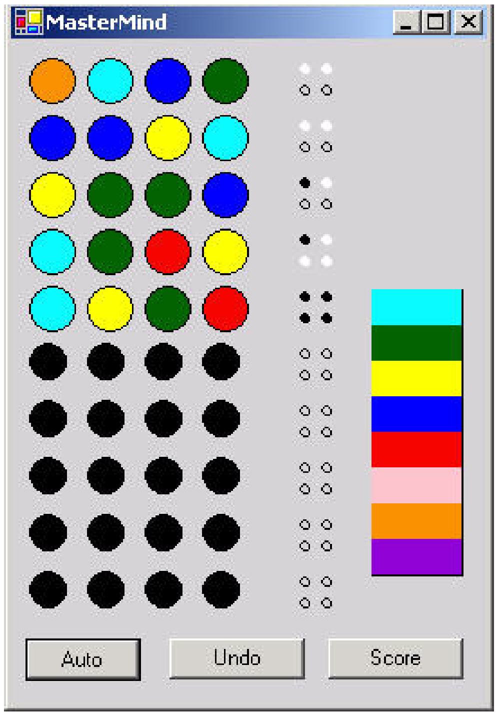
```
Enter the file name: DogRanking.csv

Total AKC Breeds: 155
AKC Breeds with the same ranking in 2006, and 2005 are:
1    Retrievers (Labrador)
5    Beagles
6    Dachshunds
7    Boxers
8    Poodles
9    Shih Tzu
10    Miniature Schnauzers
11    Chihuahuas
14    Pomeranians
21    Doberman Pinschers
24    Great Danes
25    Siberian Huskies
34    Australian Shepherds
35    Papillons
54    Rhodesian Ridgebacks
64    Chow Chows
73    Pointers (German Wirehaired)
74    Spaniels (English Cocker)
79    Welsh Corgis (Cardigan)
99    Standard Schnauzers
124    Spaniels (Welsh Springer)
125    Affenpinschers
131    Black and Tan Coonhounds
151    Foxhounds (American)
152    Otterhounds

16.13% of the 2006 breeds have the same ranking in 2005 (25 breeds)
```

Name the program: `DogShowStatsXX.cpp`, where XX are your initials.

3. In the game Mastermind, one player creates a secret code of four pegs, each of which can be chosen from one of six colors. The other player then has to guess the color of each peg, with the order mattering. The player who made the secret code then has to give feed back to the player guessing. This feedback is in the form of white and black pegs. A black peg means that the guesser has chosen the correct color in the correct slot. These are first "calculated " and awarded. Once these are counted, these pegs are ignored. Then the white pegs are awarded. These are for pegs that are the correct color but are in the incorrect slot. There is no "double dipping" of pegs in the response, so the sum total of white and black pegs the guesser can receive is four, and no one peg in the guesser's answer may earn them more than one black or one white peg.



Write a C++ program to implement a simplified version of the game where the colored pegs are replaced with a sequence of four random numbers (master sequence) that have possible values between 0 and 5. There are two program options. The first option is the demo mode- a demo mode is required so the program can be graded. In demo mode, the user enters the master sequence. The user will always enter valid combinations, meaning they will always enter exactly four integers in between 0 and 5, inclusive, separated by spaces. This will allow you to test your logic, results, etc. The second option is game mode. In this mode, the program will randomly generate the master sequence and the user can guess at the sequence. In either mode, the four numbers will then be compared to the computer's master sequence and gives feedback to the player after each guess. Output the total number of tries it took to guess the correct sequence. If after ten tries, the user fails to guess the correct sequence stop the program and tell the user the correct value.

From the master sequence entered the program will respond to each guess with a statement of the following form:

```
    You have X perfect matches and Y imperfect matches.
```

where X is the number of perfect matches (right number, right spot) and Y is the number of imperfect matches (right number, wrong spot with no double counting). At the end of the game, if the user wins, output the number of guesses they needed printed in the following format:

```
    You have won the game in X turns
```

where X is the number of turns taken. If the user loses (after 10 ten turns), then output a message stating so along with a printout of the correct game master sequence given using the following form:

```
       Sorry, you have exceeded the maximum number of turns. You lose.
       Here is the winning board: A B C D
```

where A, B, C and D are the numbers (in order) of the secret pegs (master sequence). Finally, prompt the user if they wish to run the program again. Output should look similar to below.

Sample Run:

```
Welcome to MasterMind!

There are two modes for this program: demo and game.
Which do you want to use? Enter D-demo mode or G-game mode
Please enter your choice: d

Enter the master sequence: 5 5 5 3

You have 10 chances, good luck!

Guess 1: 0 1 2 3
You have 1 perfect matches and 0 imperfect matches.

Guess 2: 4 4 4 4
You have 0 perfect matches and 0 imperfect matches.

Guess 3: 5 5 5 3
You have 4 perfect matches and 0 imperfect matches.

You have won the game in 3 turns

Do you wish to play again (Y/N): n

Thanks for playing
```

Name the program: `MasterMindXX.cpp`, where XX are your initials.

4 (**). Write a C++ program that scores a game of bowling. A game of bowling has 10 frames. The tenth frame works in a slightly different way than the first 9. For the first 9 frames, each will consist of 1 or 2 rolls. You will be presented with 10 pins. If you hit all 10 pins in the first roll you make a "strike", and the frame is over. Otherwise you have a second roll to hit the remaining pins. If you hit all remaining pins you score a "spare";otherwise you have an "open frame". The number of points an open frame gives is simply the total number of pins you hit. A spare gives 10 points plus the number of pins hit on your next roll (in the next frame). A strike gives 10 points plus the number of pins hit on the next two rolls (in thenext frame, or possibly the next two frames if you hit a strike in the nextroll). The last frame consists of 2 or 3 rolls. The way it works is as follows:

- 10 pins are presented
- If you hit all 10 pins, you score a strike (10 points), and 10 pins are presented again:
    - If you hit all 10 pins again, you score a new strike, and 10 pins are presented again, for the last time.
        - If you hit 10 pins again, you score one more strike.
        - If you hit fewer than 10 pins, you score that number of pins.
    - If you hit fewer than 10 pins, you score that number of points and the remaining pins are presented.
        - If you hit all remaining pins, you score a spare (10 points).
        - If you hit fewer than that, you score that many pins.
- If you hit fewer than 10 pins on the first roll, the remaining pins are presented.
    - If you hit all remaining pins, you score a spare and 10 pins are presented again.
        - If you hit all 10 pins on the third roll, you score a strike.
        - If you hit fewer than 10 pins, you score as many points as pins hit.
    - If you hit fewer than the remaining pins, you score the number of pins hit in both rolls.

Note that the number of points in the last frame is the total number of pins hit in all rolls.

Input will be from the keyboard containing the rolls for a single game, concatenated into a single string. Each character will be one of the following: `0, 1, 2, 3, 4, 5, 6, 7, 8, 9, /`, or `X`, where the digits represent the number of pins knocked down, the `'/ '` indicates a spare, and the `'X'` represents a strike. Note that the frames are not delimited in any way-it's your job to figure out if a frame has one, two, or three rolls. Assume valid input. Output the frame/total game score using the format shown below. Finally, ask the use if he/she wishes to run the program again.

Sample Runs:

```
Enter bowling scores for a game: 81XX3/451/XX3/4/8

Frames
 1      2      3       4       5      6      7      8       9      10
-------------------------------------------------------------------

8-1    X      X      3 /    4-5    1 /   X      X      3 /    4/8
9      32     52     66     75     95    118    138    152    170

Total Score: 170


Run again(Y/N): y
```

```
Enter bowling scores for a game: XXXXXXXXXX9/

Frames
 1      2      3       4       5      6      7      8       9      10
-------------------------------------------------------------------

X      X      X      X      X      X     X      X      X      X9/
30     60     90     120    150    180   210    240    269    289

Total Score: 289


Run again(Y/N): Y


Enter bowling scores for a game: 101/22X33X1/3/X12

Frames
 1      2      3       4       5      6      7      8       9      10
-------------------------------------------------------------------

1-0    1/     2-2    X      3-3    X     1/     3/     X      12
1      13     17     33     39     59    72     92     105    108


Total Score: 108


Run again(Y/N): n
```

Name the program: `BowlingXX.cpp`, where XX are your initials.