BCS230
Lab6 Handout and Assignment

This lab handout provides three tasks that students are required to attempt during lab session.

Lab objectives are for students to practice with:

- Files
- Arrays.
- Classes and Objects.
- Object-Oriented Analysis and Design.

Task1:

Attached on Blackboard:

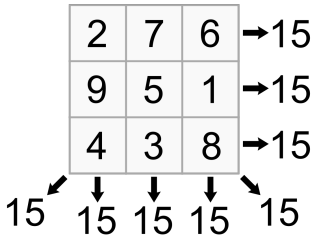
1- inputFile.txt

- Write a program that reads the numbers from the attached file and place them in an array.
- Print all the elements of the array using regular for-loop.
- Create a text file called output.txt and write to it the average, minimum and maximum values separated by tab space.

Task2:

Magic Squares

One interesting application of two-dimensional arrays is magic squares. A magic square is a square matrix in which the sum of every row, every column, and both diagonals is the same. Magic squares have been studied for many years, and there are some particularly famous magic squares.



In this exercise you will modify the provided shell code to determine whether a square is magic or not assuming that all elements are between 1 and 9.

Your program should simulate a magic square using two-dimensional 3x3 array. It should have a Boolean function isMagicSquare that accepts the array as an argument and returns true if it is determines it is a Magic square and false it not as shown in the shell code bellow.

```
#include <iostream>
using namespace std;

const int SIZE = 3;  // Arrays will be 2-D 3x3 arrays
// Function prototype
void inputMagicSq(int[][SIZE],int);
void showArray (int[][SIZE]);
bool isMagicSquare(int[][SIZE]);

int main()
{
   int array1[SIZE][SIZE];
   cout << "Enter square elements 3x3. \n\n";
   //ToDo 1: call the inputMagicSq to enter values
   cout << "You entered. \n\n";
   //ToDo 2:  call showArray to print elements
   //ToDO 3: check if the square is magic or not
   return 0;
}

void showArray (int A[][SIZE])
{
   // ToDo: print the square elements
}

void inputMagicSq(int sqArr[][SIZE], int size){
   //ToDo: read square elements
}

bool isMagicSquare(int A[][SIZE])
{
   int sum1 = 0, sum2 = 0, sum3 = 0;

   // ToDo: Check column sums
   for (int row = 0; row < SIZE; row++)
   {
     // ToDo: Check each column seperately
   }
   if (!(sum1 == sum2 && sum2 == sum3))
      return false;

   // If column sums are OK, check row sums
   // But first reinitialize the accumulators
   sum1 = sum2 = sum3 = 0;

   for (int col = 0; col < SIZE; col++)
   {
     // ToDo: Check each row seperately
   }
   if (!(sum1 == sum2 && sum2 == sum3))
      return false;

   // If row sums are also OK, compute the 2 diagonal sums
   // Let sum3 continue to hold the value the diagonal sums
should equal
   sum1 = A[0][0] + A[1][1] + A[2][2];  // Right diagonal
   sum2 = A[0][2] + A[1][1] + A[2][0];  // Left diagonal
   if (!(sum1 == sum2 && sum2 == sum3))
      return false;

   // If we got this far. Everthing adds up to the same value.
   return true;
}
```

Task3 (based on page 595):

Download from Blackboard the following two files:

1) Teams.txt – this files contains an alphabetical list of number of Major League baseball teams that have won the World Series at least once.
2) WorldSeriesWinners.txt – this file contains a chronological list of World Series winning teams from 1950 through 2014. The first line in the file is the name of the team that won 1950, and the last line is the name of the team that won in 2014.

   (Note that the World Series was not played in 1994.)

```
Anaheim Angels
Arizona Diamondbacks
Atlanta Braves
Baltimore Orioles
Boston Americans
Boston Braves
Boston Red Sox
Brooklyn Dodgers
Chicago Cubs
Chicago White Sox
Cincinnati Reds
Cleveland Indians
Detroit Tigers
Florida Marlins
Kansas City Royals
Los Angeles Dodgers
Milwaukee Braves
Minnesota Twins
New York Giants
New York Mets
New York Yankees
Oakland Athletics
Philadelphia Athletics
Philadelphia Phillies
Pittsburgh Pirates
San Francisco Giants
St. Louis Cardinals
Toronto Blue Jays
Washington Senators
```

Figure 1: Teams.txt

Step1:

- Modify the skeleton code of Task1 to count how many different teams that won the World Series.

Step2:

- Read the contents of the files into two arrays.
- Check how many different teams won the Series from date1 to date2. (i.e: 1995 to 2005)
- Ask the user to enter a team name and then check if the team has won before or not.
  - If 'Yes', then show additional options to display how many times and/or in which year.

```
New York Yankees
New York Yankees
New York Yankees
New York Yankees
New York Giants
Brooklyn Dodgers
New York Yankees
Milwaukee Braves
New York Yankees
Los Angeles Dodgers
Pittsburgh Pirates
New York Yankees
New York Yankees
Los Angeles Dodgers
St. Louis Cardinals
Los Angeles Dodgers
Baltimore Orioles
St. Louis Cardinals
Detroit Tigers
New York Mets
Baltimore Orioles
Pittsburgh Pirates
Oakland Athletics
Oakland Athletics
Oakland Athletics
Cincinnati Reds
Cincinnati Reds
```

Figure 2: Sample of WorldSeriesWinners.txt

Lab6 – Assignment.

What to hand-in

Problem)

1) Design a PayRoll class that has data members for an employee's hourly pay rate and number of hours worked.
2) Write a program with an array of eleven PayRoll objects.
3) The program should read the number of hours each employee worked and their hourly pay rate (before tax) from a file and call class function to store this information in the appropriate objects.
4) It should then call class function, once for each object, to return the employee's net pay after 20% tax deduction, so this information can be displayed.
    a. Sample data to test this program can be found in the ***payroll.dat*** file located on Blackboard.

Submission instructions

- Test your code before submission.
- Add comments to explain your solution.
- Due date is Friday 03/16/2018 @ 11:59pm/
- You are allowed to submit late with -2% per day.
- Submission will not be available after Sunday March /18th/2018 @ 11:59pm.
- Check rubric.