Assignment 6 worksheet

FINAL PROGRAMMING PROJECT

This ASSIGNMENT contains the following activities:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Activity 6.1 | Choose Your Final Programming Project – Part **1** / 4  See your syllabus (Final Project - 4 parts) or the schedule below for details:   |  |  |  | | --- | --- | --- | | **Week** | **Assignment This Week** | **Due Date** | | **10** | ***Final Project – Part 1 / 4 (Choose 1 and begin)*** | ***10/29*** | | 12 | *Final Project – Part* ***2*** */ 4 (Submit Progress Report)* | *11/12* | | 15 | *Final Project – Part* ***3*** */ 4 (Submit Pseudocode)* | *12/3* | | 17 | *Final Project – Part* ***4*** */ 4 (Submit Source code file)* | *12/16* | |

|  |  |
| --- | --- |
| Activity 6.1 | Choose your Final Programming Project |
| Overview | In Activity 6.1, you choose your Final Programming Project. |

1. Review the “Appendix” at the end of this document and choose one out of nine projects:

1. Lo Shu Magic Square

2. Lottery Application

3. World Series Champions

4. Name Search

5. Tic‐Tac‐Toe Game

6. String Search

7. File Cipher

8. Average Number of Words

9. Other

1. Write down the Final Programming Project of your choice (1 through 9) below:

**<< WRITE YOUR “FINAL PROGRAMMING PROJECT” OVER THIS TEXT >>**

**MAGIC SQUARE**

**Note:**

*Once you have filled in the required information, save the file to your flash drive / hard disk. Then, you can submit it to your instructor through your Blackboard Course for review and grading.*

**Appendix**

**1. Lo Shu Magic Square**

The Lo Shu Magic Square is a grid with 3 rows and 3 columns shown in Figure 1. The Lo Shu Magic Square has the following properties:

* The grid contains the numbers 1 through 9 exactly.
* The sum of each row, each column, and each diagonal all add up to the same number. This is shown in Figure 2.

In a program you can simulate a magic square using a two-dimensional array. Write a

method that accepts a two-dimensional array as an argument, and determines whether the

array is a Lo Shu Magic Square. Test the method in a program.



Figure 1 Figure 2

**2. Lottery Application**

Write a program that simulates a lottery. The program should have an array of five integers named lottery and should generate a random number in the range of 0 through 9 for each element in the array. The user should enter five digits, which should be stored in an integer array named user. The program is to compare the corresponding elements in the two arrays and keep a count of the digits that match. For example, the following shows the lottery array and the user array with sample numbers stored in each. There are two matching digits (elements 2 and 4).

lottery array:



user array:



The program should display the random numbers stored in the lottery array and the number of matching digits. If all of the digits match, display a message proclaiming the user as a grand prize winner.

**3. World Series Champions**

You will find the following files in this BlackBoard Module's folder:

* *Teams.txt*-This file contains a list of several Major League baseball teams in alphabetical order. Each team listed in the file has won the World Series at least once.
* *WorldSeriesWinners.txt*- This file contains a chronological list of the World Series' winning teams from 1903 to 2012. (The first line in the file is the name of the team that won in 1903, and the last line is the name of the team that won in 2012. Note that the World Series was not played in 1904 or 1994.)

Write a program that displays the contents of the Teams.txt file on the screen and prompts the user to enter the name of one of the teams. The program should then display the number of times that team has won the World Series in the time period from 1903 to 2012.

**TIP:** Read the contents of the WorldSeriesWinner.txt file into an array (or any suitable data structure). When the user enters the name of a team, the program should step through the array counting the number of times the selected team appears.

**4. Name Search**

You will find the following files in this BlackBoard Module's folder:

* GirlNames.txt-This file contains a list of the 200 most popular names given to girls born in the United States from 2000 to 2009.
* BoyNames.txt-This file contains a list of the 200 most popular names given to boys born in the United States from 2000 to 2009.

Write a program that reads the contents of the two files into two separate arrays or vectors. The user should be able to enter a boy's name, a girl's name, or both, and the application should display messages indicating whether the names were among the most popular.

**5. Tic‐Tac‐Toe Game**

Write a program that allows two players to play a game of tic-tac-toe. Use a two-dimensional char array with three rows and three column s as the game board. Each element of the array should be initialized with an asterisk (\*). The program should run a

loop that

* Displays the contents of the board array
* Allows player 1 to select a location on the board for an X. The program should ask the user to enter the row and column number.
* Allows player 2 to select a location on the board for an O. The program should ask the user to enter the row and column number.
* Determines whether a player has won, or a tie has occurred. If a player has won, the program should declare that player the winner and end. If a tie has occurred, the program should say so and end.

Player 1 wins when there are three Xs in a row on the game board. The Xs can appear in a row, in a column, or diagonally across the board. A tie occurs when all of the locations

on the board are full, but there is no winner.

**6. String Search**

Write a program that asks the user for a file name and a string to search for. The program should search the file for every occurrence of a specified string. When the string is found, the line that contains it should be displayed. After all the occurrences have been located, the program should report the number of times the string appeared in the file.

**N OTE:** Using an editor, you should create a simple text file that can be used

to test this program.

**7. File Cipher**

File encryption is the science of writing the contents of a file in a secret code.

Although there are complex encryption techniques, you should come up with a simple one of your own. For example, you could read the first file one character at a time, and add 10 to the ASCII code of each character before it is written to the second file.

Your ‘encryption’ program should work like a filter, reading the contents of one file,

modifying the data into a code, and then writing the coded contents out to a second file. The second file will be a version of the first file, but written in a secret code.

Write a second program that decrypts the file produced by the program. The decryption program should read the contents of the coded file, restore the data to its original state, and write it to another file.

**8. Average Number of Words**

The text in the file named *text .txt* is stored as one sentence per line. Write a program

that reads the file's contents and calculates the average number of words per sentence.

**9. Other**

Describe your final programming project here:

Your final programming project must connect the multiple topics, we have/are learned/learning, such as:

* Textbook Chapters 3-8, 10-12 & 14, including: Decision Structures; Loops and Files; Methods; A First Look At Classes; Arrays and the ArrayList Class; A Second Look at Classes and Objects; Inheritance; Exceptions and Advanced File I/O; A First Look at GUI Applications; Applets and More.
* Textbook theory and Programming Challenges (Review source code solutions in BlackBoard Learning Modules)
* MPL Programming Projects: Chapters 3-8, 10-12 & 14.