CSC 2000
Introduction to C++ Programming Language
Winter Term 2020
Project 02
50 points
Due 04/13/2020 (11:45 A.M.)

The objectives of this Project are:

1. Being able to deal with **Functions**
2. Being able to work with **1-D** & **2-D Arrays**
3. Understand the principle of **abstraction** and **encapsulation** and use them to implement classes.
4. Being able to Analyze, Design, implement, and test a practical real-world application

Requirements:

In a word file:

- Analyze each problem; outline the problem and its solution requirements. (Describe the problem including input and output in your own words.))
- Design an algorithm to solve each problem. (Describe the major steps for solving the problem.)
- Using visual Studio C++ 2019 software, implement the algorithm in C++.
- Test the code for each problem and verify that the algorithm works; include a screenshot for each program's output.

Restrictions:

You must work individually. Use only material from class or from the text book (chapters 1- 10). All code must be the work of the individual. Do not share your code or copy from external resources.

Submission

Submit two files for each problem, one-word file (for analysis, design, and screenshot) and the source file. Extra Credit question must contain three files: a header file (.h), an implementation file (.cpp), and a driver file (.cpp).
Upload all file to Canvas by the due date. DO NOT Email your files.

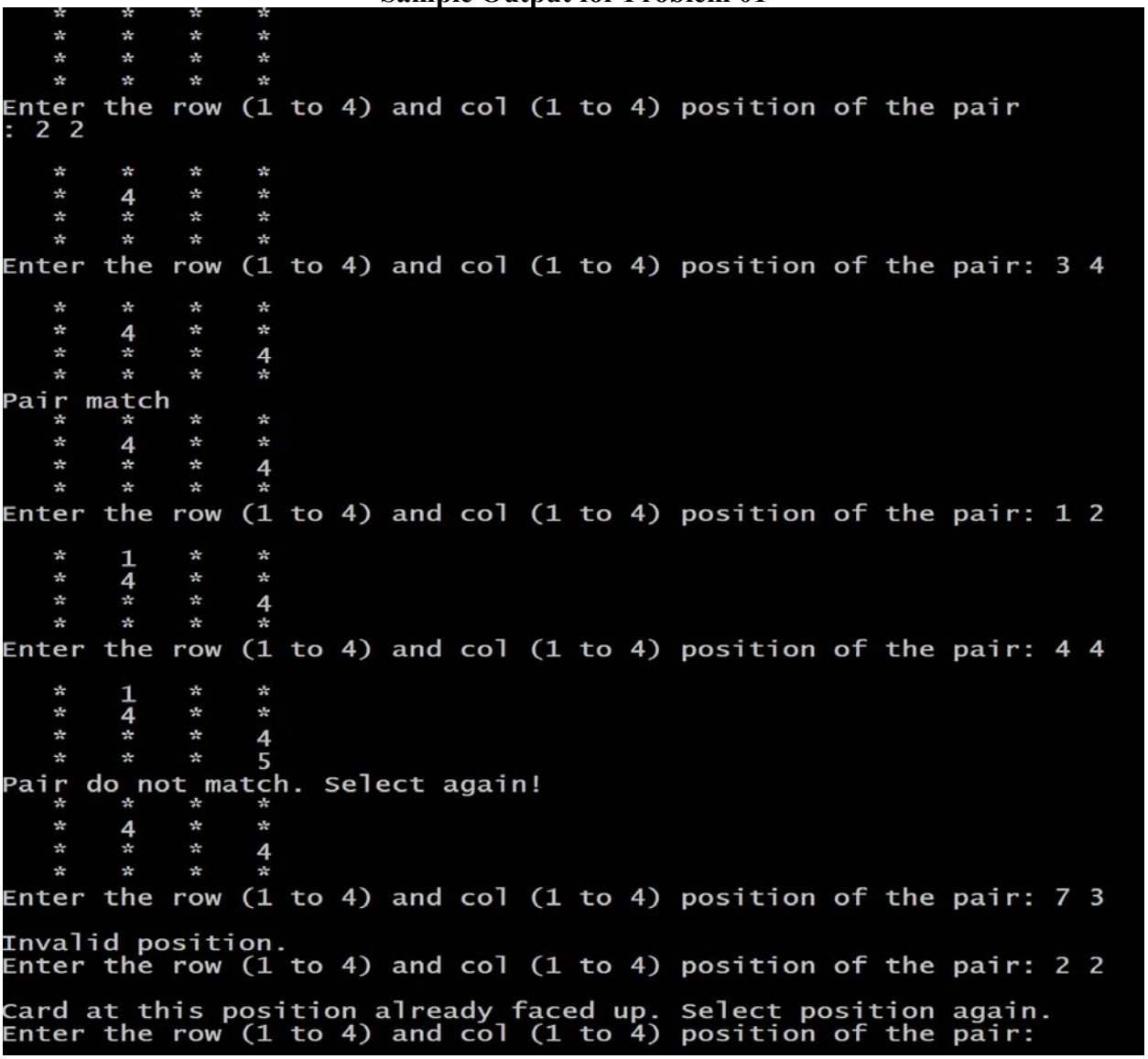
Grading Criteria:

The grade of each program will be based on the creation of a program that works correctly, up to some details (35%), clear problem analysis and algorithm design (2.5%), the appropriate use of classes, functions, and arrays (40%), the production of clear output, with readable formatting and without unnecessary repetition (15%), composition of informative comments (2.5%), and testing the program with different inputs (5%). Programs must compile.

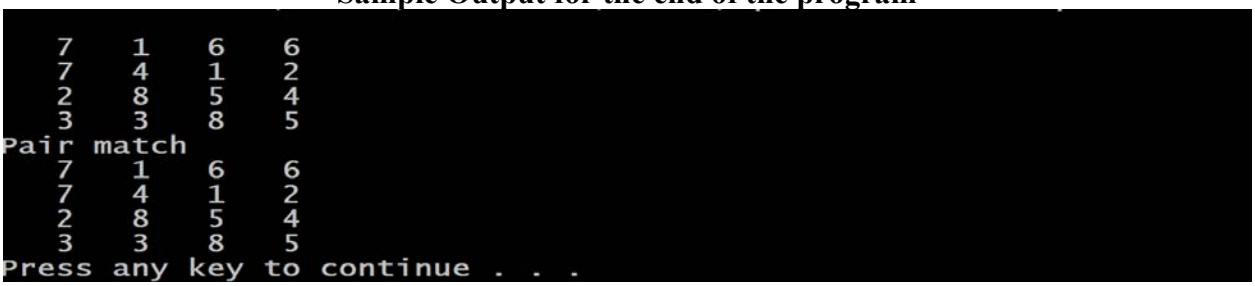
Problem 01: Memory Game (30 points)

Children often play a memory game in which a deck of cards containing matching pairs is used. The cards are shuffled and placed face down on a table. The players then take turns and select two cards at a time. If both cards match, they are left face up; otherwise, the cards are placed face down at the same positions. Once the players see the selected pair of cards and if the cards do not match, then they can memorize the cards and use their memory to select the next pair of cards. The game continues until all the cards are face up. Write a program to play the memory game. Use a two-dimensional array of 4 rows and 4 columns for a deck of 16 cards with 8 matching pairs. You can use numbers 1 to 8 to mark the cards. Use random number generators to randomly store the pairs in the array. Use appropriate functions in your program, and the main program should be merely a call to functions.

Sample Output for Problem 01



Sample Output for the end of the program



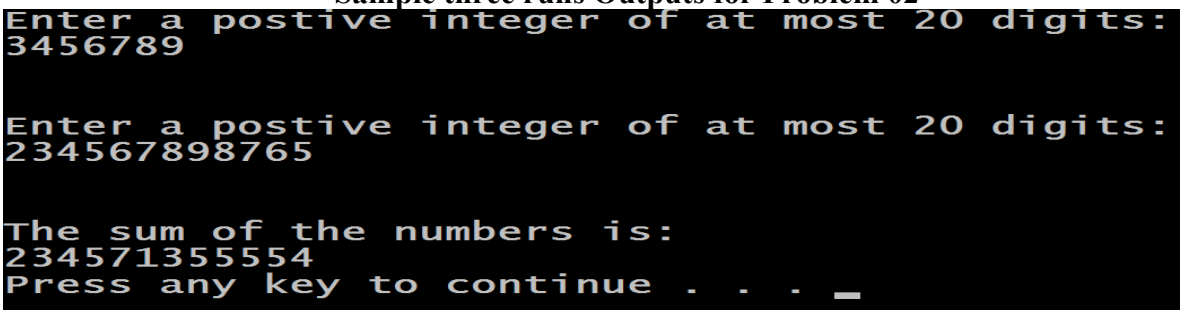
Problem 01 Grading Criteria:

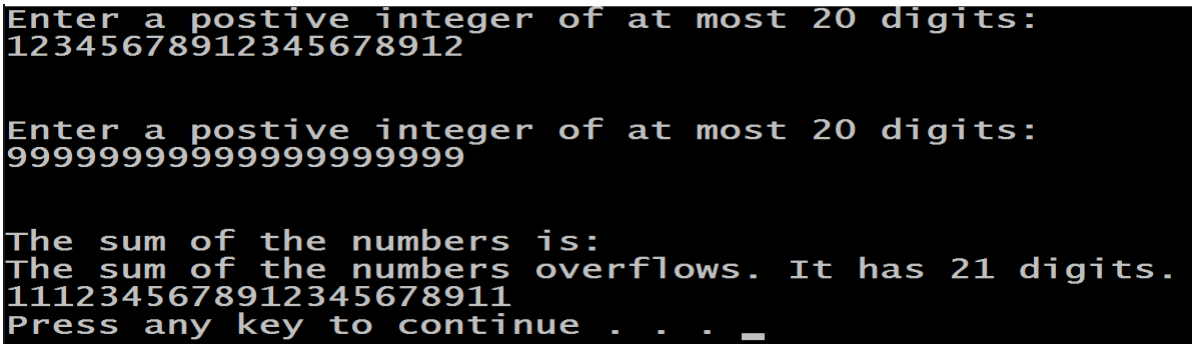
- *8 points - a function to play the gam*
- *8 points - a function to fill the board correctly.*
- *8 points - a function to get the card location correctly.*
- *6 points - a function to show the Board.*
- *5 points will be deducted if you do not generate the cards randomly.*
- *5 points will be deducted if you do not check for card invalid position.*
- *5 points will be deducted if you ignore card already faced up case.*
- *5 points will be deducted if you do not use 2D array correctly.*
- *5 points will be deducted if you program does not terminate after you complete the game successfully.*
- *2.5 points will be deducted if you do not include at least one paragraph for analyzing the problem and one paragraph for designing a solution.*
- *2.5 points will be deducted if you do not include comments within your code.*

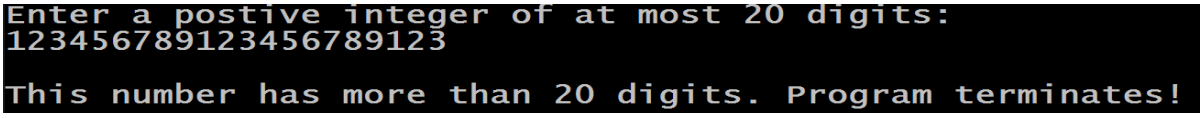
Problem 02: Large Integer (20 points)

In C++, the largest int value is 2147483647. So, an integer larger than this cannot be stored and processed as an integer. Similarly, if the sum or product of two positive integers is greater than 2147483647, the result will be incorrect. One way to store and manipulate large integers is to store each individual digit of the number in an array. Write a program that inputs two positive integers of, at most, 20 digits and outputs the sum of the numbers. If the sum of the numbers has more than 20 digits, output the sum with an appropriate message. Your program must, at least, contain a **function** to read and store a number into an array and another **function** to output the sum of the numbers. (Hint: Read numbers as strings and store the digits of the number in the reverse order.)

Sample three runs Outputs for Problem 02







Problem 02 Grading Criteria:

- *5 points - a function to read the number.*
- *15 points - a function to sum the two numbers.*
- *5 points will be deducted if you handle the overflow case.*
- *5 points will be deducted if you do not handle more than 20 digits case.*
- *2.5 points will be deducted if you do not include at least one paragraph for analyzing the problem and one paragraph for designing a solution.*
- *2.5 points will be deducted if you do not include comments within your code.*

Extra Credit Problem - Rational Class (25 points)

Create a class called Rational for performing arithmetic with fractions.
Write a program to test your class.
Use integer variables to represent the private data of the class—the numerator and the denominator.
Provide a constructor that enables an object of this class to be initialized when it's declared. The constructor should contain default values in case no initializers are provided and should store the fraction in reduced form. For example, the fraction would be stored in the object as 1 in the numerator and 2 in the denominator. Provide public member functions that perform each of the following tasks:

a) Adding two Rational numbers. The result should be stored in reduced form.
b) Subtracting two Rational numbers. The result should be stored in reduced form.
c) Multiplying two Rational numbers. The result should be stored in reduced form.
d) Dividing two Rational numbers. The result should be stored in reduced form.
e) Printing Rational numbers in the form a/b, where a is the numerator and b is the denominator.
f) Printing Rational numbers in floating-point format.

Create an object of class Rational in the main function and test all functions.

$$Sum\ of\ two\ fractions: \frac{a}{b} + \frac{c}{d} = \frac{ad + cb}{bd}$$

$$Difference\ of\ two\ fractions: \frac{a}{b} - \frac{c}{d} = \frac{ad - cb}{bd}$$

$$Product\ of\ two\ fractions: \frac{a}{b} \times \frac{c}{d} = \frac{ac}{bd}$$

$$Division\ of\ two\ fractions: \frac{\left(\frac{a}{b}\right)}{\left(\frac{c}{d}\right)} = \frac{ad}{bc}$$

Sample Output: