Computer Science 1 - Program 1
Lottery
Assigned: 1/14/04
Due: 1/23/04 (Friday) by 11:55pm (WebCT time)

Objective

1. To review reading from a file.
2. To use records.
3. To create an array dynamically.
4. To use enumerated types in an useful manner.

The Problem

You are hired by the state of Florida to decide who gets winnings for the Florida Lottery. The state has a system that automatically creates a file that contains the name of each ticket purchaser along with the combination of 6 ***distinct*** numbers picked by that person. Your job is to use this file in determining everyone's winnings. In particular, here is the payout for matching a certain number of values:

Numbers Matched	Winnings
3	$10
4	$100
5	$10000
6	$1000000

Your program will ask the user for an input file with the data for the ticket purchases. Then your program will ask the user for the winning combination of numbers. (When playing the Florida lottery, you must pick 6 ***distinct*** numbers from the set {1,2,3,...,52,53}.) The user MUST enter these numbers in ascending order. Once these have been entered, your program should print out the names of all the winners, along with how much money they have won. You are guaranteed that each person playing the lottery has bought EXACTLY one ticket and is listed once in the file.

In your implementation please adhere to the following guidelines:

1) Use a record to store information about each ticket.
2) Dynamically allocate an array to store all the ticket information based on the first number in the file.
3) Use an enumerated type for the four possible winning values.

References

Textbook: Chapter 2, 3

Input File Format

The input file your program will take in will have the following format:

The first line will contain a single integer n, the total number of tickets bought.
The next 2n lines will contain information about each ticket bought. The information for a single ticket will be on two lines. The first of the two lines will contain the last name of the ticket buyer, followed by a space, followed by the first name of the ticket buyer. Both the first and last name are guaranteed to be 19 characters or less. The following line will contain the six integers chosen by that buyer all in ascending order separated by spaces.

Here is a sample file, input.txt:

```
5
Llewellyn Mark
1 15 19 26 33 46
Young Brian
17 19 33 34 46 47
Guha Arup
1 4 9 16 25 36
Siu Max
17 19 34 46 47 48
Balci Murat
5 10 17 19 34 47
```

User Input Specification

1. The file name entered will be valid.
2. The winning lottery numbers will be valid and entered in ascending order.

Output Specification

Your output should contain one winner per line. Each line of output should be of the following format:

```
First Last matched X numbers and won $Y.
```

where First is the first name of the winner, Last is the last name of the winner, X is the number of winning numbers the player picked correctly, and Y is the prize money won.

Output Samples

Here is one sample output of running the program. Note that this is NOT a comprehensive test. You should test your program with different data than is shown here based on the specifications given. The user input is given in *italics* while the program output is in bold. (Note: The following output is based upon the sample input file shown above.)

```
Enter the name of the file with the ticket data.
input.txt
Enter the winning Lottery numbers
17 19 33 34 46 47
Mark Llewellyn matched 3 numbers and won $10.
Brian Young matched 6 numbers and won $1000000.
Max Siu matched 5 numbers and won $10000.
Murat Balci matched 4 numbers and won $100.
```

Grading Details

Your program will be graded upon the following criteria:

1) Adhering to the implementation specifications listed on the first page.
2) Your algorithmic design.
3) Correctness.
4) The frequency and utility of the comments in the code, as well as the use of white space for easy readability. (We're not kidding here. If your code is poorly commented and spaced and works perfectly, you could earn as low as 80-85% on it.)
5) Compatibility to either cygwin in Windows or gcc under olympus. (If your program does not compile in either of these environments, you will get a sizable deduction from your grade.)

Note: There is a more efficient way to determine the number of matching tickets compared to the straightforward method. No credit will be taken off if you don't discover this method, but please do try to look for an efficient solution to this part of the problem.

Restrictions

Name the file you create and turn in *lottery.c*. Although you may use other compilers, your program must compile and run using gcc or cygwin. Your program should include a header comment with the following information: your name, course number, section number, assignment title, and date. You should also include comments throughout your code, when appropriate.

Deliverables

A single source file named *lottery.c* turned in through WebCT.