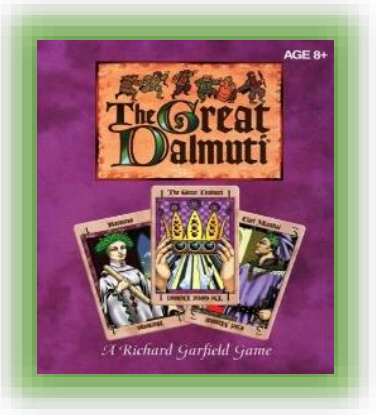
Bluegrass Community and Technical College

Programming Requirements Document

Dalmuti Deal – Two-Dimensional Array

NARRATIVE DESCRIPTION

We will once again use ***The Great Dalmuti*** card deck to practice Java concepts.



Last week you created a one-dimensional array to hold the 80-card deck for ***The Great Dalmuti.*** This week you will create a two-dimensional array to hold cards dealt to the players. You will use the one-dimensional array to hold the entire card deck (ease of use for shuffling and playing new games) and you will use the two-dimensional array to hold each player's card for a current game being played.

For this assignment and to make your code is less complicated, we will assume the game is played with 4.

An example of a shuffled one-dimensional card deck might be:

One-Dimensional String Array Shuffled Card Deck	
Index	Name of Card
0	7: Seamstress
1	2: Archbishop
2	9: Cook
3	9: Cook
4	4: Baroness
5	1: Dalmuti
6	11: Stonecutter
7	12: Peasant
8	3: Earl Marshal
9	7: Seamstress
10-75	etc.
76	6: Knight
77	2: Archbishop
78	6: Knight
79	5: Abbess

Since there are 80 cards and 4 players, each player will have 20 cards. You will use a two-dimensional String array (20 rows by 4 columns) to hold cards dealt to the players. The columns will represent the 4 players and the rows will represent the cards.

After the one-dimensional array has been created and the cards shuffled, you can "deal" the cards to the players by copying a card from the one-dimensional array to the two-dimensional array.

It may be helpful to remember how nested for-loops are used to process elements of a two-dimensional array (sample array name is someArray):

```
for (int row = 0; row < someArray.length; row++)      {
      for (int column = 0; column < someArray[row].length; column++)   {

            //   Code to process elements in the two-dimensional array
            //   playerCards[row][column] is an element in the array
      }
}
```

Using the one-dimensional array from the previous page, dealing the cards would create the following two-dimensional array:

Two-Dimensional String Array *Player Cards*			
Player1	Player2	Player3	Player4
7: Steamstress	2: Archbishop	9: Cook	9: Cook
4: Baroness	1: Dalmuti	11: Stonecutter	12: Peasant
3: Earl Marshall	7: Seamstress	*Other valid card*	*Other valid card*
Other valid cards for several rows	*Other valid cards for several rows*	*Other valid cards for several rows*	*Other valid cards for several rows*
6: Knight	2: Archbishop	6: Knight	5: Abbess

Once the two-dimensional array is populated, it is easy to display the cards from a row, or column. As you can see, the first column represents all the cards for Player1, the second column represent all the cards for Player2, etc.

Step 1: Create and Populate the Arrays

Create a one-dimensional String array to hold cards for ***The Great Dalmuti*** game and a two-dimensional String array to hold cards after they "dealt" to the 4 players:

- Create the one-dimensional array and populate the array with the 80 cards from the Dalmuti deck (like the Module 10 assignment)
- Shuffle the cards in the one-dimensional array (like the Module 10 assignment)
- Deal the cards to the four players one at a time from the top of the deck (populate the two-dimensional array using the shuffled cards from the one-dimensional array)

To "modularize" the application, at a minimum, use static methods for:

- Shuffling the card deck (one-dimensional array)
- Dealing the cards (loading values into the two-dimensional array)

Step 2: Displaying Cards for Players

Create a static method to display a single player's cards. The method should have one parameter: the column to display (which player).

From the main method, call this new static method to display the card for the players (from the two-dimensional array). Display the cards one player at a time. Make sure the display is reading the two-

dimensional array and not from the one-dimensional array. This task helps assess your ability to manipulate and use a two-dimensional array.

Step 3: Allow Repeat Games

We will not code the remainder of the game. However, allow the players to start another game. This requires

- the one-dimensional array to be shuffled again, and
- the cards to be dealt again (populating the two-dimensional array again) and
- the card to be displayed again one player at a time.

Restrictions:

Do not use any user-defined classes for this assignment. We will include used-defined classes with arrays in Module 12.

Software Requirements

R1: The user interface (what the use sees and how they interact with the program) is intuitive, clear, and easy to use.
R2: An 80-element single-dimension array is used to hold strings which represent cards from a Dalmuti deck.
R3: The single-dimension array is created as defined and with the 80 cards it should hold.
R4: The cards in the array are "shuffled" by repetitively and randomly swapping cards in the array.
R5: A two-dimensional String array is used to hold players' cards.
R6: Static methods are used to shuffle cards, deal cards, and display cards (by player).
R7: There are no out-of-bound issues with either array.
R8: Cards are properly dealt from the top of the deck (beginning of the array) one at a time alternately between players. In other words, player 1 get the first card, player 2 gets the second card, player 3 gets the third card, etc.
R9: The game can be repeated multiple times with newly shuffled cards and newly dealt cards. The new cards are displayed properly.
R10: Previous standard requirements are included (documentation, selection of proper data types, constants, etc.).

Security Considerations

Array out-of-bound errors can cause security vulnerabilities.

Special Notes

None provided for this assignment.

Change Request Form

Students who wish to obtain written permission to alter the assignment or to use features/statements/structures before they are introduced in class, must complete a *Change Request Form* (link in Blackboard in left-hand navigation bar) and follow all guidelines provided there.