It turns out that BINGO is much like games based upon a deck of cards. One must randomize the selection of the BINGO items, select one at a time, keep using items from the collection until none remain, and then refill the container holding the items, etc.
The basic objective of the programs you will be writing for Programming Project 2 is to create classes that enable you to create the basics for games that require keeping track of a set of items from which one or more may be removed at a time. The games you will be working with are Bingo and a game requiring a deck of cards. Both games require very similar functionality that can be realized by implementing an interface. In BINGO, you must produce bingo tokens from a BINGO container holding a set of items (no duplicates). Note that I am not going to use the standard BINGO language which is "BINGO Bag." The reason I will not use the word "bag" is because "bag" is a special kind of collection in computer science: one in which order does not matter, but duplicates are allowed. So, our Bingo set contains 75 tokens which include both a letter and a number. The letters are 'B', ' I ', 'N', 'G' and 'O'. The numbers are 1-75. The numbers 1-15 are associated with 'B', 16-30 are associated with ' I ' and so on. The tokens are pulled randomly from the set and not reused until the end of the game.

By the way, there are approximately 552,446,474,061,128,648,601,600,000 (five hundred fifty-two septillion, four hundred forty-six sextillion, four hundred seventy-four quintillion, sixty-one quadrillion, one hundred twenty-eight trillion, six hundred forty-eight billion, six hundred one million, six hundred thousand) possible arrangements of the numbers on a bingo card.
In most card games using a single standard deck the basic operations are almost exactly the same. You start with a set of 52 cards in random order without duplicates (a set again). The cards are pulled from the deck and given to individual players to create poker hands (we will assume 5 cards per hand) and are not returned to the deck until it is time to reshuffle the cards.
At this point notice that bingo uses a bingo token which can be identified by a number between 1-75. Poker uses a card which can be identified with a number between 0-51 (as in assignment #1). The task of generating those tokens is nearly the same for both games.
Also notice that the tokens (either a bingo token or a playing card) contain several of the same properties. Each has a number. Each will need to set other values such as a letter (BINGO) or face value and suit (king of clubs).

We will speak generically about the activities we might perform on these items in order to create a game:

1. Create a new set of BINGO tokens or a new deck of cards:
reset()
2. Mix them up:
mixup() // this was shuffle() the deck of cards in the first assignment
3. Deal a card or return the next BINGO number:
getNextItem()
4. Examine the next item without returning it:
peek()
5. Determine how many items are left:
int numberRemaining()
6. Return a string containing all the items in the collection:
toString()

The task in this project will be to write an *interface definition* for the above functionality (called GameOfChance) and then to implement the interface as appropriate for the DeckOfCards/Card classes from the first assignment and for BingoSet/BingoToken classes (analogous to DeckOfCard and Card) that you will implement from scratch.

Note that you must make an additional change to the DeckOfCards class and you must use this sort of approach for the BingoToken/BingoContainer classes. We want you to get some experience working with arrays instead of ArrayLists so your DeckOfCards and BingoSet classes will employ arrays instead of ArrayLists.

You will have some additional methods that you will need to create:

- String dealPokerHand(int numHands)
- String generateBingoBoard()

One of the goals is to decide where these methods should go. The dealPokerHand() method will deal 5 cards to up to 10 players (the max that could come out of a deck - and obviously this would be 5-card stud!!). The generateBingoBoard() method will fill a two-dimensional array with randomly generated values according to the rules of the game. The methods should handle empty deck or out of tokens respectfully. (return a null) Both method will return a nicely formatted String displaying the poker hand or the Bingo card. A typical Bingo card might look like this:

B	**I**	**N**	**G**	**O**
14	19	45	50	63
5	29	32	59	75
15	18	**Free**	46	61
7	24	39	52	72
6	17	37	47	66

In addition to the 4 classes and the interface you will write a GameOfChanceTester class. This class will:

1. Test all the Interfaces methods of both classes polymorphically by:
    a. Creating an array of GameOfChance Objects and fill it with one DeckOfCards and one BingoContainer.
    b. Loop through the Array and call each of the Interface methods.
2. Test both the dealPokerHand( )and generatorBingoBoard( ) by calling each and displaying the results.
3. No user interaction should be required.

Submission Requirements:

Your project must be submitted using the instructions below. Any submissions that do not follow the stated requirements will not be graded.

1. You should have three files for this assignment:

- Card.java The Card class
- DeckOfCards.java The Deck of Cards
- BingoToken.java - Bingo chip class

- BingoSet.java - All the bingo tokens
- GameOfChance - Interface for games of chance
- GameOfChangeTester - the tester that test all the methods.
- UML Class diagram of your 5 Classes (Card, DeckOfCards, BingoToken, BingoSet and GameOfChance)(Dia file or image file , jpg, gif, pdf etc)
- The javadoc files for all the classes except the tester class. (Do not turn in html file just generate them)