CIS 181 – Lab 9

War Card Game

Objectives

- To gain experience with using stacks to solve a problem.
- To learn more about using an Interface.

Description

War is a card game typically played between 2 players. A single deck of cards is split between the two players, resulting in each player starting with 26 cards. During a turn, each player draws the top card of their respective deck and compares it to the other player's card. The player with the higher card wins the round and collects both cards, putting them into their pile of winnings. If both players' cards are of equal value, then War occurs. During War, each player draws 3 cards (or however many are remaining if less) and places them into their respective war piles without looking. Then, a 4th card is drawn and compared, just like the initial draw. War can occur multiple times in the same turn. If a player runs out of cards in their deck, the player takes the cards from their pile of winnings and shuffles them into their deck. The game is over when one player no longer has any cards in their deck and pile of winnings.

Check it out online here: https://cardgames.io/war/ (Credit goes to the website's author)

Stacks

A Stack is an Abstract Data Type (ADT) that works similarly to an array in which you can only access one end. A real world example of this would be similar to a deck of cards. In most card games, a card can only be drawn from the top of the deck.

Stacks work with a concept known as Last-In-First-Out (LIFO). This means that the last element added to the stack is the first element accessible. There are several operations that can be used with a stack:

Push() Add one element to the top of the stack.

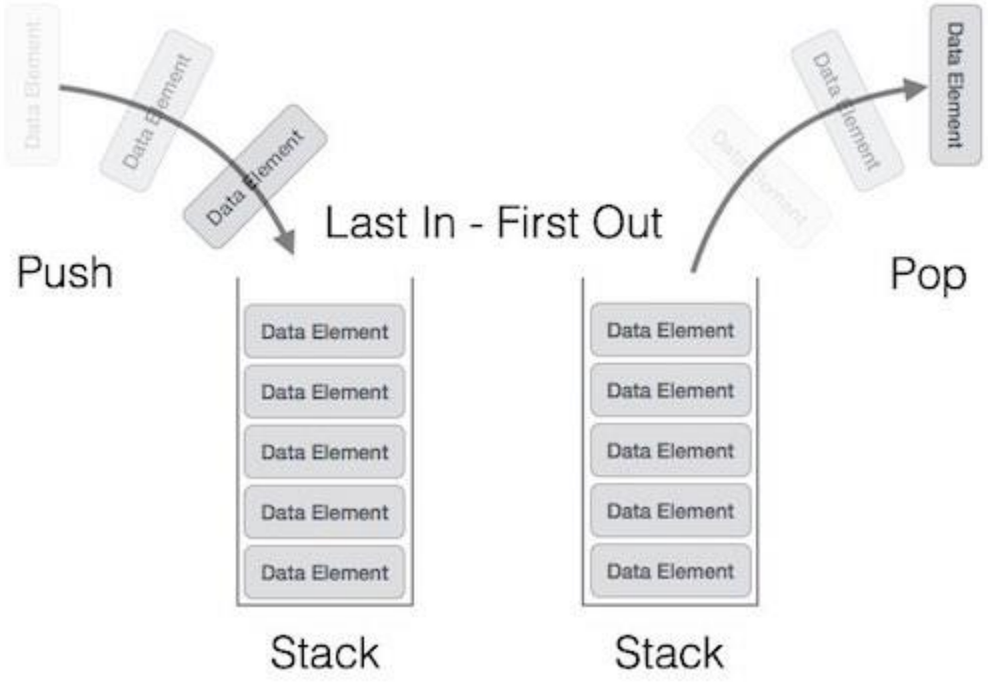
Pop() Remove one element from the top of the stack and return that element.

Peek() Return the first element from the top of the stack, but do not remove it from the stack.

IsFull() Return a Boolean value that is true if the number of elements in the stack is equal to the stack's maximum size.

IsEmpty() Return a Boolean value that is true if there are zero elements in the stack.

A visual representation of this shows as follows:



For image credit, or more information on stacks, check out https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm.

Exercises

1. Download the following source files:
   - Deck.java
   - DeckInterface.java
   - Card.java
   - WarCardGame.java (**Note:** The only class you need to implement.)
2. In Eclipse, create a new Java project called “Lab 9”, and import the 4 files.
3. Compile and run the program. A prompt should appear asking the user to select a game mode. When prompted to select a game mode, choose “Playable”. Pressing “Q” will quit the playable version. If you do not select playable, the program will be stuck in an infinite loop and will need to be terminated manually.
4. Read through the downloaded files to understand the methods that you will need to utilize in the WarCardGame.java file.
   - Deck.java
     - public void shuffle()
     - public void push(Card card)
     - public Card pop()
     - public boolean isEmpty()
     - public Card peek()

- Card.java
    - public String getName()
    - public int getValue()

5. Following the commented steps inside of WarCardGame.java, implement the functionality of the card game. Pay attention to the order of the steps as it may require you to complete methods before calling them.
6. Compile and run the program again. Make sure it works correctly.
7. Comment your code.
8. Demo the project and submit WarCardGame.java to myCourses.