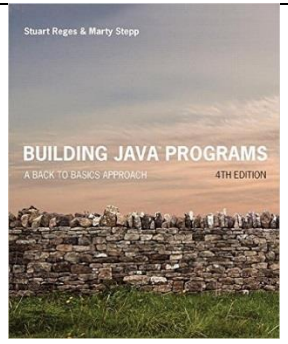
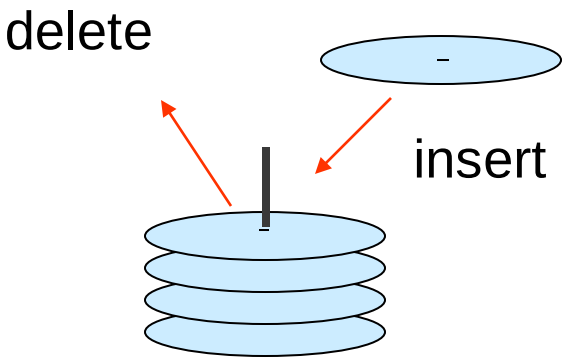
Stacks and Queues

Reading Assignment: Read Chapter 14
Building Java Programs (Stuart Reges and Marty Stepp)



Stack ADT

- A stack is a list in which insertions and deletions take place at the same end. This end is called the top of the stack.
- A stack is know as a LIFO (Last in First Out) list.

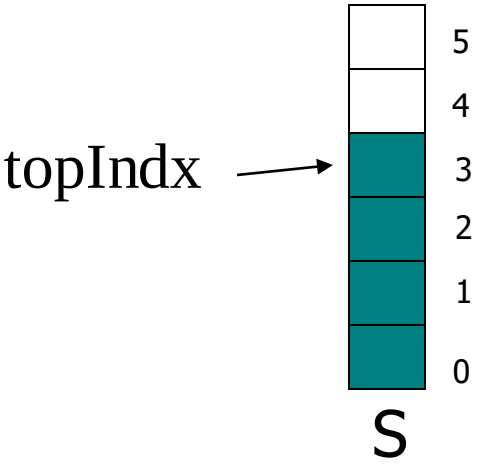


Source: http://sciencebusiness.technewslit.com/?p=6715

- Primary operations:
  - push(e) – Add an element e to the top of the stack.
  - pop() – Remove an element from the top of the stack.
  - top() – returns the top element.
  - isEmpty – Returns true if the stack is empty.
  - isFull – Returns true if the stack is full.

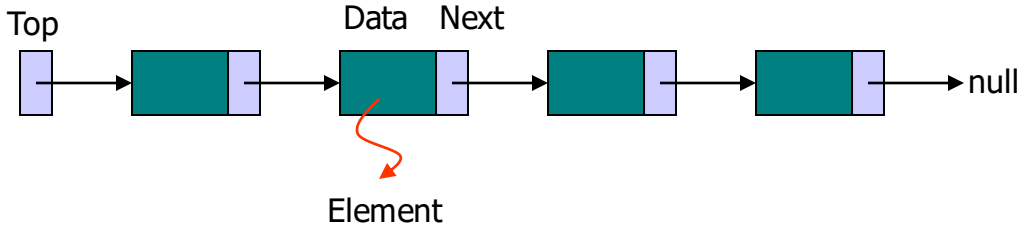
Array Implementation Stack ADT

```
public class Stack {
        private int topIndx = -1, capacity = default;
        private Object S[];
        public Stack(int size) { capacity = size; S = new Object[capacity]; }
        public boolean isEmpty() { return topIndx<0; }
        public boolean isFull() { return topIndx-1==capacity; }
        public void push(Object Element) { S[++topIndx] = Element; }
        public Object pop() {
                if (isEmpty()) return null;
                Object Element;
                Element =  S[topIndx];
                S[topIndx--] = null;
                return Element;
        }
        public Object top() { return S[topIndx]; }
}
```



Linked Implementation Stack ADT

```
public class Stack {
        private Node Top = null;
        private int size = 0;
        public boolean isEmpty() { return Top==null; }
        public boolean isFull() { ???  }
        public void push(Object Element) {
                Node Tmp = new Node(Element);
                Tmp.Next = Top; Top = Tmp; size++;
        }
        public Object pop() {
                Node Tmp = Top;
                Top =  Top.Next; size--;
                return Tmp.Data;
        }
}
```



Applications of Stacks

Many collections of items are naturally regarded as stacks:

- **A stack of plates in a cafeteria.**
- **Return addresses of method calls in an executing program.**
- **The simplest application of a stack is to reverse a word. You push a given word to stack - letter by letter - and then pop letters from the stack.**
- **Another application is an "undo" mechanism in text editors; this operation is accomplished by keeping all text changes in a stack.**

Plates in a Cafeteria

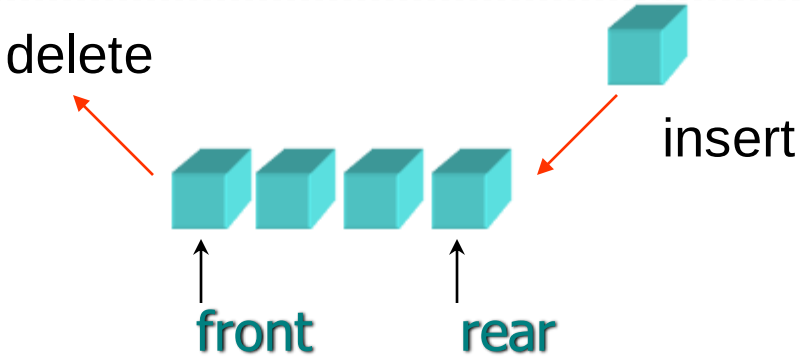
- Plates are stacked in a column.
- A new plate is added to the top of the column of plates.
- Plates are removed from the top of the column of plates, so that the last plate added is the first removed.

Method Return Addresses

- Programs often make chains of method calls.
- The last method called is the first to return.
- The compiler uses a stack to maintain the list of return addresses for executing methods in a stack.

Queue ADT

- A queue is a list in which insertions take place at one end called the rear of the queue and deletions take place at the other end called the front of the queue.
- A queue is know as a FIFO (**First in First Out**) list.



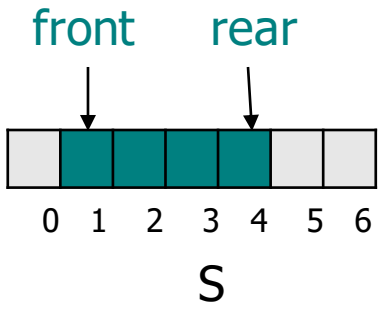
Source: https://www.fhwa.dot.gov/publications/research/operations/its/06108/03.cfm

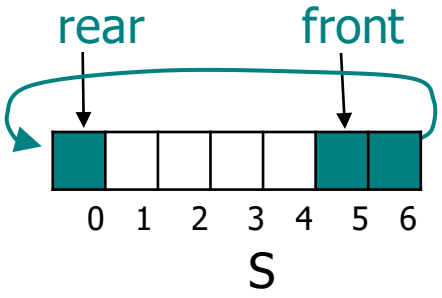
Primary operations:

- enqueue(e) – Add element e to the rear of the queue.
- dequeue() – Remove the front element from the queue.
- first() – returns the front element.
- isEmpty – Returns true if the queue is empty.
- isFull – Returns true if the queue is full.

Array Implementation Queue ADT

```
public class Queue {
    private int front = 0, rear = -1, capacity = default;
    private Object S[];
    public Queue(int size) { capacity = size; S = new Object[capacity]; }
    public boolean isEmpty() { return front==(rear+1)%capacity; }
    public boolean isFull() { ??? }
    public void enqueue(Object Element) {
        rear = (rear+1)%capacity; S[rear] = Element;
    }
    public Object dequeue() {
        Object Element =  S[front]; S[front] = null;
        front = (front+1) % capacity;
        return Element;
    }
}
```

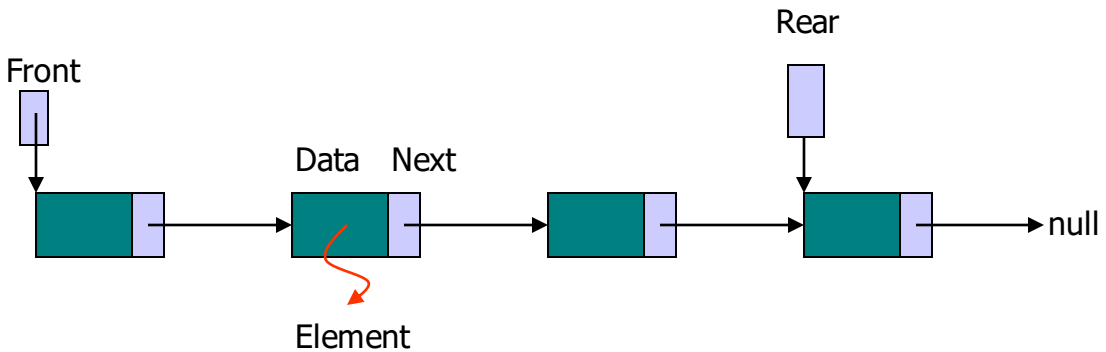




- Array S is treated as circular (joined at the ends)

Linked Implementation Queue ADT

```
public class Queue {
    private Node Rear = null, Front = null;
    private int size = 0;
    public boolean isEmpty() { return Front==null; }
    public void enqueue(Object Element) {
        Node Tmp = new Node(Element);
        if (Rear==null) Rear = Front = Tmp;
        else { Rear.Next = Tmp; Rear = Tmp;}
        size++;
    }
    public Object dequeue() {
        Node Tmp = Front;
        Front =  Front.Next; size--;
        if (Front==null) Rear = null;
        return Tmp.Data;
    }
}
```

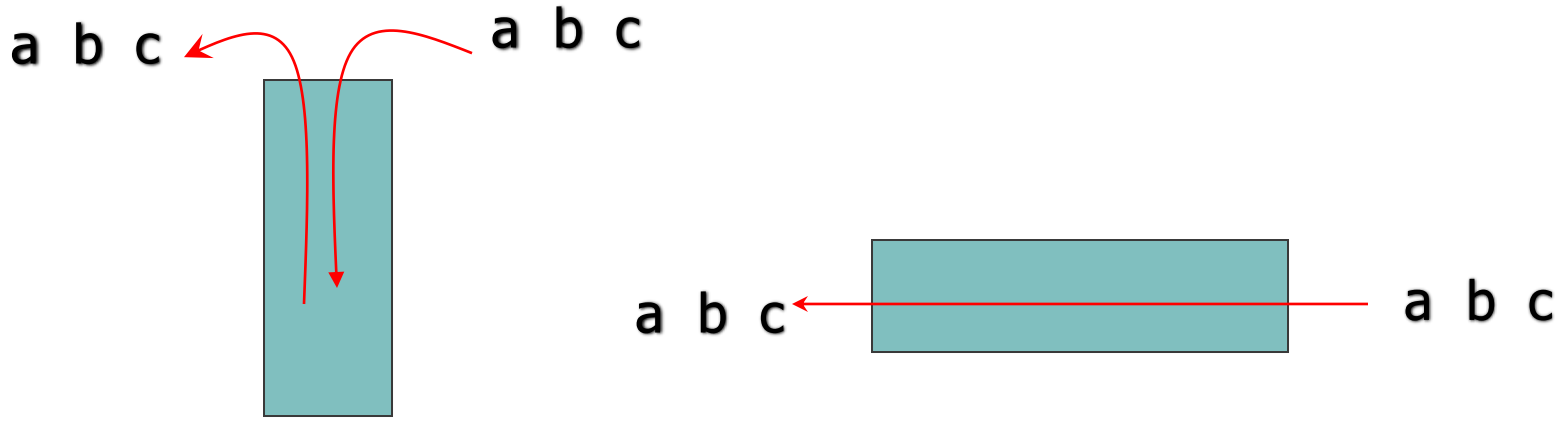


Queue Applications

- A print server may service many workstations in a computer network. The print server uses a queue to hold print job requests waiting to be sent to the printer.

- A simulation of cars going through a toll booth would use a queue to hold cars lining up to pay toll.

Stacks v.s. Queues



Application of Stack and Queue palindromes

A string of characters is a palindrome if and only if it reads the same forward and backward.

Examples:

Radar

level

step on no pets

pull up if i pull up

was it a rat I saw

Application of Stack and Queue palindromes

A recursive definition follows:

(1) The empty string is a palindrome.

(2) A string consisting of a single character is a palindrome.

(3) If w is a palindrome and a is a letter in the alphabet, then awa is a palindrome.

(4) A string of characters is a palindrome if and only if its being so follows from finitely many applications of rules (1) through (3) above.

Stacks v.s. Queues –Checking for palindromes