Stacks
• Last in First Out (LIFO)
  – Processing Data in Reverse Order
  – Time
  – Backtracking
  – Call Stack

• Stack Operations
  – Push: Add new element to top of the stack
  – Pop: Remove and return top element of the stack
  – Peek: Observe but not remove the first element in a Stack
  – Print: Print all elements in a Stack
Stack Implementations
- Array
- Linked List

One Major Reference
- Head

Array Stack

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>6</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>-</td>
</tr>
</tbody>
</table>

Linked List Stack

7 → 8 → 5 → 6

Head
Stacks

Stack\<T\>

+push(T): void
+pop():T
+peek():T
+print():void

ArrayStack\<T\>

-stack[]):T
+push(T): void
+pop():T
+peek():T
+print():void

LLStack\<T\>

-head: ListNode
+push(T): void
+pop():T
+peek():T
+print():void
• References
  – Head Index (First NULL Element)
  – Items in the Stack start from the Head Index - 1 and goes until index 0
• Push
  – Add element at the Head Index
  – Increase the Head Index by 1
• Push
  – Add element at the Head Index
  – Increase the Head Index by 1
• Push
  – Add element at the Head Index
  – Increase the Head Index by 1
• Pop
  – Save a reference to the data stored in Head Index – 1
  – Move the Head Index Backwards
  – Return the stored data
Pop
- Save a reference to the data stored in Head Index – 1
- Move the Head Index Backwards
- Return the stored data
- Pop
  - Save a reference to the data stored in Head Index – 1
  - Move the Head Index Backwards
  - Return the stored data

Array Stack

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>7</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Return Value | 6

Head Index
• **Push**
  – Create a new List Node with the Data
  – Point new List Node’s reference to Head
  – Update the Head Reference to the new List Node
• Push
  – Create a new List Node with the Data
  – Point new List Node’s reference to Head
  – Update the Head Reference to the new List Node
• **Push**
  – Create a new List Node with the Data
  – Point new List Node’s reference to Head
  – Update the Head Reference to the new List Node
• Push
  – Create a new List Node with the Data
  – Point new List Node’s reference to Head
  – Update the Head Reference to the new List Node
- Pop
  - Save reference to data store in the head
  - Move Head forward / Head = Head.link
  - Return the stored data
• Pop
  – Save reference to data store in the head
  – Move Head forward / Head = Head.link
  – Return the stored data
- Pop
  - Save reference to data store in the head
  - Move Head forward / Head = Head.link
  - Return the stored data

Linked List Stack

Return Value 6
- Pop
  - Save reference to data store in the head
  - Move Head forward / Head = Head.link
  - Return the stored data