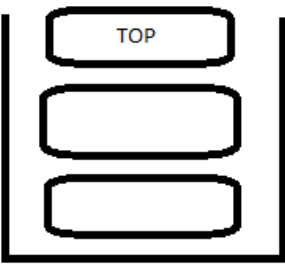


Stack		
		
Size \geq 0		
Stack() Size() Push() Pop() Peek() Empty() Search()	Stack Stack Stack Stack Stack Stack Stack x T	Stack Integer Elemento Elemento Boolean Integer

HashTable																
	<table><tr><th>Key</th><th>Value</th></tr><tr><td>k1</td><td>v1</td></tr><tr><td>k2</td><td>v2</td></tr><tr><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td></tr><tr><td>kn</td><td>vn</td></tr></table>	Key	Value	k1	v1	k2	v2	kn	vn	
Key	Value															
k1	v1															
k2	v2															
.	.															
.	.															
.	.															
kn	vn															
Size>=0																
Put() KeySet() Get()	HashTable x K , V HashTable HashTable x K	K V														

LinkedList		
<div><p>Diagram illustrating a Singly Linked List structure:</p><pre>graph LR; TOP --> Node1; Node1 --> Node2; Node2 --> Node3; Node3 --> Node4; Node4 --> NULL;</pre><p>The diagram shows a sequence of four nodes, each containing a data field (Dato) and a pointer field (Sig). The pointer field of each node points to the next node in the sequence, and the last node points to NULL. A pointer labeled TOP points to the first node.</p></div>		
Size>=0		
Pop()	LinkedList	T
Clear()	LinkedList	
isEmpty()	LinkedList	Boolean
push()	LinkedList x T	
clone()	LinkedList	LinkedList
get()	LinkedList x Interger	T