Stacks

codecademy

Stack *overflow*

Every stack has a size that determines how many nodes it can accommodate. Attempting to push a node in a full stack will result in a stack overflow. The program may crash due to a stack overflow.

A stack is illustrated in the given image.

`stackA.push(xg)` will result in a stack overflow since the stack is already full.

xf
xe
xd
xc
xb
xa

stackA

The *stack* data structure

A *stack* is a data structure that follows a last in, first out (LIFO) protocol. The latest node added to a stack is the node which is eligible to be removed first. If three nodes (`a` , `b` and, `c`) are added to a stack in this exact same order, the node `c` must be removed first. The only way to remove or return the value of the node `a` is by removing the nodes `c` and `b` .

Main methods of a *stack* data structure

The stack data structure has three main methods: `push()` , `pop()` and `peek()` . The `push()` method adds a node to the top of the stack. The `pop()` method removes a node from the top of the stack. The `peek()` method returns the value of the top node without removing it from the stack.