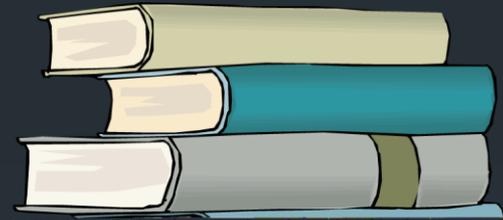




Stacks

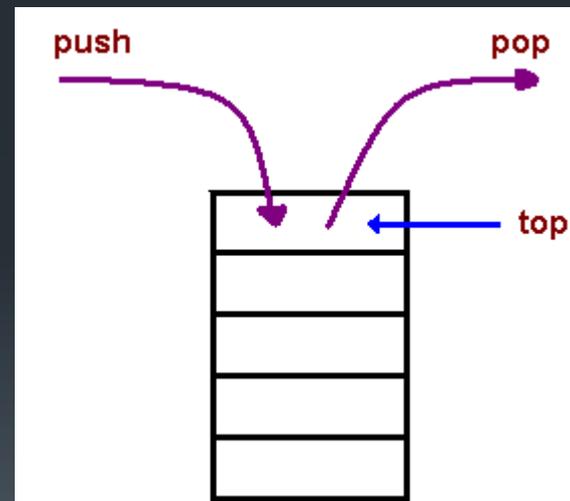
Definition

- The stack is a list-like structure in which elements may be inserted or removed from only one end.
- While this restriction makes stacks less flexible than lists, it also makes stacks both efficient and easy to implement.
- LIFO = Last-In, First-Out



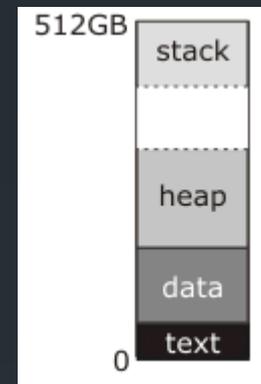
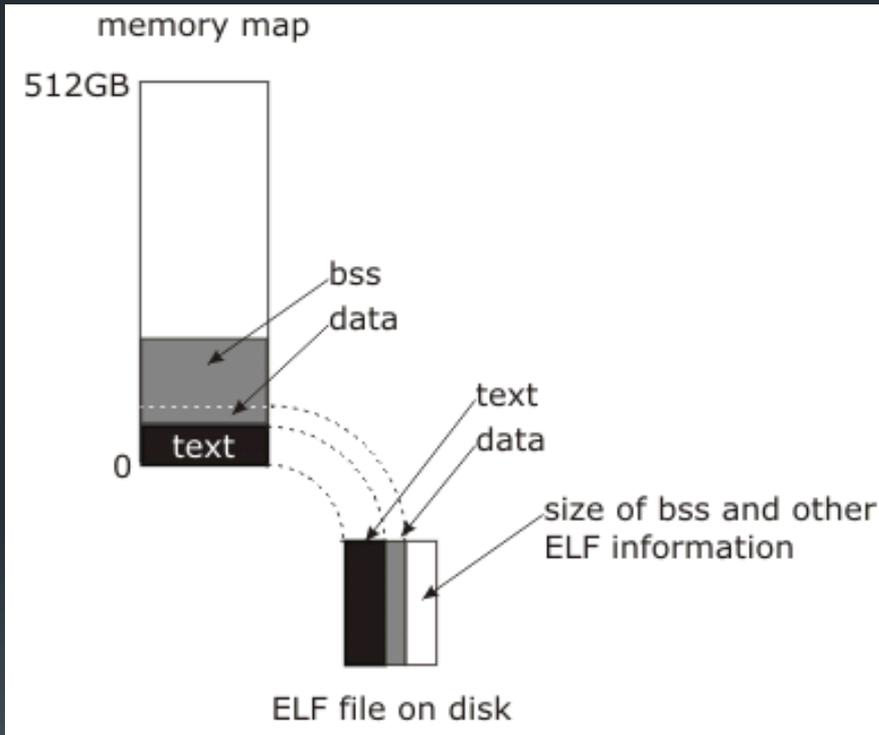
Elements and Operations

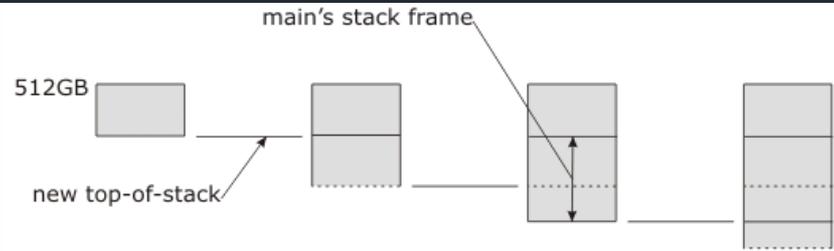
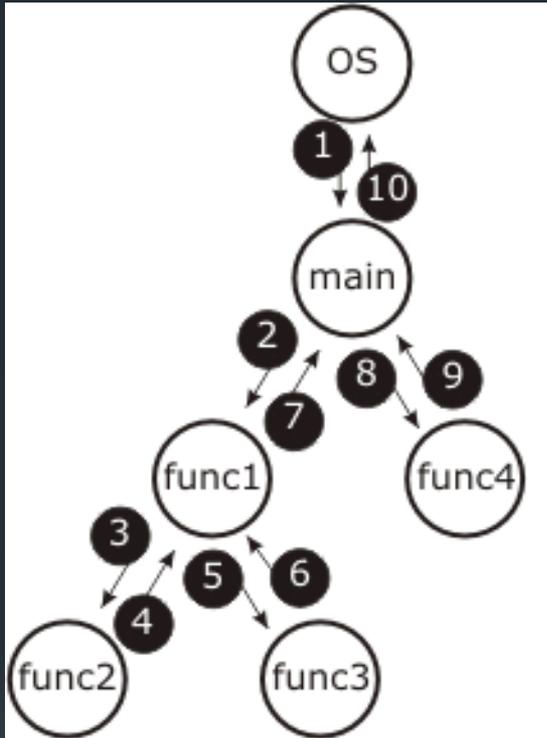
- The accessible element of the stack is called the top element.
- Elements are not said to be inserted, they are pushed onto the stack. When removed, an element is said to be popped from the stack.



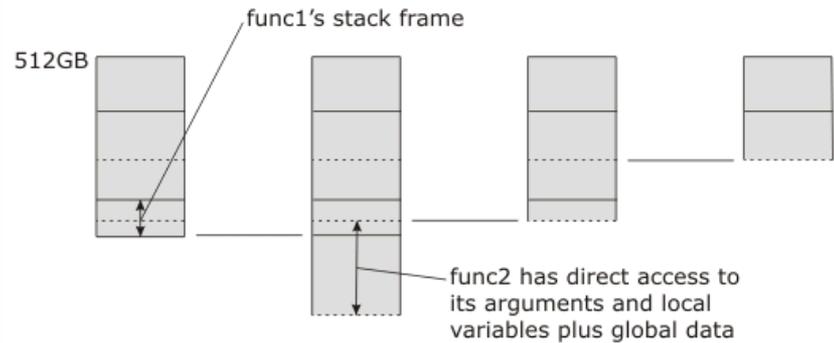
- 
- A stack grows if a single node is inserted into its top position. No way of assigning values directly to stack nodes is available in a pure stack. The formal operation that inserts a node into a stack is PUSH.
 - The top node must be removed from a stack before the values of other nodes can be retrieved.

Example

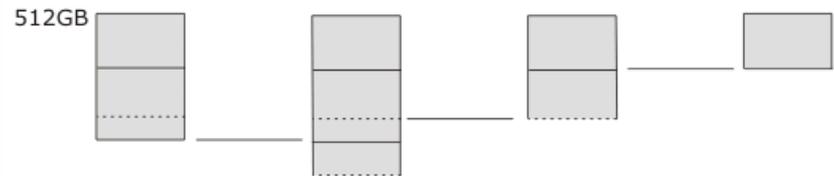




- (a) OS pushes command line on top of stack and calls main
- (b) main pushes local variables on top of stack
- (c) main pushes arguments to func1 and calls func1
- (d) func1 pushes local variables on top of stack



- (e) func1 pushes arguments to func2 and calls func2
- (f) func2 pushes local variables on top of stack
- (g) func2 returns to func1 removing arguments to func2
- (h) func1 returns to main removing arguments to func1



- (i) main pushes arguments to func4 and calls func4
- (j) func4 pushes local variables on top of stack
- (k) func4 returns to main removing arguments to func4
- (l) main returns to OS