5.1 Stack abstract data type (ADT)

Stack abstract data type

A ***stack*** is an ADT in which items are only inserted on or removed from the top of a stack. The stack ***push*** operation inserts an item on the top of the stack. The stack ***pop*** operation removes and returns the item at the top of the stack. Ex: After the operations "Push 7", "Push 14", "Push 9", and "Push 5", "Pop" returns 5. A second "Pop" returns 9. A stack is referred to as a ***last-in first-out*** ADT. A stack can be implemented using a linked list, an array, or a vector.

PARTICIPATION ACTIVITY 5.1.1: Stack ADT.

Animation captions:

1. A new stack named "route" is created. Items can be pushed on the top of the stack.
2. Popping an item removes and returns the item from the top of the stack.

PARTICIPATION ACTIVITY 5.1.2: Stack ADT: Push and pop operations.

1) Given numStack: 7, 5 (top is 7).
Type the stack after the following push operation. Type the stack as: 1, 2, 3

Push(numStack, 8)

Check Show answer

2) Given numStack: 34, 20 (top is 34)
Type the stack after the following two push operations. Type the stack as: 1, 2, 3

Push(numStack, 11)
Push(numStack, 4)

Check Show answer

3) Given numStack: 5, 9, 1 (top is 5)
What is returned by the following pop operation?

Pop(numStack)

Check Show answer

4) Given numStack: 5, 9, 1 (top is 5)
What is the stack after the following pop operation? Type the stack as: 1, 2, 3

Pop(numStack)

Check Show answer

5) Given numStack: 2, 9, 5, 8, 1, 3 (top is 2).
What is returned by the second pop operation?

Pop(numStack)
Pop(numStack)

Check Show answer

6) Given numStack: 41, 8 (top is 41)
What is the stack after the following operations? Type the stack as: 1, 2, 3

Pop(numStack)
Push(numStack, 2)
Push(numStack, 15)
Pop(numStack)

Check Show answer

Common stack ADT operations

Table 5.1.1: Common stack ADT operations.

Operation	Description	Example starting with stack: 99, 77 (top is 99).
Push(stack, x)	Inserts x on top of stack	Push(stack, 44). Stack: 44, 99, 77
Pop(stack)	Returns and removes item at top of stack	Pop(stack) returns: 99. Stack: 77
Peek(stack)	Returns but does not remove item at top of stack	Peek(stack) returns 99. Stack still: 99, 77
IsEmpty(stack)	Returns true if stack has no items	IsEmpty(stack) returns false.
GetLength(stack)	Returns the number of items in the stack	GetLength(stack) returns 2.

Note: Pop and Peek operations should not be applied to an empty stack; the resulting behavior may be undefined.

PARTICIPATION ACTIVITY 5.1.3: Common stack ADT operations.

1) Given inventoryStack: 70, 888, -3, 2
What does GetLength(inventoryStack) return?
   - 4
   - 70

2) Given callStack: 2, 9, 4
What are the contents of the stack after Peek(callStack)?
   - 2, 9, 4
   - 9, 4

3) Given callStack: 2, 9, 4
What are the contents of the stack after Pop(callStack)?

- 2, 9, 4
- 9, 4

4) Which operation determines if the stack contains no items?

- Peek
- IsEmpty

5) Which operation should usually be preceded by a check that the stack is not empty?

- Pop
- Push

CHALLENGE ACTIVITY 5.1.1: Stack ADT.

Start

Given numStack: 60, 77, 82 (top is 60)
What is the stack after the following operations?

Push(numStack, 64)
Pop(numStack)

Ex: 1, 2, 3

After the above operations, what does GetLength(numStack) return?

Ex: 5

1 2 3

Check Next

5.2 Stacks using linked lists

A stack is often implemented using a linked list, with the list's head node being the stack's top. A push is performed by creating a new list node, assigning the node's data with the item, and prepending the node to the list. A pop is performed by assigning a local variable with the head node's data, removing the head node from the list, and then returning the local variable.

PARTICIPATION ACTIVITY 5.2.1: Stack implementation using a linked list.

Animation content:

undefined

Animation captions:

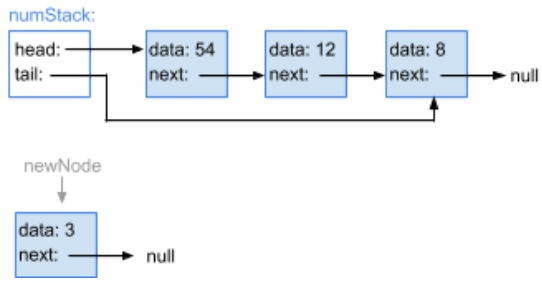
1. Pushing 45 onto the stack allocates a new node and prepends the node to the list.
2. Each push prepends a new node to the list.
3. A pop assigns a local variable with the list's head node's data, removes the head node, and returns the local variable.

PARTICIPATION ACTIVITY 5.2.2: Stack push and pop operations with a linked list.

Assume the stack is implemented using a linked list.

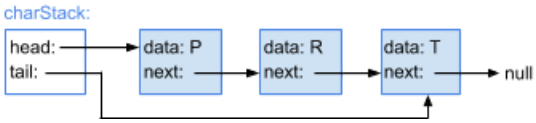
1) An empty stack is indicated by a list head pointer value of _____.
   - newNode
   - null
   - Unknown

2) For StackPush(numStack, item 3), newNode's next pointer is pointed to _____.



- Node 54
- Node 12
- null

3) The operation StackPop(charStack) will remove which node?



- Node P
- Node R
- Node T

4) StackPop returns list's head node.

- True
- False

CHALLENGE ACTIVITY 5.2.1: Stacks using linked lists.

Start

Given an empty stack numStack, what does the list head pointer point to? If the pointer is null, enter null.

Ex: 5 or null

After the following operations, which node does the list head pointer point to?

StackPush(numStack, 66)
StackPush(numStack, 69)

Ex: 5 or null

1 2 3

Check Next

©zyBooks 03/24/21 10:59 926027
Eric Knapp
STEVENSCS570Spring2021