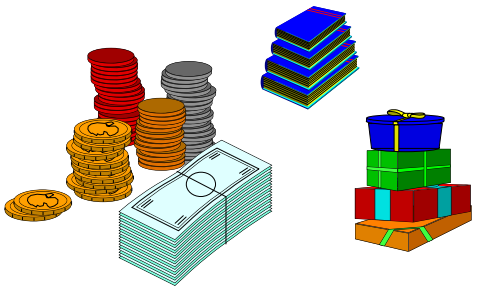
Review

- What is Template?
- Function Template.
- Class Template

Preview

- What is Stack
- Functions for managing Stack
  - push
  - pop
  - isEmpty
  - isFull
  - top
- Stack Implementation with Array
- Stack Implementation with Linked List

Stacks

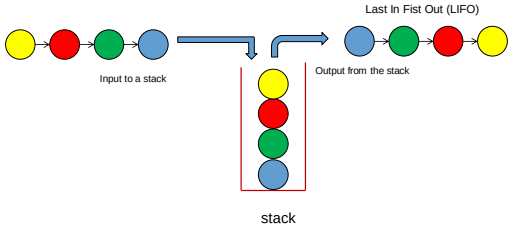


What is Stack?

Stack

- An abstract data type in which elements are added and removed from only one end.
- Rules of Stack – Last In First Out (LIFO)
- Stack can be implemented with an array or linked list.
- Stack can be used to implement Algorithms (ex. Depth-First Search)

What is Stack



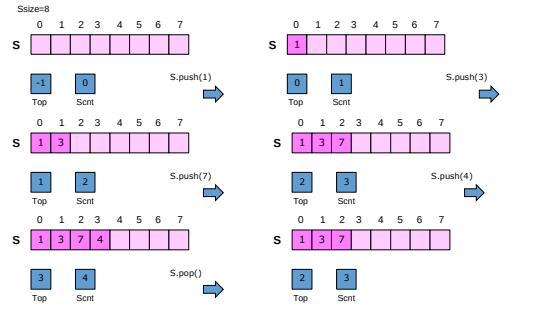
Functions for Managing Stack

- **push** – Push a new element in the top of the stack
- **pop** – Pop out a element from the top of the stack
- **isEmpty** – return true if a stack is empty, return false if a stack is not empty
- **isFull** – return true if a stack is full, return false if there is any space in the stack
- **top** – return a element from the top of the stack without pop out.

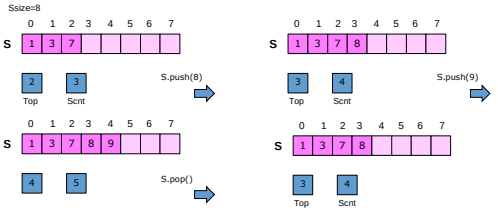
Stack Implementation with Array

- To implement a stack with array, we need keep a index for top of the stack to
  - check full stack to push a new element in the stack,
  - check empty stack to pop a element from the stack.
  - get a element information from the top of the stack.

Stack Implementation with Array



Stack Implementation with Array

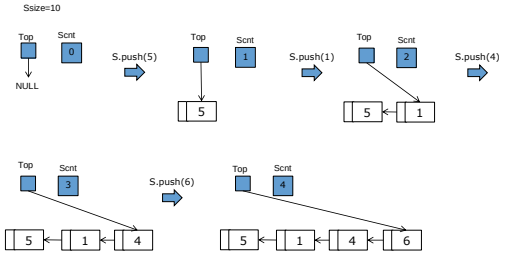


- Before pushing new element in a stack, need to check whether stack is full or not.
- Before Pop a element from the stack, we need to check whether stack is empty or not

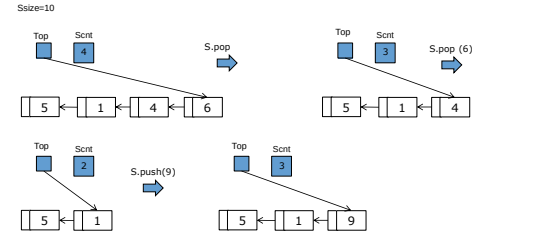
Stack Implementation with Linked List

- Initially top of stack point to NULL.
- A new node always push to the stack as a first element of a linked list.
- A node can always pop from the first element in the list.
- Stack is empty if top of stack point to NULL.
- Stack is full if the number of nodes in the list are same as size of the stack.

Stack Implementation with Linked List



Stack Implementation with Linked List

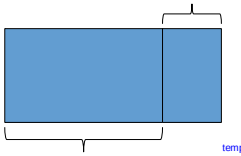


- Before pushing new element in a stack, need to check whether stack is full or not.
- Before Pop a element from the stack, we need to check whether stack is empty or not

Stack Implementation with Linked List

Stack Node

Pointer to Stack node



Any Data Type

```
template <class DataType>
struct StackNode
{
  DataType Data; // data can be any types
  StackNode<DataType> *Next; // point to the next node
};
```



```
#ifndef STACK_H
#define STACK_H

template <class DataType>
struct StackNode
{
   DataType data; // data can be any types
   StackNode<DataType> *next; // point to the next node
};


template <class DataType>
class Stack{
private:
         StackNode <DataType> *top;  // point to top of node in the stack
         int stacksize;              // Stack size
         int nodeCount;              // number of node in the stack
         void clean();               // utility function to clean memory space
public:
         Stack (int=10);           // constructor create a stack with size
         ~Stack();                 // destructor
         bool isEmpty() const;     // check empty stack for pop
         bool isFull() const;      // check full stack to insert new data
         void push (const DataType);     // push a node in the stack
         DataType  pop();
         DataType  topStack() const;
};

#endif
```