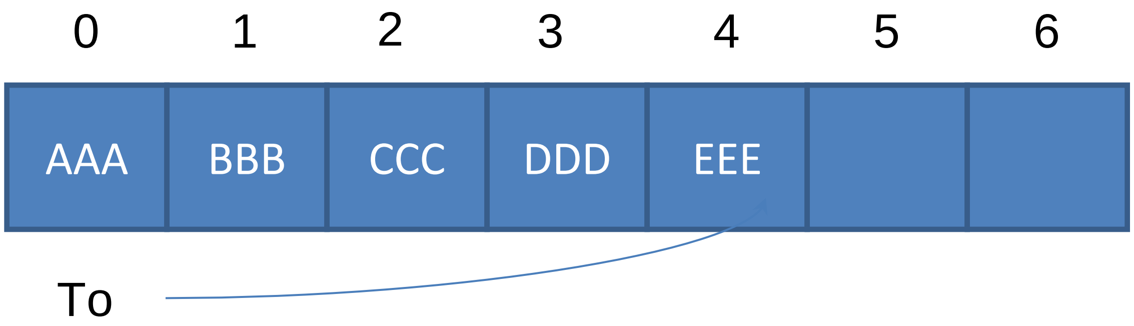
Data Structures
CSE207
(Stack)

Stack

- It is a linear data structure consisting of list of items.
- In stack, data elements are added or removed only at one end, called ***the top of the stack***.

Stack: AAA, BBB, CCC, DDD, EEE

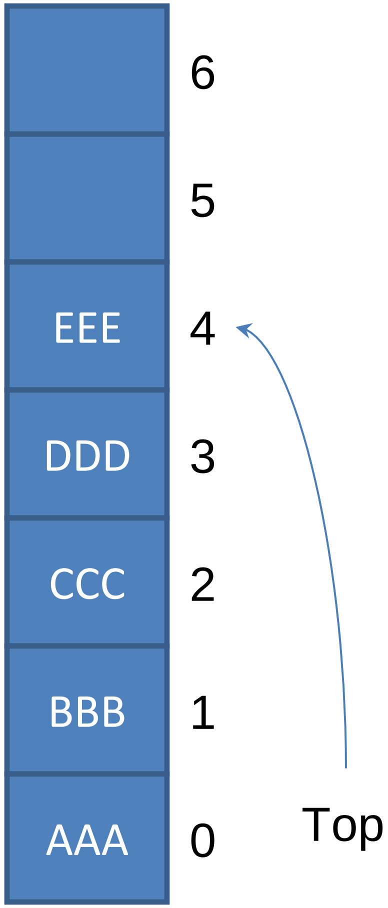


0	1	2	3	4	5	6
AAA	BBB	CCC	DDD	EEE		

Figure: Diagrams of Stacks

Stack

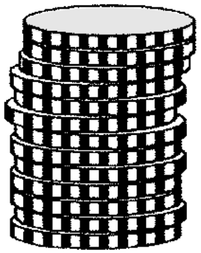
- Two basic operations are associated with stack:
- “Push” operation is used to insert an element into a stack.
- “Pop” operation is used to delete an element from a stack.



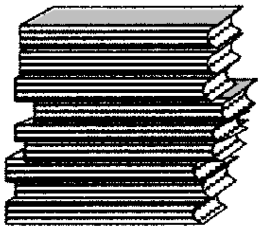
Stack: AAA, BBB, CCC, DDD, EEE

Stack

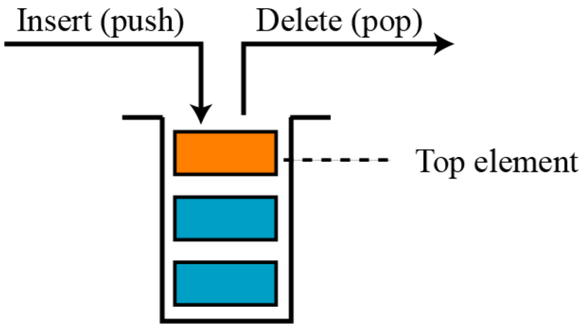
- A stack is a restricted linear list in which all additions and deletions are made at one end, the top.
- If we insert a series of data items into a stack and then remove them, the order of the data is reversed.
- This reversing attribute is why stacks are known as last in, first out (LIFO) data structures.



Stack of coins



Stack of books



Computer stack

Stack

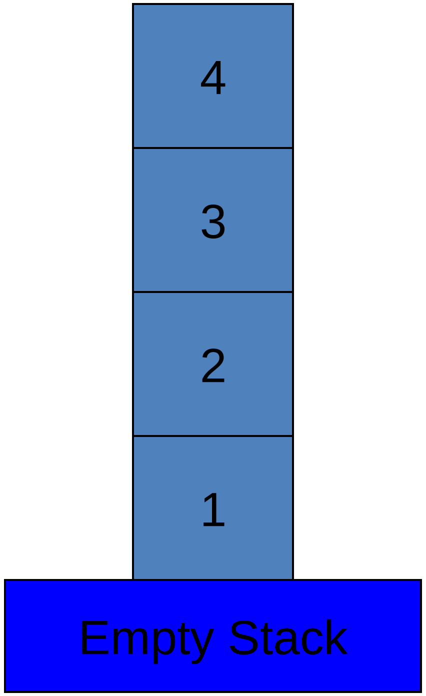
Stack has only two operation

Push: insert data on top

Pop: remove data from top

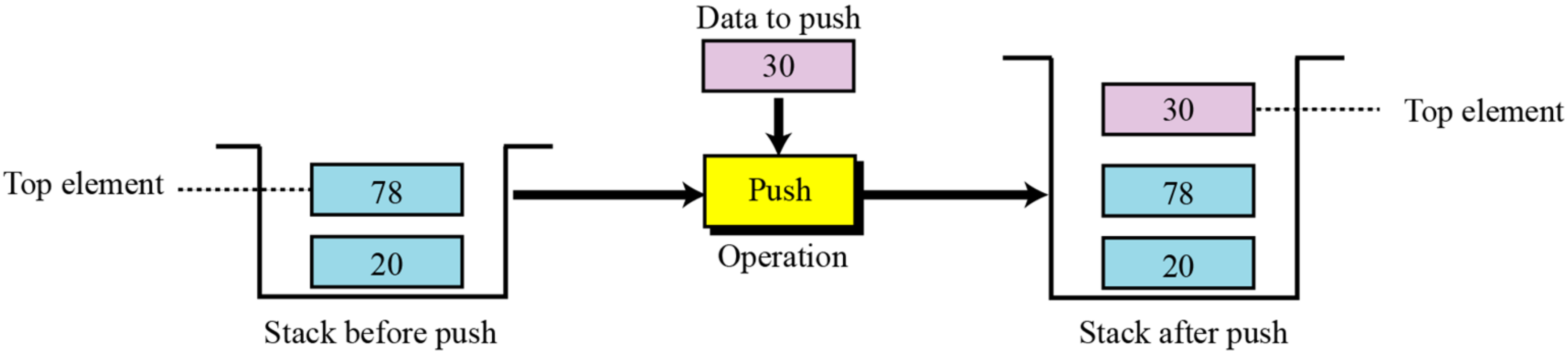
Also known as LIFO structure

Last In First Out



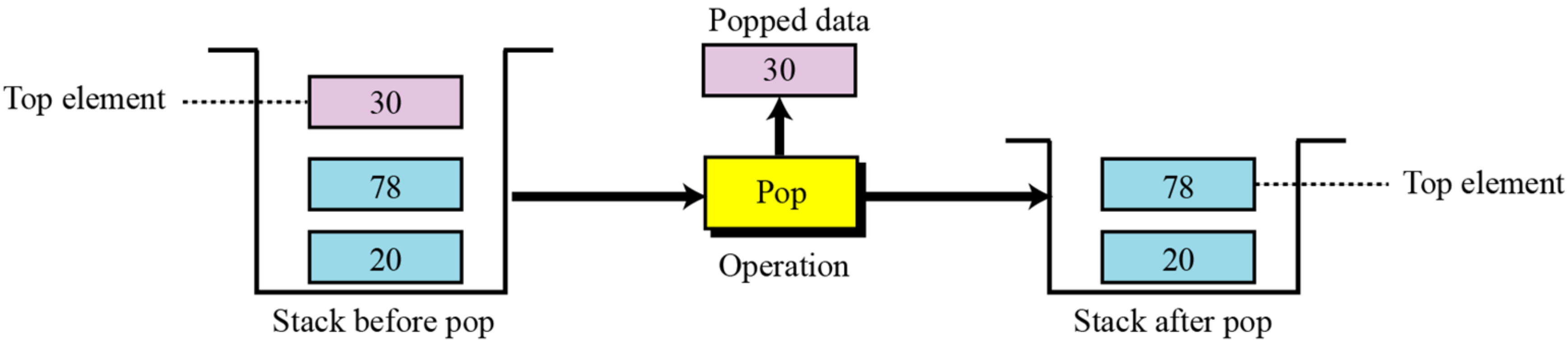
The *push* operation

- The *push* operation inserts an item at the top of the stack. The following shows the format.



The pop operation

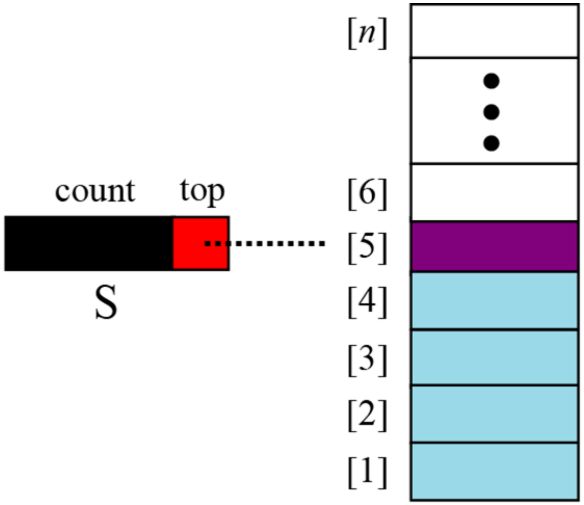
- The pop operation deletes the item at the top of the stack. The following shows the format.



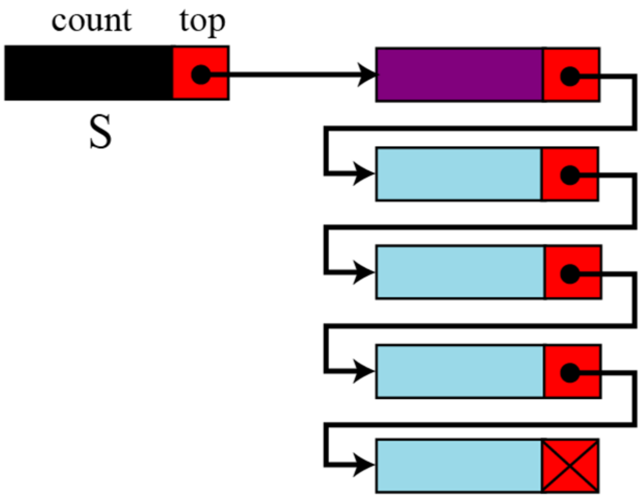
Error checking

- There are two stack errors that can occur:
  - Underflow: trying to pop (or peek at) an empty stack
  - Overflow: trying to push onto an already full stack

Implentation



b. Array implementation



c. Linked list implemenation

Algorithms for Push and Pop

- Stack is the place where to store data.
- Top represents in which location the data is to be inserted.
- MaxSTK is the maximum size of the stack and finally, Item is the new item to be added.

For Push Operation

```
1. If Top = MaxSTK then Print: Overflow and Return.
   /*…Stack already filled..*/
2. Set Top := Top +1
3. Set Stack[Top] := Item
4. Return.
```

For Pop Operation

```
1. If Top = -1 then Print: Underflow and Return.
   /*…Stack already Empty..*/
2. Set  Item := Stack[Top]
3. Set Top := Top - 1
4.  Return.
```

Some uses of stacks

- Stacks are used for:
  - Any sort of nesting (such as parentheses)
  - Evaluating arithmetic expressions (and other sorts of expression)
  - Implementing function or method calls
  - Keeping track of previous choices (as in backtracking)