Book

A Simplified Approach
to
Data Structures

Prof.(Dr.)Vishal Goyal, Professor, Punjabi University Patiala

Dr. Lalit Goyal, Associate Professor, DAV College, Jalandhar

Mr. Pawan Kumar, Assistant Professor, DAV College, Bhatinda

Shroff Publications and Distributors

Edition 2014

STACKS

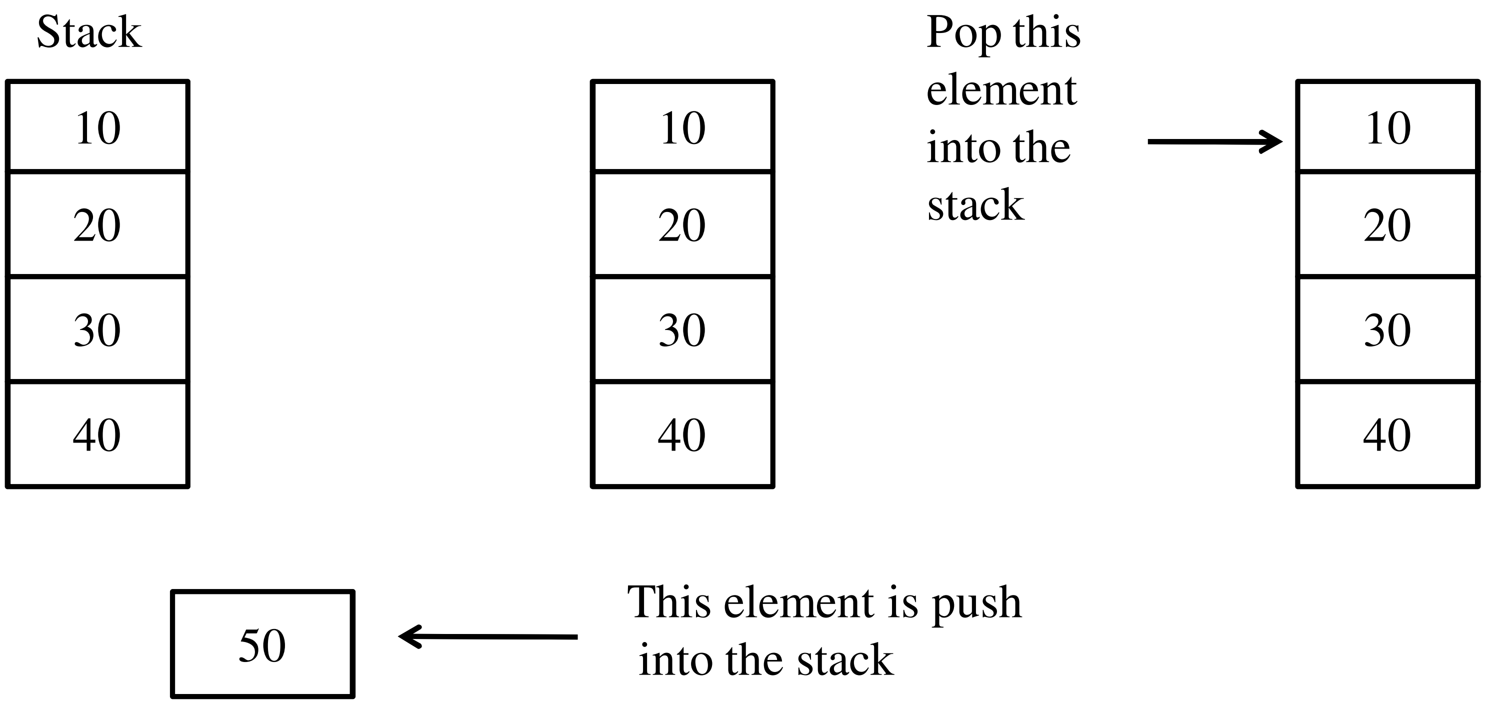
Contents of Stack

- Introduction to stack
- Operations on the stack
- Memory Representation

Introduction

- Stack is linear data structure of variable size.
- In array insertion and deletion can occur at any place but in stack it can occur at only one end known a **Top**
- Insertion is known as **Push** and deletion is known as **Pop**
- Stack is known as **Last In First Out (LIFO)** List.
- LIFO means the last item added to the stack will be the first item to be removed from the stack.

Example of Stack



Stack

10

20

30

40

10

20

30

40

Pop this element into the stack

10

20

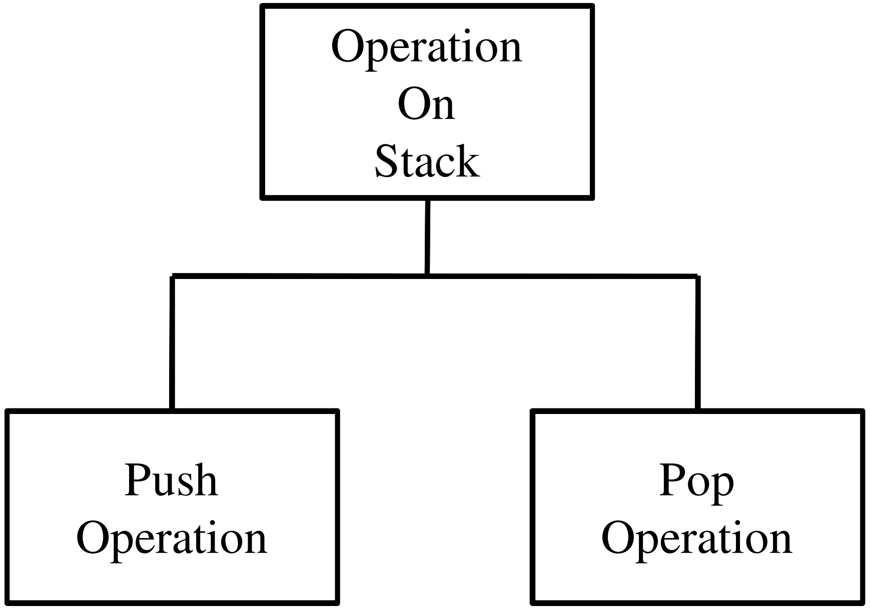
30

40

50

This element is push into the stack

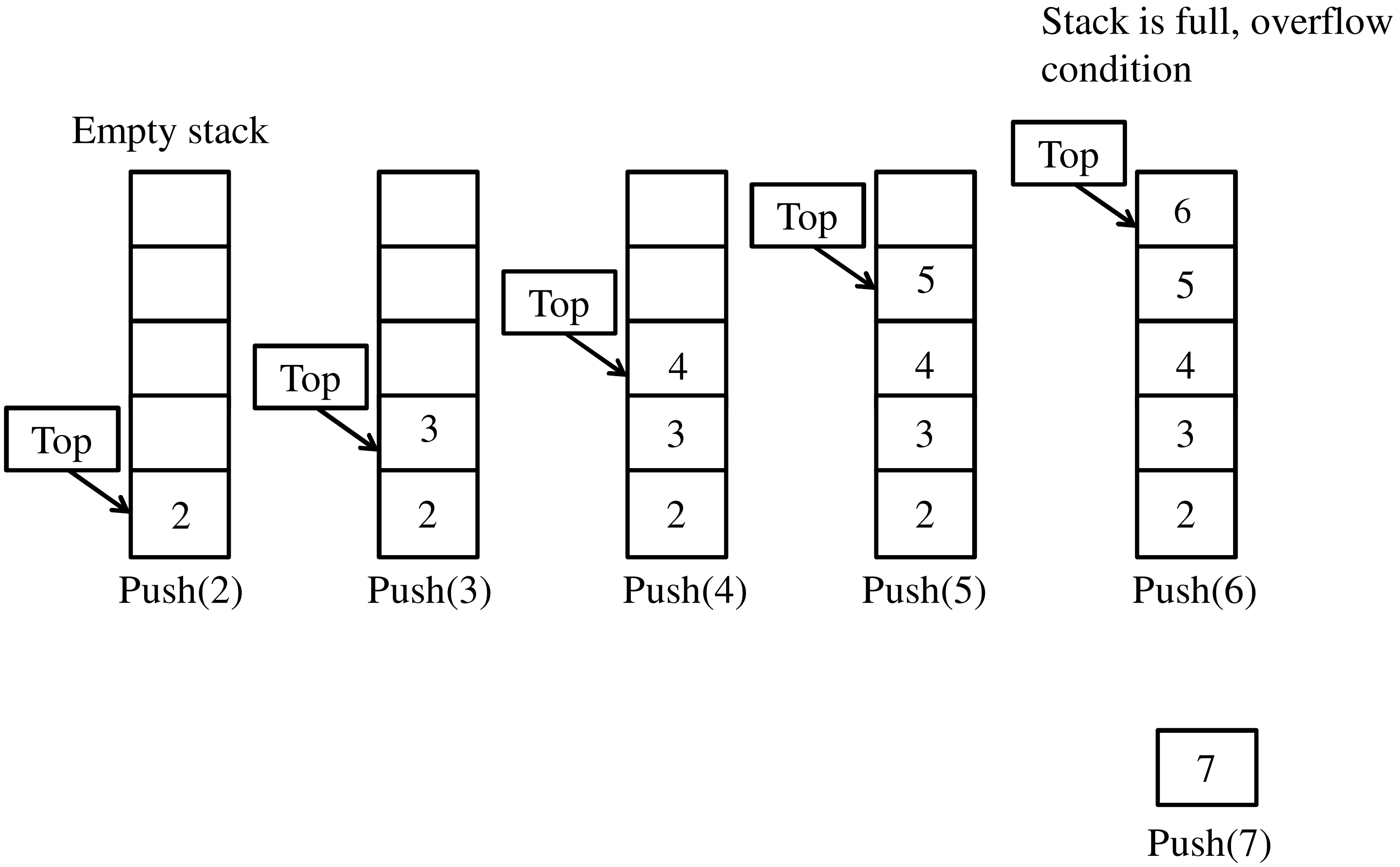
Operations On Stack



Push Operation

- **Push** operation refers to insertion of a new element into the stack.
- It will be inserted at the top of the stack.
- We can perform push operation only when stack is not full i.e. stack has space for new element.
- When the stack is full this condition is known as **overflow**.

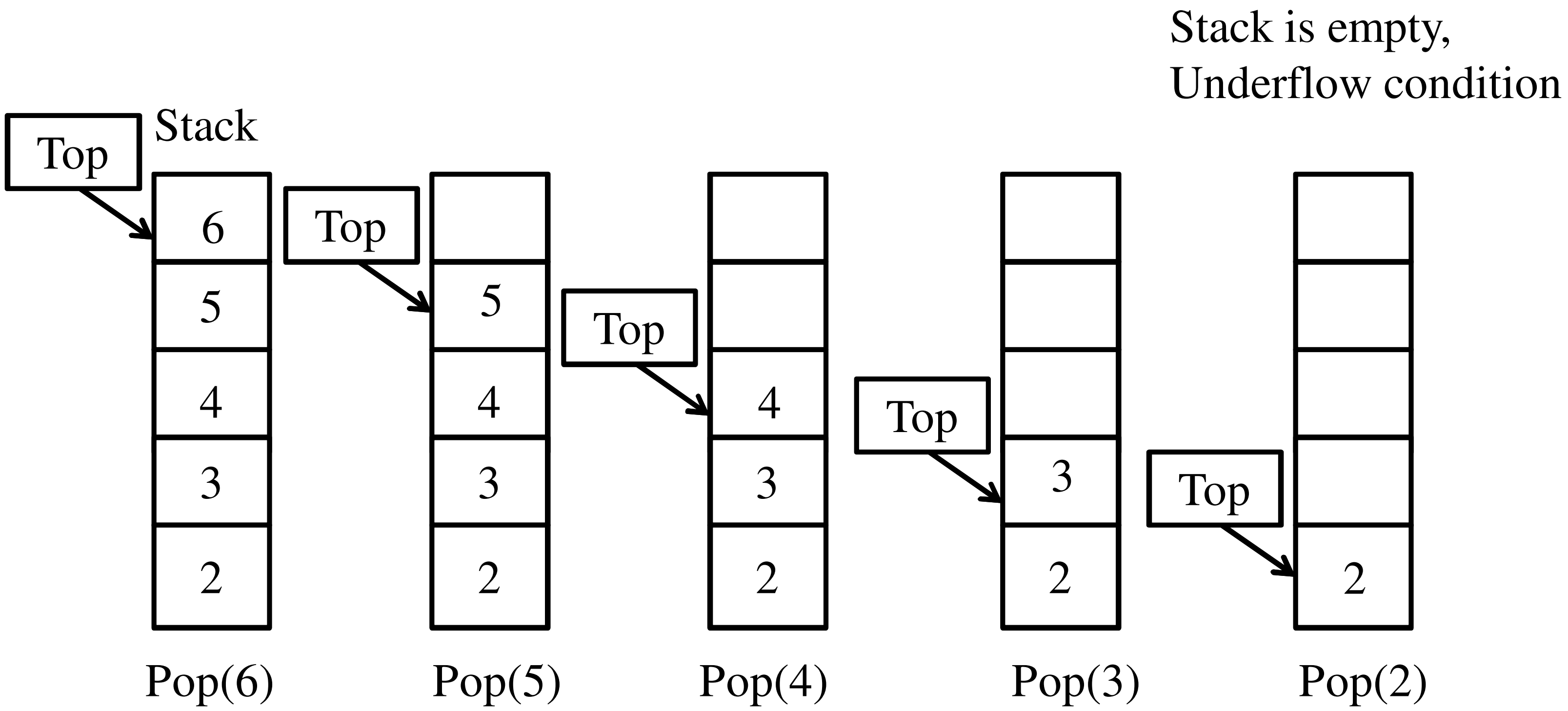
Example of Push Operation



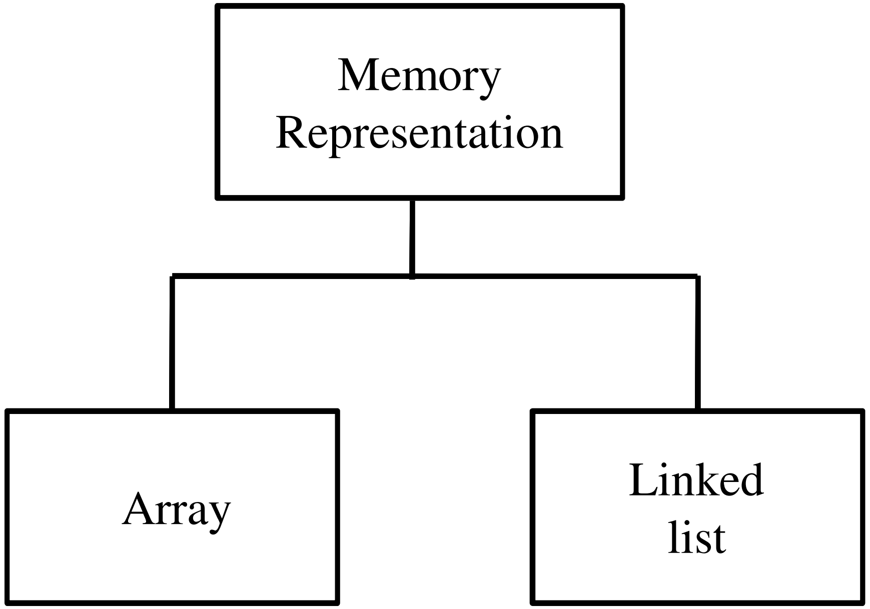
Pop Operation

- **Pop** operation refers to removal of an element form the top of the stack.
- We can perform pop operation only when stack is not empty.
- When the stack is empty and we are attempting to remove element from the stack this condition is known as **underflow.**

Example of Pop Operation



Memory Representation of Stack



Memory Representation

Array

Linked list

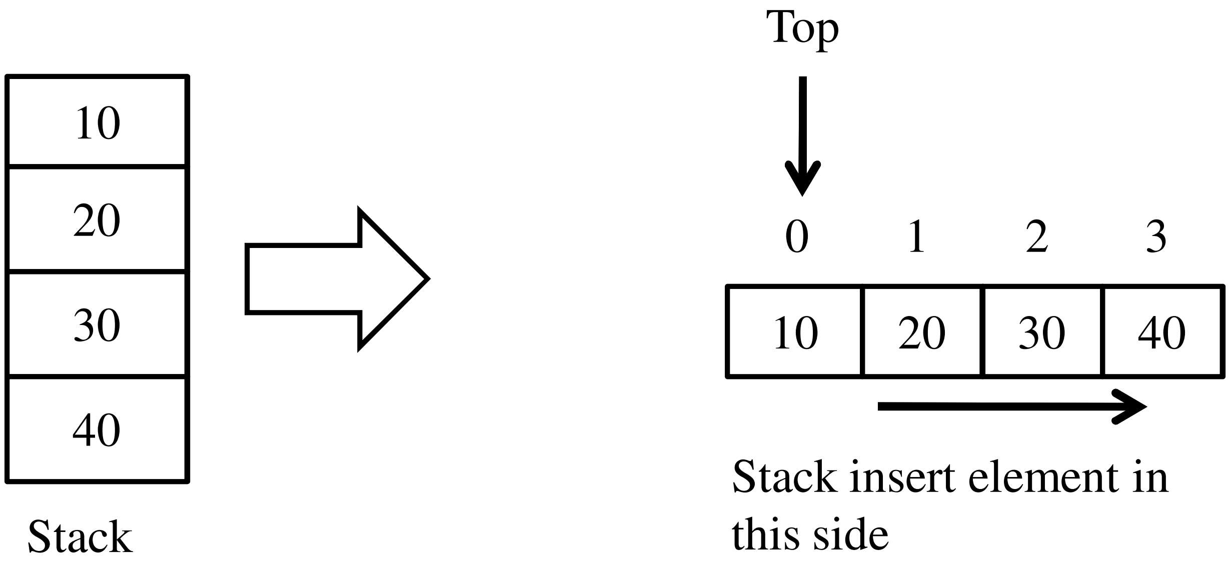
Array Representation of Stack

- It is simplest form of stack representation. But an array puts certain restrictions while representing the stack:
- The stack must contain **homogeneous** data elements.
- One must **specify the upper bound** of the array i.e. **maximum size** of the stack must be defined before implementing it.

Continued

- While implementing a stack using an array a variable **Top** is used to hold the index of stack's topmost element. Initially the stack is empty and Top is zero its value increases by one when values are added into the stack and decreases by one when element is removed from the stack.
- **Max** represents the maximum size of the stack.

Array Representation of Stack



10

20

30

40

Stack

Top

0 1 2 3

10 20 30 40

Stack insert element in this side

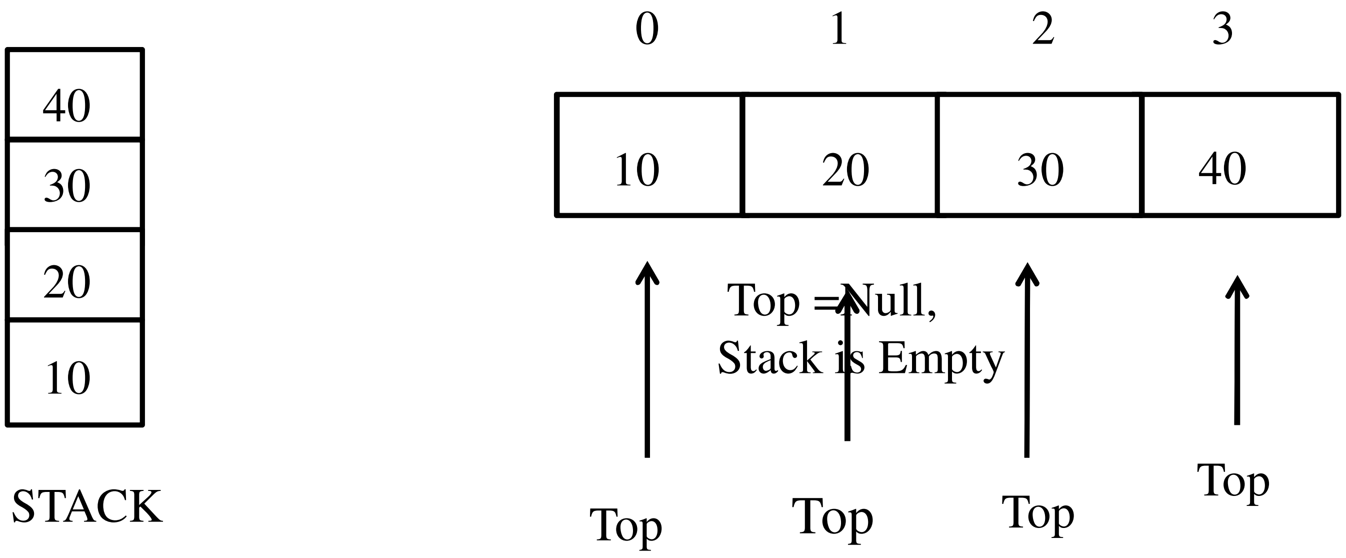
Algorithm: Push Operation

- Insert a new element **‘Data’** at the top of the stack represented by **’S’** of size **‘Max’** with a stack index variable **‘Top’** pointing to the topmost element of the stack.
- Step 1: If **Top** = **Max** Then

```
                    Print:”Stack is Full, Overflow Condition”
                    Exit
                    [End If]
```

- Step 2: Set **Top** = **Top** + 1
- Step 3: Set **S[Top]** = **Data**
- Step 4: Exit

Example of Push Operation



Algorithm: Pop Operation

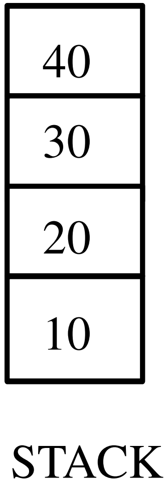
- Delete an element from the stack represented by an array **‘S’** and return the element **‘Data’** which is at the top of the stack.
- Step 1: If **Top** = **Null** Then

  Print:”Stack is empty, Underflow Condition”

  Exit

  [End If]
- Step 2: Set **Data** = **S[Top]**
- Step 3: Set **Top** = **Top - 1**
- Step 4: Exit

Example of Pop Operation



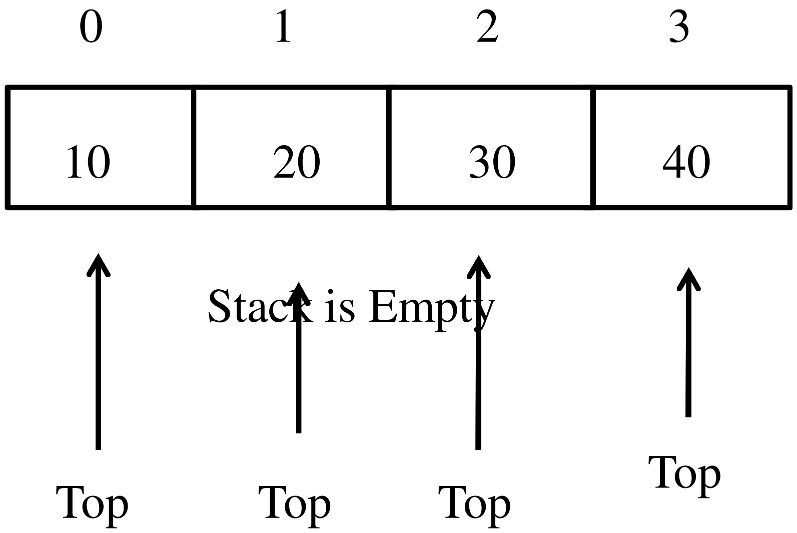
40

30

20

10

STACK



0	1	2	3
10	20	30	40

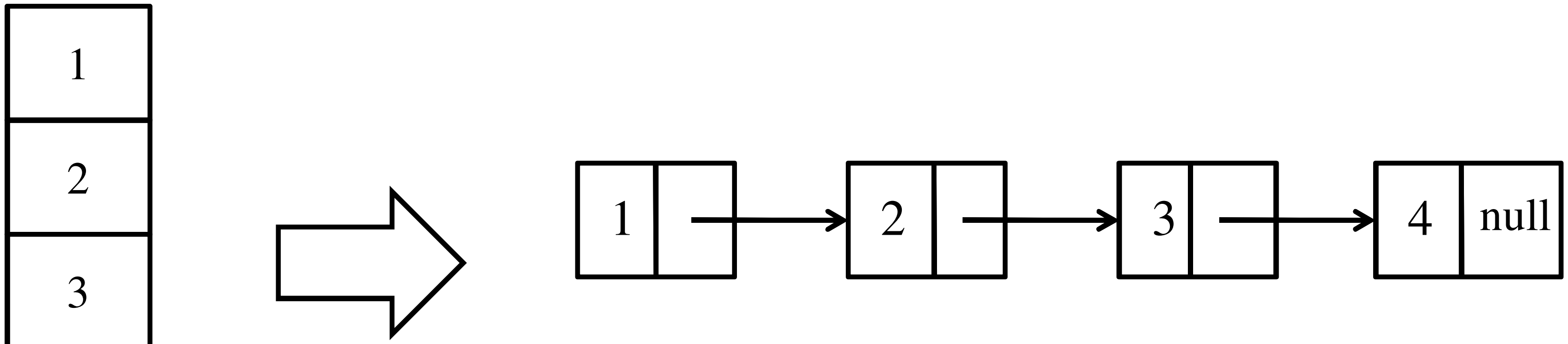
Stack is Empty

Top Top Top Top

Linked List Representation of Stack

- Stack can also implementing by using linked list. It will eliminate the drawbacks of implementation of stack using array.
- There is no need to know advance about the size of the stack.

Example of Linked List



Linked list

Stack

Algorithm: Push Operation

- Insert a new element **‘Data’** at the top of the stack represented by the linked list with a stack pointer variable **‘Top’** pointing to the topmost element of the stack.
- Following algorithms explain the push operation on the stack when it is represented using linked

  list.

Continued

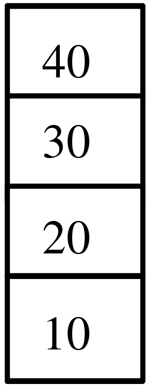
- Step 1:If Free = **Null** Then

  Print:“free space not available,Overflow Condition

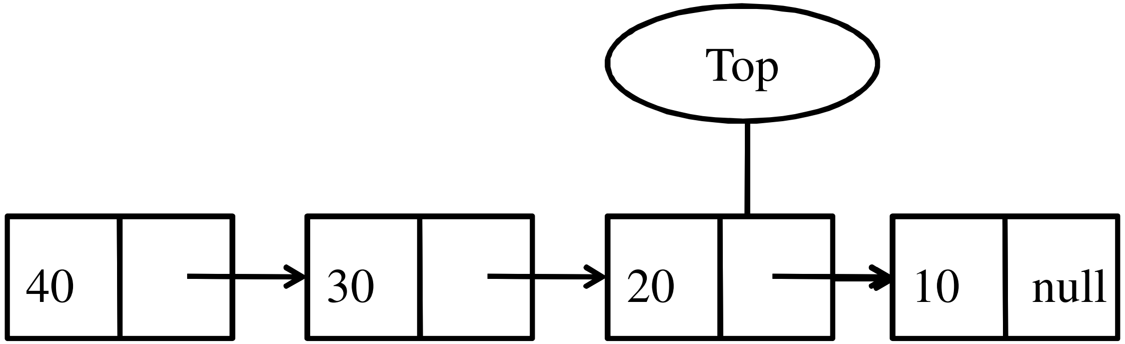
  Exit

  [End If]
- Step 2:Set **New** = **Free** And **Free** = **Free** → **Next**
- Step 3: Set **New** = **Info Data** And **New** →**Next** = **Top**
- Step 4: Set **Top** = **New**
- Step 5: Exit

Push Operation



STACK



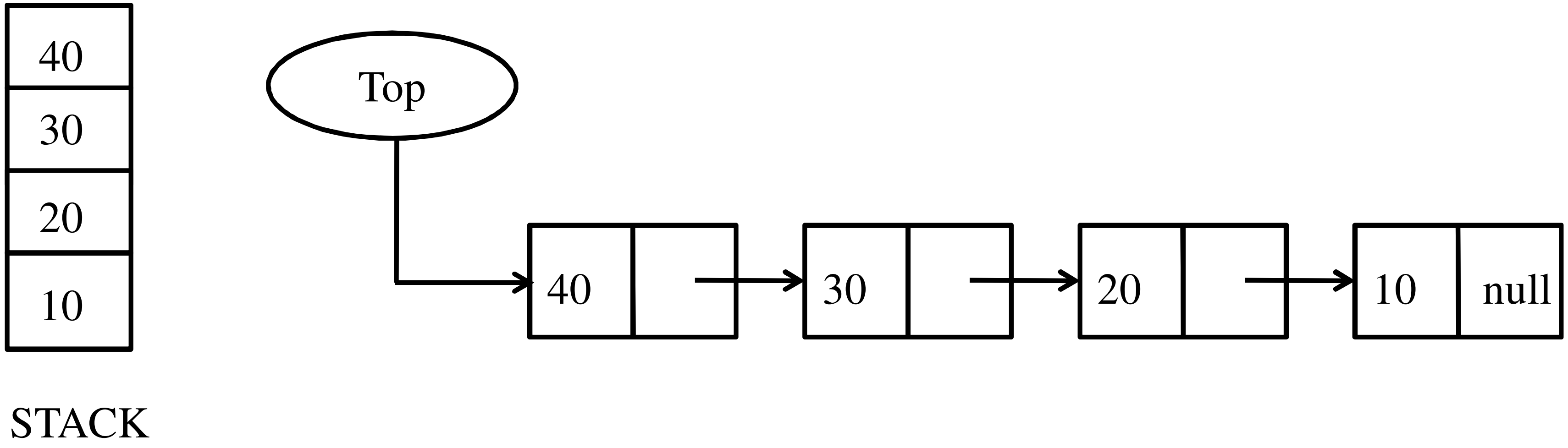
Algorithm: Pop Operation

- Delete an element from the stack represented by the linked list returns the element **'Data'** which is at the top of the stack.
- Following algorithms explain the push operation on the stack when it is represented using linked list.

Continued

```
• Step 1: If Top = Null Then
                 Print: “Stack is empty,Underflow Condition”
                  Exit
                  [End If]
• Step 2:Set Data=Top → Info And Temp = Top
• Step 3: Set Top = Top → Next
• Step 4: Set Temp → Next = FreeAnd Free = Temp
• Step 5: Exit
```

Pop Operation


40
30
20
10
STACK
Top
40
30
20
10
null