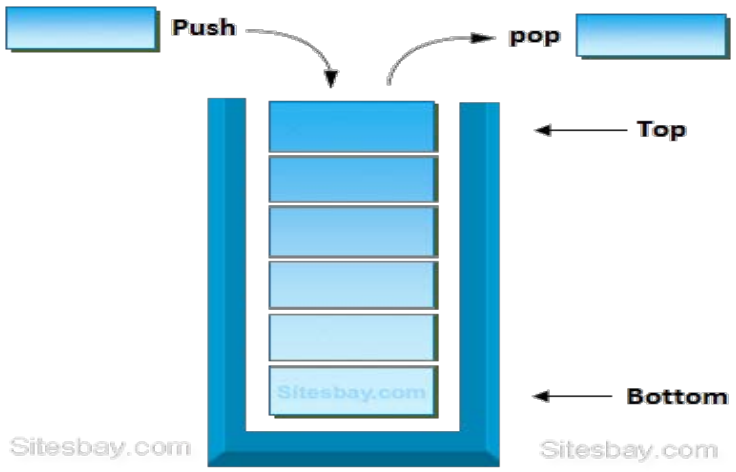
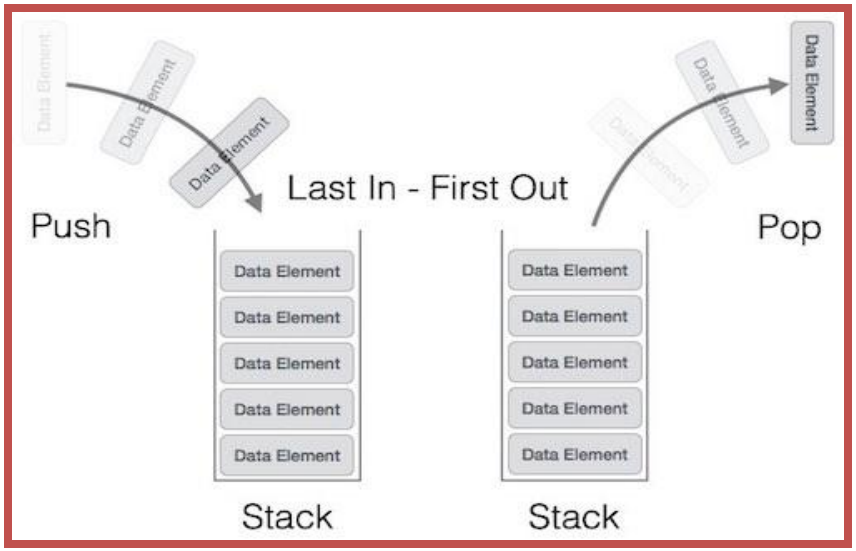
STACK

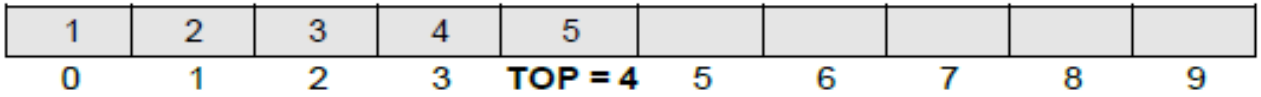
- A stack is a linear data structure, which works on the principle of LIFO/FILO.
- Stack is closed at one end and opened at other end.
- The elements can be added and removed from the stack only at the top(opened end).
- Basic Operations performed on stack are:
  - **PUSH-Insertion**
  - **POP-Deletion**
  - **PEEK-Returns top most element**
- Stack maintains a variable called TOP,
  if TOP=-1 then the stack is empty.





Stack…

- When an element is pushed into the stack TOP value is incremented by 1 i,e TOP++.
- When an element is popped from the stack TOP value is decremented by1 I,e TOP--.
- Stack can be implemented in two ways:
  I. Stack Implementation using Arrays
  II. Stack Implementation using Linked List
- **Stack Implementation using Arrays:**
  - Stack can be represented using array as

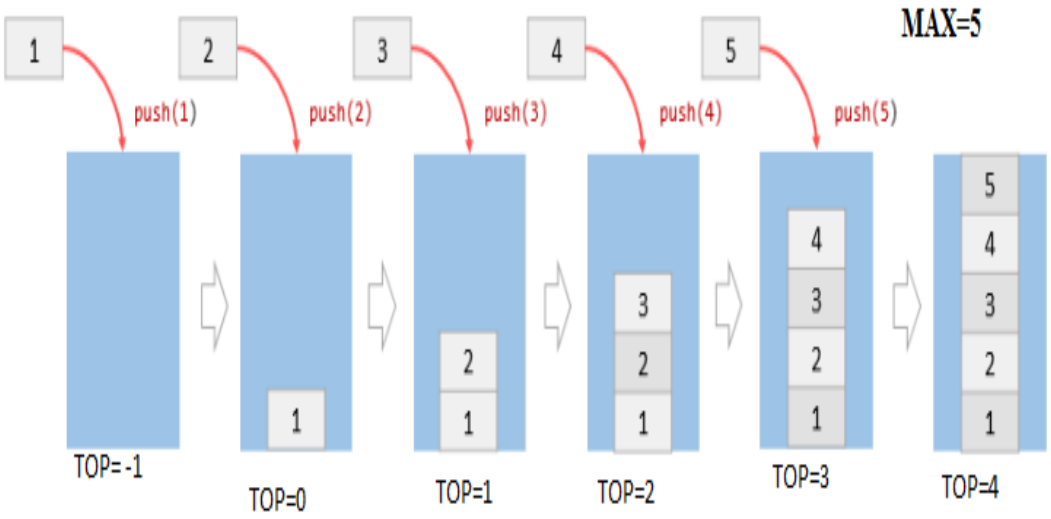


  - Here, the maximum size(MAX) of the stack is 10 i,e it can hold up to 10 elements.
- Basic Operations performed on stack implemented using arrays are:
  - PUSH
  - POP
  - PEEK

Stack Implementation using Array…

➢ PUSH:

- ➢The push operation is used to insert an element into the stack. The new element is added at the topmost position of the stack.
- ➢While inserting the value, we must first check if TOP=MAX–1, because if that is the case, then the stack is full and no more insertions can be done .This condition is called as **stack overflow .**
- ➢If the above condition fails then we can perform the insertion into the stack , then TOP value is incremented by 1.

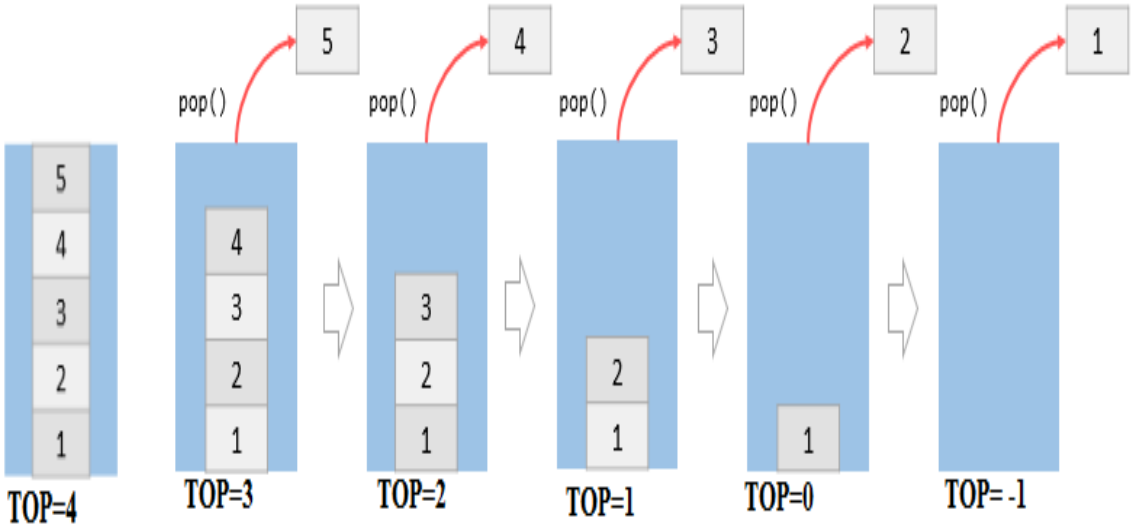



```
Step 1: IF TOP = MAX-1
            PRINT "OVERFLOW"
            Goto Step 4
        [END OF IF]
Step 2: SET TOP = TOP + 1
Step 3: SET STACK[TOP] = VALUE
Step 4: END
```

Stack Implementation using Array…

➢ POP:

- ➢The POP operation is used to delete an element from top of the stack.
- ➢Before deleting an element we have to check whether TOP=-1 ,if that is the case POP operation is not possible . This condition is called as **stack underflow.**
- ➢Other wise we can pop an element from the stack by decrementing the TOP by 1.



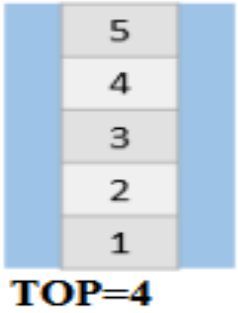

```
Step 1: IF TOP = NULL
            PRINT "UNDERFLOW"
            Goto Step 4
        [END OF IF]
Step 2: SET VAL = STACK[TOP]
Step 3: SET TOP = TOP - 1
Step 4: END
```

Stack Implementation using Array…

➢ PEEK:

➢PEEK is an operation that returns the value of the topmost element of the stack without deleting it from the stack.

➢If we perform PEEK operation on the stack ,**it will first check whether the stack is empty or not,** if stack is not empty then it returns the top most element in the stack i,e 5 other wise it has to print stack is empty.

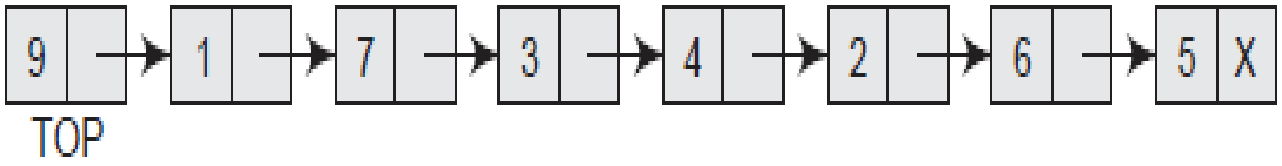



```
Step 1: IF TOP = NULL
            PRINT "STACK IS EMPTY"
            Goto Step 3
Step 2: RETURN STACK[TOP]
Step 3: END
```

Stack Implementation using Linked List

➢ Stack Implementation using Linked List:

- ➢In a linked stack, every node has two parts—one that stores data and another that stores the address of the next node.
- ➢ if the stack size cannot be determined in advance, then linked representation is used.
- ➢The START pointer of the linked list is used as TOP.
- ➢All insertions and deletions are done at the node pointed by TOP.
- ➢If TOP = NULL, then it indicates that the stack is empty.
- ➢The linked representation of a stack is :



- ➢A linked stack supports all the three stack operations, that is, push, pop, and peek.

Stack Implementation using Linked List….

➢Operations on a Linked Stack :

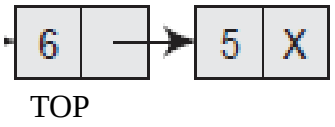
➢PUSH:

➢The push operation is used to insert an element into the stack. The new element is added at the topmost position of the stack.

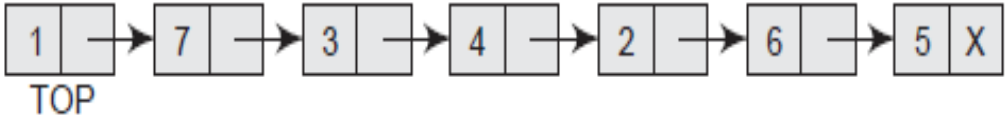
Push 5


```
Step 1: Allocate memory for the new
        node and name it as NEW_NODE
Step 2: SET NEW_NODE -> DATA = VAL
Step 3: IF TOP = NULL
            SET NEW_NODE -> NEXT = NULL
            SET TOP = NEW_NODE
        ELSE
            SET NEW_NODE -> NEXT = TOP
            SET TOP = NEW_NODE
        [END OF IF]
Step 4: END
```

Push 6



similarly Push 2,4,3,7,1 we get

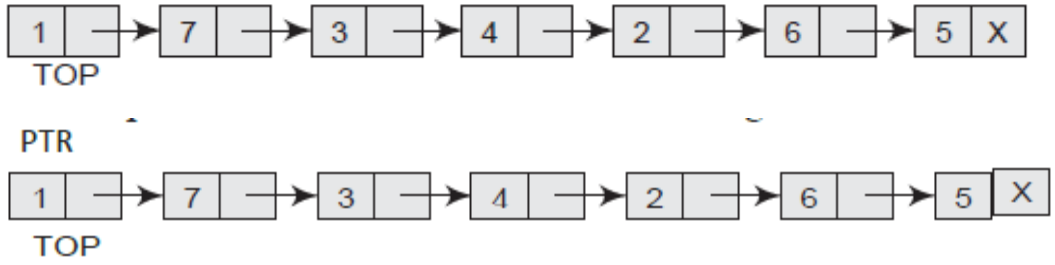


Stack Implementation using Linked List….

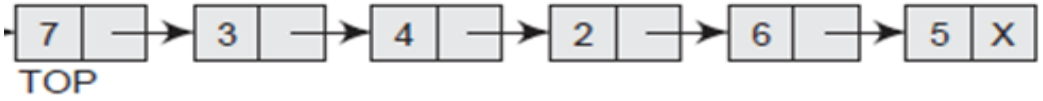
➢POP

- ➢The pop operation is used to delete the topmost element from a stack.
- ➢Before deleting the element, we must first check if TOP=NULL (stack empty).
- ➢ If an attempt is made to delete a value from a stack that is already empty, an **Underflow message** is printed.
- ➢If TOP!=NULL , then top most element is deleted from the stack i,e in this case element 1 is deleted from the stack.

```
Step 1: IF TOP = NULL
            PRINT "UNDERFLOW"
            Goto Step 5
        [END OF IF]
Step 2: SET PTR = TOP
Step 3: SET TOP = TOP -> NEXT
Step 4: FREE PTR
Step 5: END
```

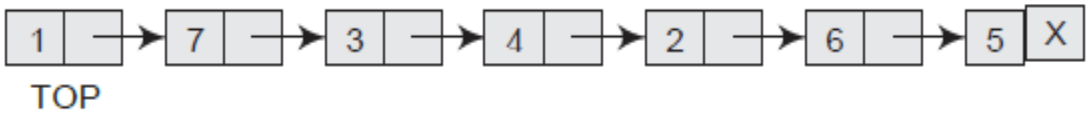


Now the new top element is 7.



Stack Implementation using Linked List….

- **PEEK**
  - It is used to print the top element of stack.
  - If TOP=NULL returns stack is empty, else prints the top element in the linked stack.



  - If Peek operation is performed on the above stack ,it will return element 1.

```
Step 1:if(top==NULL)
            return “stack is empty”;
Step 2: else
             return top ->data;
```

Applications of Stack

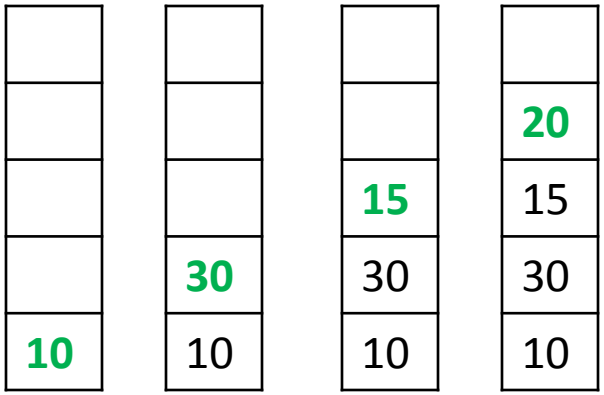
➢**The applications of the stack are**

1. **Reversing a list**

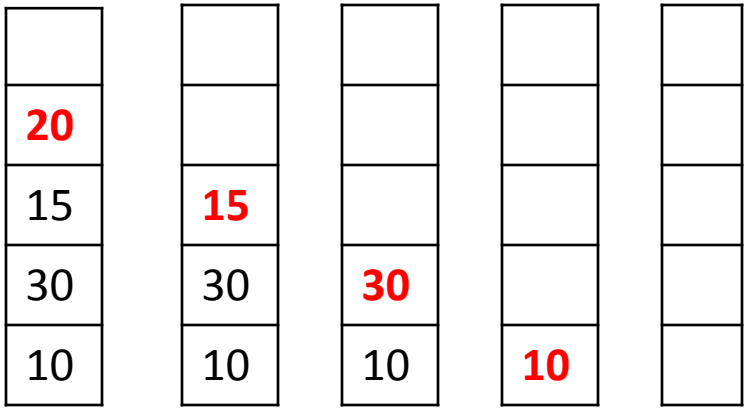
   A list of numbers can be reversed by reading each number from an array starting from the first index and pushing it on a stack. Once all the numbers have been read, the numbers can be popped one at a time and then stored in the array starting from the first index.

   Example: **10,30,15,20**

Push:



Pop:



Reverse List: 20 15 30 10