Bharathidasan University

Centre for Differently Abled Persons
Tiruchirappalli - 620024.

- Programme Name : Bachelor of Computer Applications
- Course Code : 20UCA6CC10
- Course Title : Data Structures
- Unit : Unit II
- Compiled by :Dr. M. Prabavathy
  Associate Professor
  Ms. Patric Matharasi
  Guest Faculty

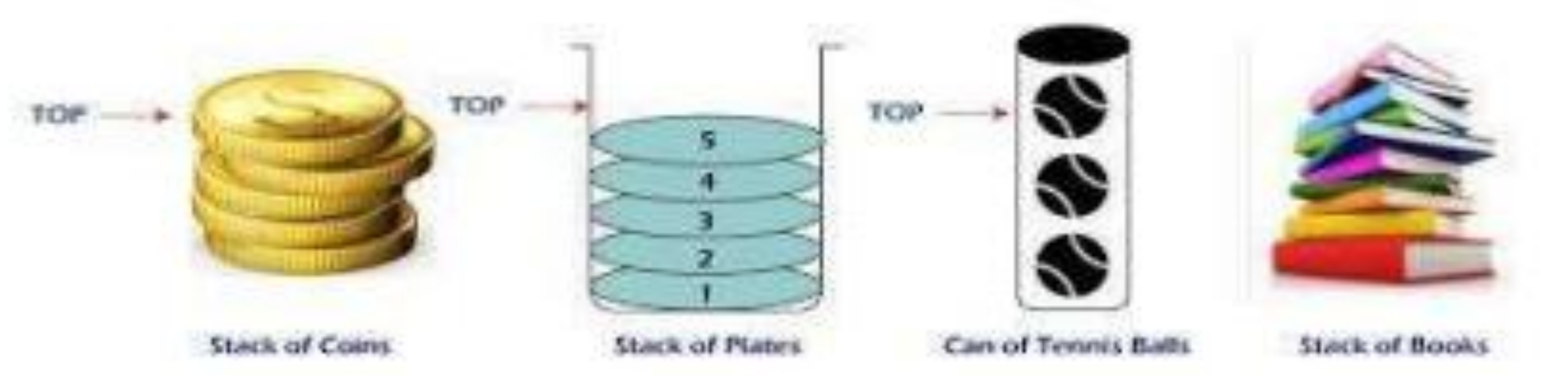
Unit –II

Stack and Its Operation

Stack

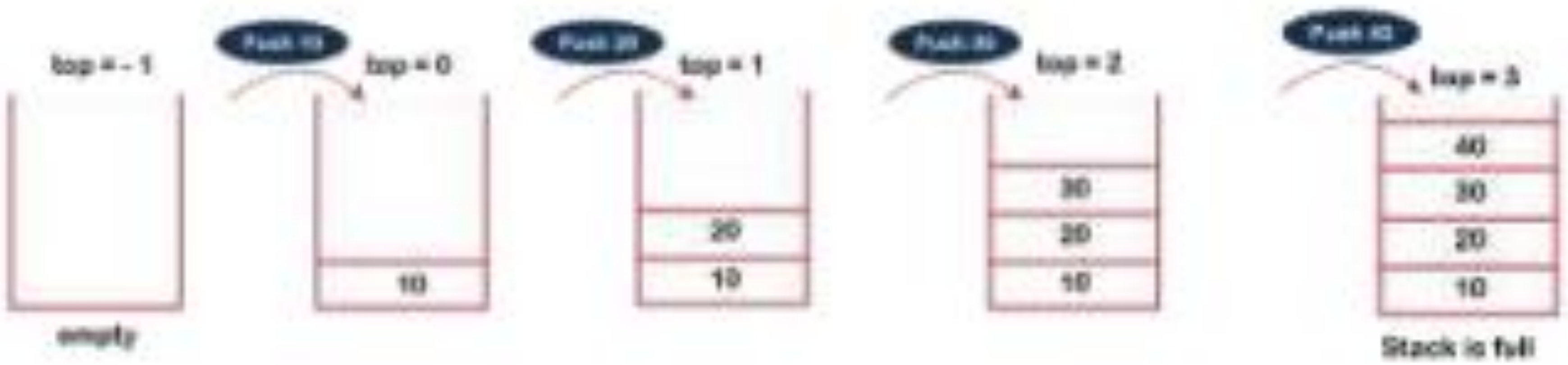
- A Stack is a linear data structure that follows the LIFO (Last–In–First–Out) principle.
- Stack has one end, whereas the Queue has two ends (front and rear).
- It contains only one pointer top pointer pointing to the topmost element of the stack.

- Whenever an element is added in the stack, it is added on the top of the stack, and the element can be deleted only from the stack.
- In other words, a stack can be defined as a container in which insertion and deletion can be done from the one end known as the top of the stack.

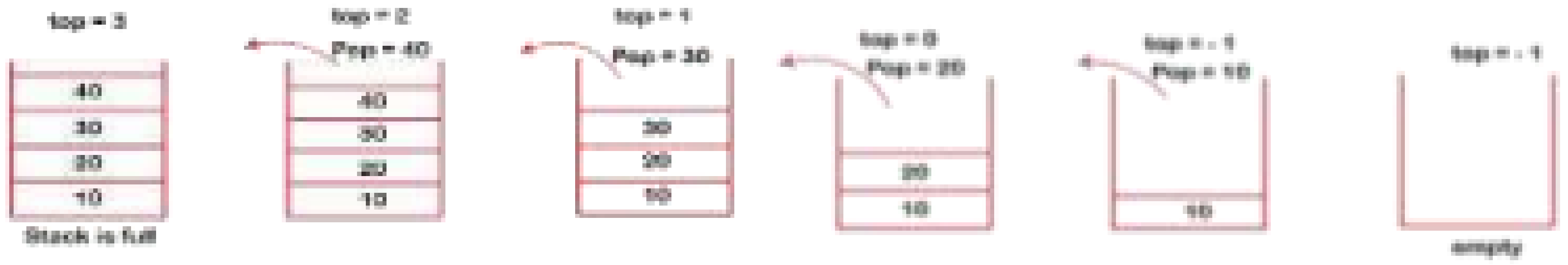


Operations implemented on the stack:

- push(): When we insert an element in a stack then the operation is known as a push. If the stack is full then the overflow condition occurs.



- **pop():** When we delete an element from the stack, the operation is known as a pop. If the stack is empty means that no element exists in the stack, this state is known as an underflow state.
- **isEmpty():** It determines whether the stack is empty or not.
- **isFull():** It determines whether the stack is full or not.



Array implementation of Stack

- In array implementation, the stack is formed by using the array.
- All the operations regarding the stack are performed using arrays.

Adding an element onto the stack (push operation)

1. Increment the variable Top so that it can now referred to the next memory location.

2. Add element at the position of incremented top. This is referred to as adding new element at the top of the stack.

Algorithm:

```
begin
 if top = n then stack full
 top = top + 1
 stack (top) : = item;
end
```

Deletion of an element from a stack (Pop operation)

1. The value of the variable top will be incremented by 1 whenever an item is deleted

from the stack.

2. The top most element of the stack is stored in another variable and then the top is decremented by 1.

3. The operation returns the deleted value that was stored in another variable as the result

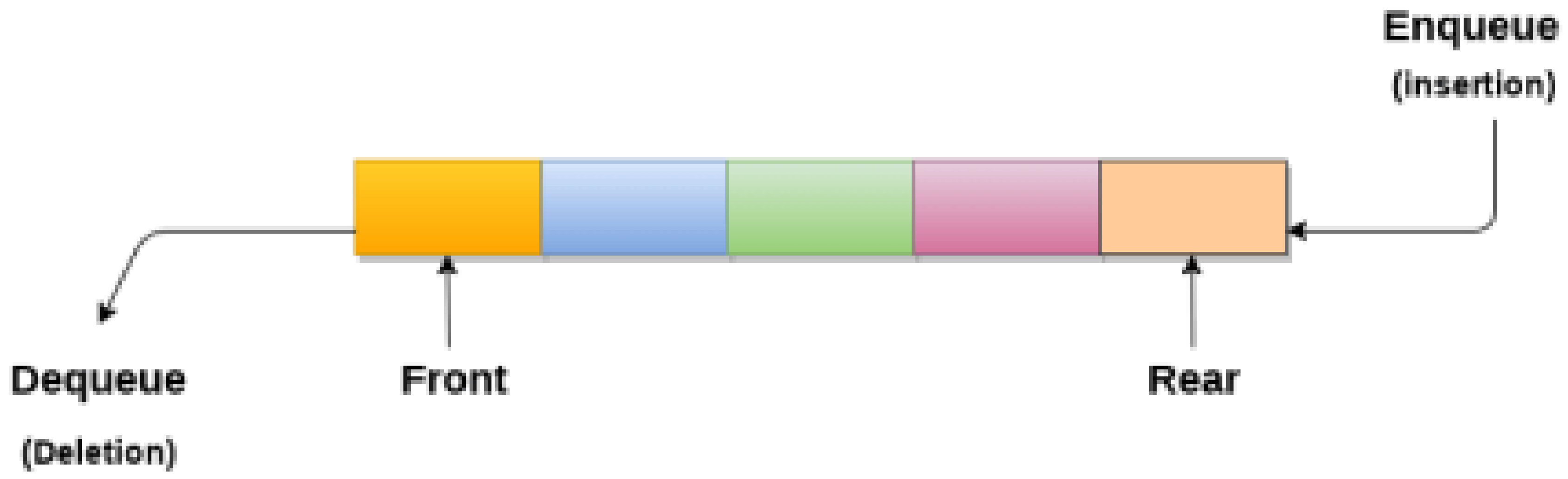
Algorithm:

```
begin
if top = 0 then stack empty
item := stack(top)
top = top – 1
end
```

Queue

- A queue can be defined as an ordered list which enables insert operations to be performed at one end called **REAR** and delete operations to be performed at another end called **FRONT.**
- Queue is referred to be as First In First Out list.
- For example, people waiting in line for a rail ticket form a queue.

QUEUE

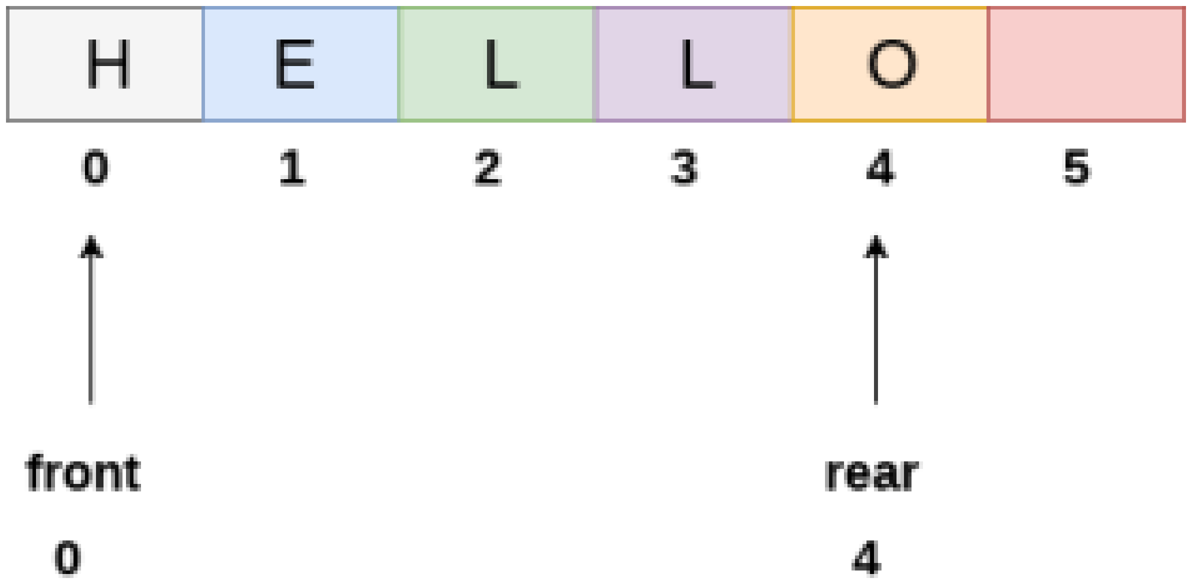


Array representation of Queue

- We can easily represent queue by using linear arrays.
- There are two variables i.e. front and rear, that are implemented in the case of every queue.
- Front and rear variables point to the position from where insertions and deletions are performed in a queue.

- Initially, the value of front and queue is -1 which represents an empty queue.
- Array representation of a queue containing 5 elements along with the respective

values of front and rear, is shown in the following figure.

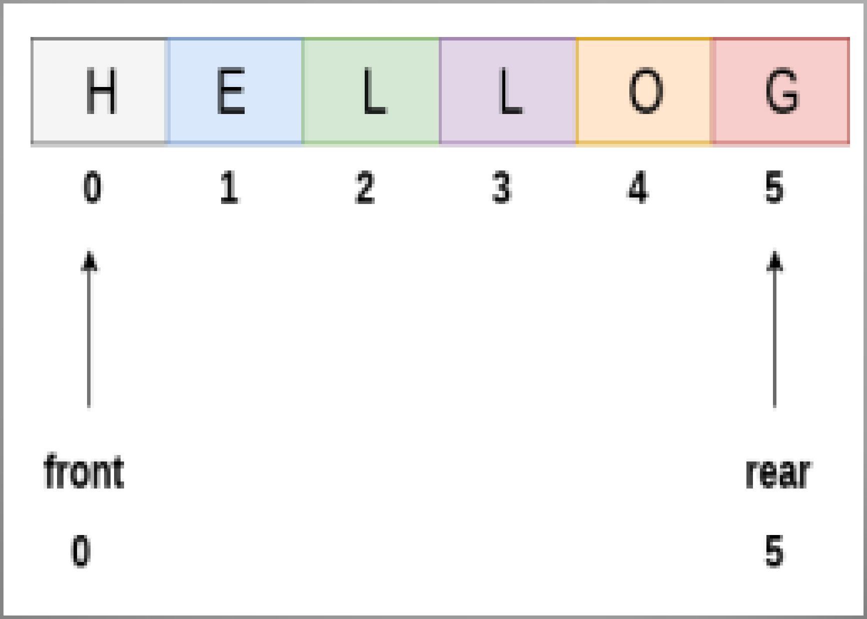


Insertion to the Queue (Enqueue)

- The value of rear increases by one every time an insertion is performed in the queue.
- After inserting an element into the queue shown in the above figure, the queue will look something like following.
- The value of rear will become 5 while the value of front remains same

Algorithm

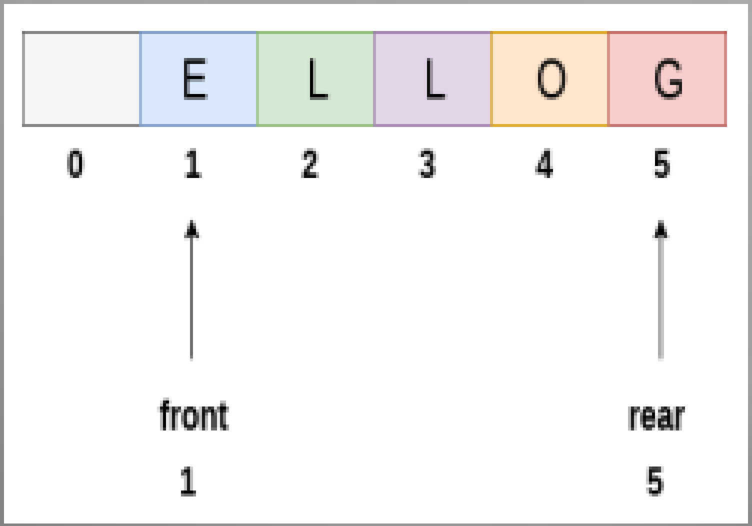
```
Step 1: IF REAR = MAX - 1
Write OVERFLOW
Go to step
[END OF IF]
Step 2: IF FRONT = -1 and REAR = -1
SET FRONT = REAR = 0
ELSE
SET REAR = REAR + 1
[END OF IF]
Step 3: Set QUEUE[REAR] = NUM
Step 4: EXIT
```



Deletion from the Queue (Dequeue)

- After deleting an element, the value of front will increase from -1 to 0.
- However, the queue will look something like following.

```
Step 1: IF FRONT = -1 or FRONT > REAR
            Write UNDERFLOW
        ELSE
            SET VAL = QUEUE[FRONT]
            SET FRONT = FRONT + 1
        [END OF IF]
Step 2: EXIT
```



Evaluation of Expressions

- An arithmetic expression can be written in three different but equivalent notations, without changing the essence or output of an expression.

These notations are

- Infix Notation
- Prefix (Polish) Notation
- Postfix (Reverse-Polish) Notation

These notations are named as how they use operator in expression.

Infix Notation

- We write expression in infix notation, e.g. a − b + c, where operators are used in between operands.
- It is easy for us humans to read, write, and speak in infix notation but the same does not go well with computing devices.

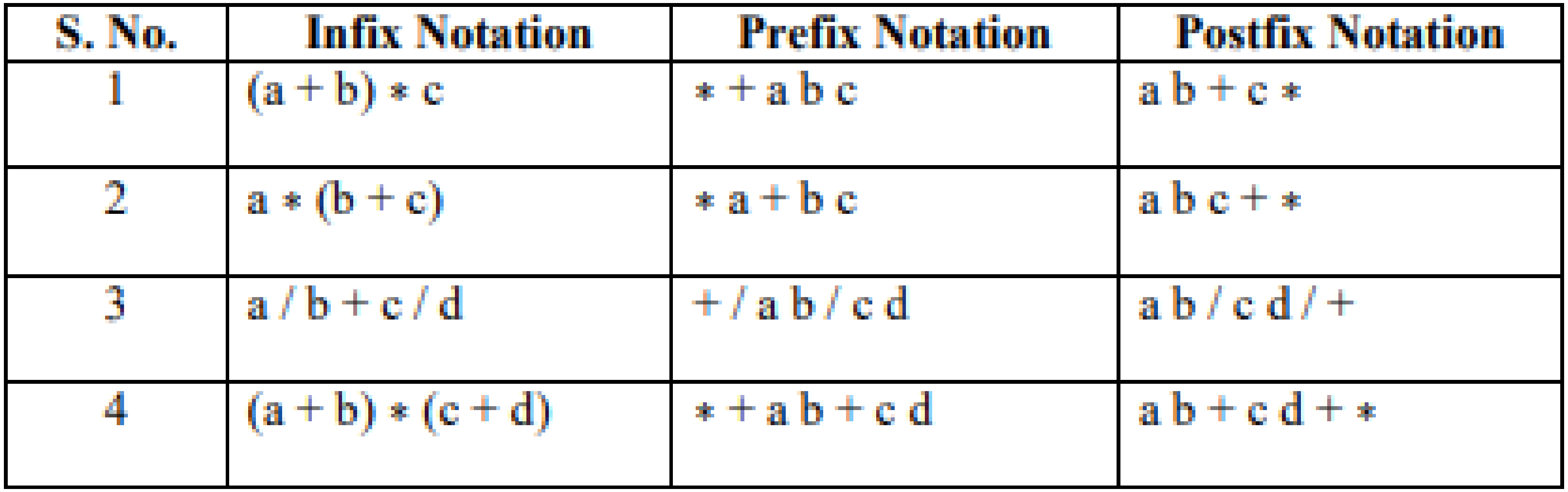
Prefix Notation

- In this notation, operator is prefixed to operands, i.e. operator is written ahead of operands.
- For example, +ab. This is equivalent to its infix notation a + b.
- Prefix notation is also known as Polish Notation.

Postfix Notation

- This notation style is known as Reversed Polish Notation.
- In this notation style, the operator is post fixed to the operands i.e., the operator is
- written after the operands.
- For example, ab+. This is equivalent to its infix notation a + b.

The following table briefly tries to show the difference in all three notations :

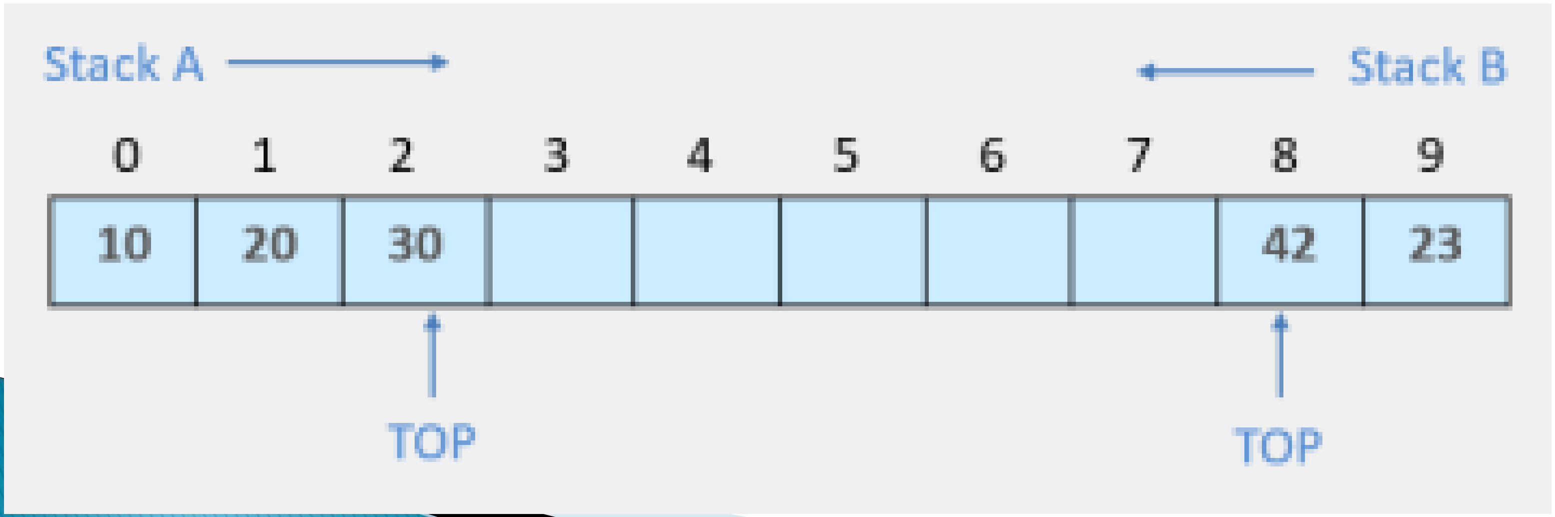


S. No.	Infix Notation	Prefix Notation	Postfix Notation
1	(a + b) * c	* + a b c	a b + c *
2	a * (b + c)	* a + b c	a b c + *
3	a / b + c / d	+ / a b / c d	a b / c d / +
4	(a + b) * (c + d)	* + a b + c d	a b + c d + *

Multiple Stacks and Queues

- A single stack is sometimes not sufficient to store a large amount of data.
- To overcome this problem, we can use multiple stack.
- For this, we have used a single array having more than one stack.
- The array is divided for multiple stacks.

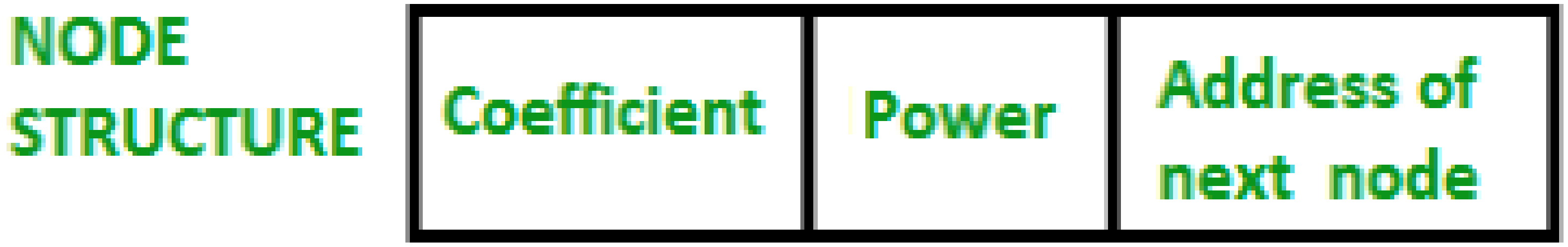
- Suppose there is an array STACK[n] divided into two stack STACK A and STACK B, where n = 10.
- STACK A expands from the left to the right, i.e., from 0th element.
- STACK B expands from the right to the left, i.e., from 10th element.
- The combined size of both STACK A and STACK B never exceeds 10.



Polynomial addition

- Polynomials are the expressions that contain the number of terms with non-zero exponents and coefficients.
- Linked List is widely used for Representing and Manipulating the polynomials.
- Such as the linked representation of polynomials, each term considered as a node, therefore these node contains three fields.

- Coefficient Field – The coefficient field holds the value of the coefficient of a term o Exponent Field – The Exponent field contains the exponent value of the term
- Link Field – The linked field contains the address of the next term in the polynomial



Example 1

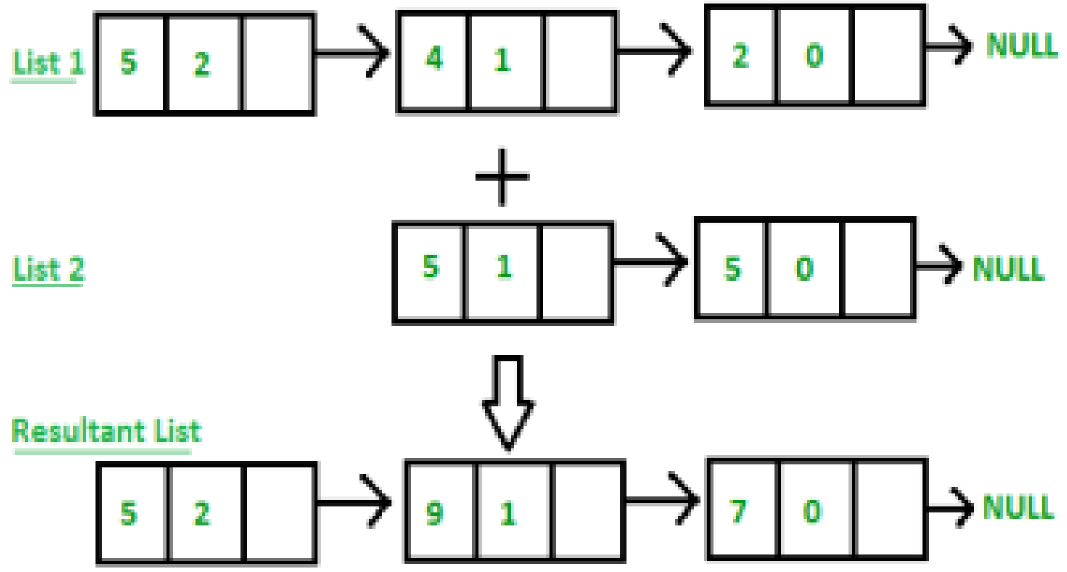
Input:

1st number = $5x^2 + 4x^1 + 2x^0$

2nd number = $5x^1 + 5x^0$

Output:

$5x^2 + 9x^1 + 7x^0$



Example 2:

Input:

1st number = $5x^3 + 4x^2 + 2x^0$

2nd number = $5x^1 - 5x^0$

Output:

$5x^3 + 4x^2 + 5x^1 - 3x^0$