ডাটা স্ট্রাকচার:

স্ট্যাক এবং কিউ (Stack and Queue)

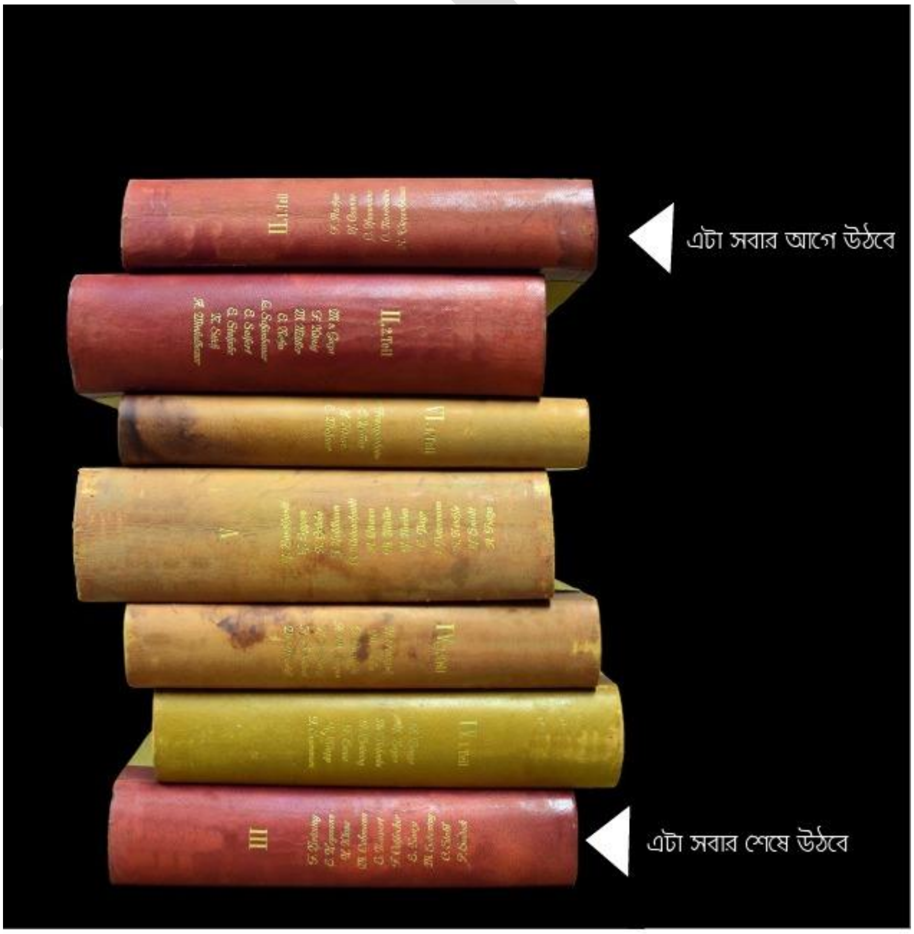
স্ট্যাক এবং কিউ (Stack and Queue) বহুল ব্যবহৃত ডাটা স্ট্রাকচার (Data structure) গুলোর মধ্যে অন্যতম। যখন এমন কোন সিচুয়েশন আসে যেখানে আমাদেরকে ডাটার পরিমাণ নির্দিষ্ট করা হয় না, আবার ডেটা রাখা এবং তুলে আনার অপারেশন এ করতে হয় তখন আমরা স্ট্যাক এবং কিউ ব্যবহার করি। স্ট্যাক এবং কিউ দুইটি আলাদা ডাটা স্ট্রাকচার হলেও আমি একই লিখায় দুইটি নিয়েই আলোচনা করবো। কারণ স্ট্যাক (Stack) আর কিউ (queue) এর ব্যাসিক প্রায় একই। একটি পারলে আরেকটি সহজেই বুঝা যায়।

স্ট্যাক ডাটা স্ট্রাকচারের সংজ্ঞা- Stack definition: Stack একটি লিনিয়ার ডেটা স্ট্রাকচার যেখানে **LIFO** ক্রম অনুসারে ডেটা গুলোর উপর অপারেশন করা হয়।

স্ট্যাক কি -What is stack?

স্ট্যাক বুঝার জন্য আমরা একটা উদাহরণ দিতে পারি। ধরা যাক, আপনি একটি টেবিলে বই একটার উপর আরেকটি রেখেছেন। এখন আপনাকে সবার নিচের বইটি বের করতে হবে। এর জন্য কিন্তু আপনি সবার নিচের বই শুরুতেই বের করতে পারবেন না। কারণ বই এর স্তূপ বড় হলে উপরের বই পরে গিয়ে এলোমেলো হয়ে যাবে।

এর জন্য যা করতে হবে, শুরুতে সবার উপরের বই তুলতে হবে। তারপর উপর থেকে দ্বিতীয় বইটি তুলতে হবে। এভাবে সবার শেষে পৌছাতে হবে। এখন যদি লক্ষ করেন তবে দেখুন, সবার উপরের বইটি কিন্তু আপনি সবার শেষে রেখেছেলেন। উপর থেকে দ্বিতীয় বইটি আপনি শেষ ধাপের আগের ধাপে রেখেছিলেন। এভাবে সবার নিচের বইটি আপনি শুরুতে রেখেছিলেন।

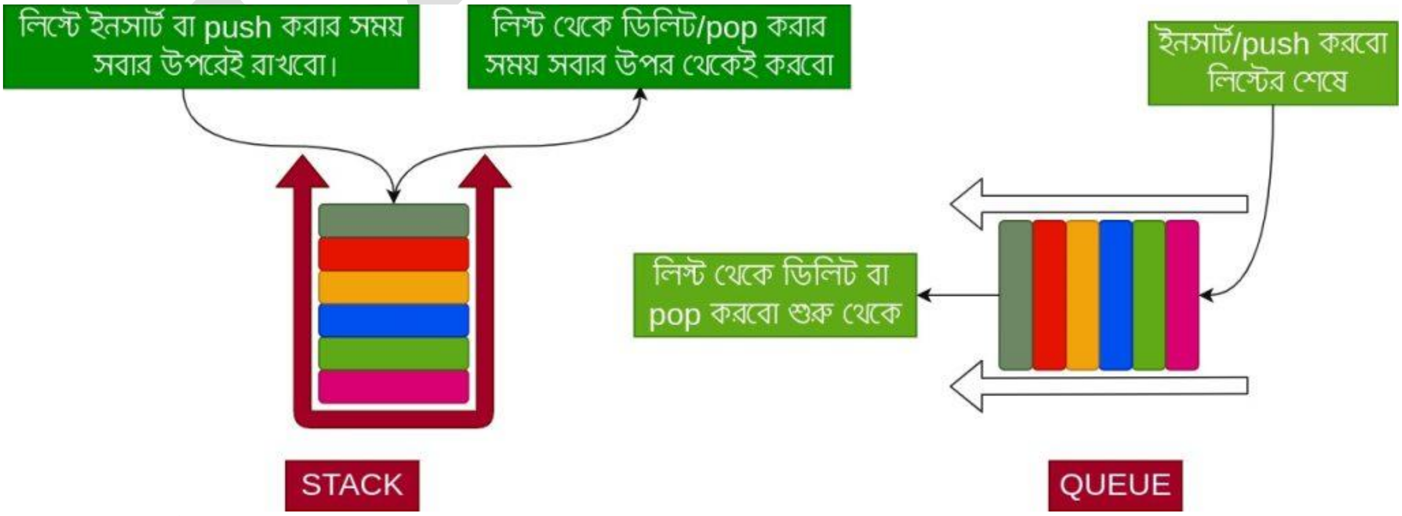


তারমানে আমরা বুঝতে পারছি, যে ডেটা স্ট্রাকচারে সবার শেষে এন্ট্রি করা ডেটাকে সবার আগে ডিলিট করা সম্ভব তাই স্ট্যাক। তাই এই ডেটা স্ট্রাকচারকে **LIFO (Last In First Out)** ডেটা স্ট্রাকচারও বলা হয়।

কিউ ডেটা স্ট্রাকচারের সংজ্ঞা-Queue definition: Queue একটি লিনিয়ার ডেটা স্ট্রাকচার যেখানে **FIFO** ক্রম অনুসারে ডেটা গুলোর উপর অপারেশন করা হয়।

এখন আরেকটি কেস বিবেচনা করি চলেন। ধরেন আপনি ব্যাংকে টাকা জমা দিবেন তাই আপনি লাইনে দাঁড়িয়েছেন। লাইনে আপনার পজিশন ধরা যাক ৬। তারমানে আপনার সামনে আরও ৫ জন আপনার আগে এসেছে। তাই তারা আগে জমা দিয়ে আগে চলে যাবে। আপনি সবার শেষে এসেছেন তাই আপনি সবশেষে টাকা জমা দিবেন।

অর্থাৎ যেসব ডেটা স্ট্রাকচারে সবার প্রথমে রাখা ডেটাকে আগে ডিলিট করা যায় তাকে কিউ ডেটা স্ট্রাকচার বলে। কিউ কে আমরা **FIFO (First In First Out)** নামেও ডাকি।



উপরের চিত্রটি দেখি, এখানে লিস্ট বলার কারণ হলো স্ট্যাক এবং কিউতে শুরুতেই সাইজ বলা থাকে না বিধায়

আমরা অ্যারের মত ফিক্সড সাইজ ডেটা স্ট্রাকচার নিয়ে কাজ করতে পারি না। তাই লিঙ্কড লিস্ট ব্যবহার করতে হবে। নিচের টেবিলে স্ট্যাক এবং লিঙ্কড লিস্টের পার্থক্য গুলো দিয়েছি।

স্ট্যাক এবং কিউ এর অপারেশনগুলো – Operations in stack and queue

1. **push** অপারেশন: স্ট্যাক বা কিউ তে কোন ডেটা রাখার অপারেশনকে push অপারেশন বলে। কিউ এর ক্ষেত্রে অনেক সময় এই অপারেশনকে enqueue অপারেশন বলা হয়।
2. **pop** অপারেশন: স্ট্যাক বা কিউ থেকে ডেটা ডিলিট করার অপারেশনকে pop অপারেশন বলে। কিউ এর ক্ষেত্রে এই অপারেশনকে dequeue ও বলা হয়।
3. **top** অপারেশন: শুধু স্ট্যাক এর জন্য এই অপারেশনের মাধ্যমে পরীক্ষা করা হয় স্ট্যাকের সবার উপরে কোন ইলিমেন্ট আছে।
4. **front** অপারেশন: শুধু কিউ এর এই অপারেশনের মাধ্যমে দেখা হয় কিউ এর সবার সামনে কোন ইলিমেন্ট আছে।

স্ট্যাক (Stack)	**কিউ (Queue)**
স্ট্যাক LIFO এর নিয়ম অনুসারে কাজ করে। মানে যে ইলিমেন্ট সবার শেষে ঢুকানো হবে তাকে সবার আগে বের করা হবে।	কিউ FIFO অনুযায়ী কাজ করে। সবার শুরুতে যেই ইলিমেন্ট ইনসার্টবা পুশ করা হবে তাকে সবার আগেই বের করা হবে।
ইনসার্ট বা ডেটা push করা এবং ডিলিট বা ডেটা pop করা সবসময় লিঙ্কড লিস্টের একই পাশ থেকে হয়। একে আমরা top বলি।	কিউ তে push করা হয় পেছন বা tail থেকে এবং pop করা হয় সামনে থেকে বা front থেকে।
স্ট্যাক ইমপ্লিমেন্ট করার জন্য আমাদের একটি পয়েন্টারের মাধ্যমে লিস্টের আইটেম push/ pop করতে হবে (top/ head pointer)।	কিউতে push/ pop এর জন্য আমাদেরকে দুইটি পয়েন্টার মেইন্টেইন করা লাগবে। একটির মাধ্যমে push করবো (tail pointer)। আরেকটির মাধ্যমে pop করবো (front/head pointer)।
রিকার্শনের মাধ্যমে সমস্যা সমাধানের ক্ষেত্রে স্ট্যাক ব্যবহার করা হয়।	ইটারেটিভ প্রসেসে সমস্যা সমাধানের জন্য সাধারণত কিউ ব্যবহার করা হয়।

Stack Overflow (স্ট্যাক ওভারফ্লো):

স্ট্যাক ওভারফ্লো ঘটে যখন স্ট্যাক ডেটা স্ট্রাকচারে প্রয়োজনের চেয়ে বেশি ডেটা যোগ করার চেষ্টা করা হয় এবং এর সীমাবদ্ধ স্থান পূর্ণ হয়ে যায়।

উদাহরণ:

- একটি স্ট্যাকের সর্বোচ্চ ক্ষমতা যদি ১০ হয়, এবং আপনি এর মধ্যে ১১তম আইটেম যোগ করতে চান, তখন "স্ট্যাক ওভারফ্লো" সমস্যা দেখা দেবে।

বাস্তব জীবনের উদাহরণ:

- একটি প্লেটের র‍্যাকে যদি আর প্লেট যোগ করার জায়গা না থাকে, তবুও আপনি প্লেট রাখার চেষ্টা করলে তা পড়ে যেতে পারে।

Stack Underflow (স্ট্যাক আন্ডারফ্লো):

স্ট্যাক আন্ডারফ্লো ঘটে যখন স্ট্যাক খালি থাকে, এবং আপনি সেখানে থেকে কোনো ডেটা সরানোর চেষ্টা করেন।

উদাহরণ:

- একটি খালি স্ট্যাক থেকে যদি কোনো আইটেম পপ (pop) করার চেষ্টা করা হয়, তখন "স্ট্যাক আন্ডারফ্লো" হবে।

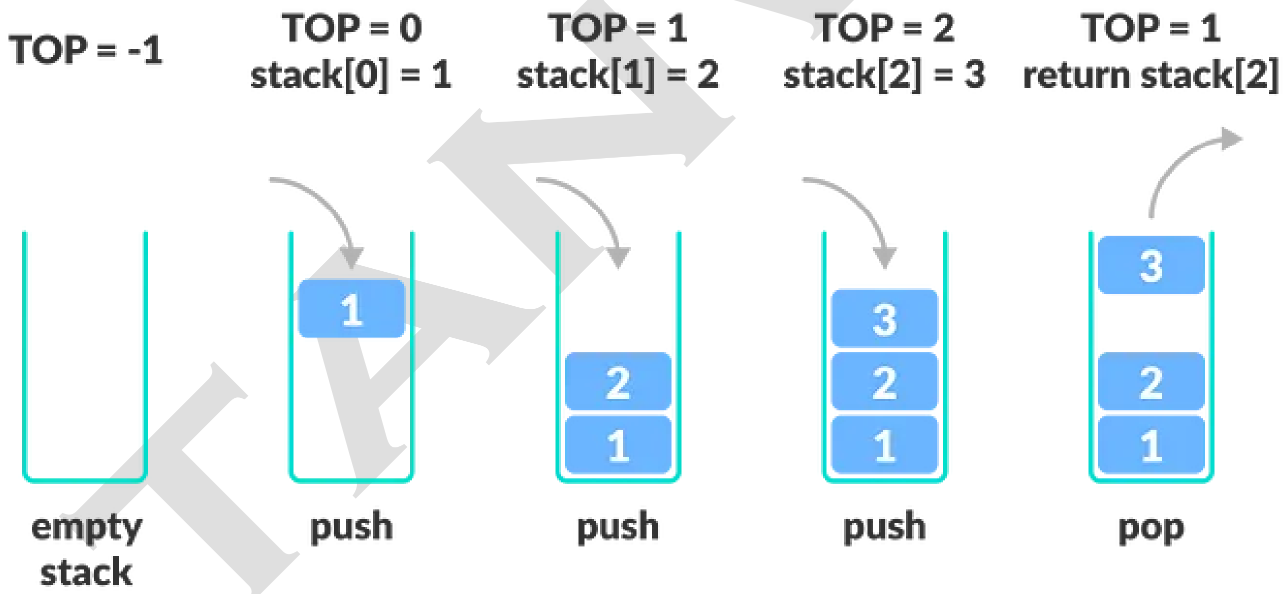
বাস্তব জীবনের উদাহরণ:

- যদি একটি খালি প্লেটের র‍্যাকে প্লেট খুঁজতে যান, তবে আপনি কিছুই পাবেন না।

মূল পার্থক্য:

বিষয়	**Stack Overflow**	**Stack Underflow**
কখন ঘটে?	যখন স্ট্যাক পূর্ণ হয়ে যায়।	যখন স্ট্যাক খালি থাকে।
কীভাবে সমাধান?	নতুন ডেটা যোগ করার আগে স্ট্যাক চেক করা।	ডেটা সরানোর আগে স্ট্যাক চেক করা।

Algorithm Stack(PUSH):



Stack	**Queue**
PUSH/POP	**PUSH**
Step 1 – Checks if the stack is full. **Step 2** – If the stack is full, produces an error and exit. **Step 3** – If the stack is not full, increments top to point next empty space. **Step 4** – Adds data element to the stack location, where top is pointing.	1. Checks if the stack is empty. 2. If the stack is empty, produces an error and exit. 3. If the stack is not empty, accesses the data element at which top is pointing. 4. Decreases the value of top by 1.

Step 5 – Returns success	5. Returns success.

Algorithm for PUSH Operation:

```
1. Star procedure push: stack, data
2. if stack is full
        return null
   else
        top = top+1
        stack[top] = data
3. End procedural
```

Code:

```
Void push(int data){
      If(!isFull()){
          top = top +1;
          stack[top] = data;
      }
      else{
          printf("Stack is Full.")
      }
}
```

Algorithm for POP Operation:

```
1. Star procedure push: stack
2. if stack is empty
        return null
   else
        data = stack[top]
        top = top – 1
        return data
3. End procedural
```

Code:

```
Void push(int data){
      If(!isFull()){
              data = stack[top];
              top = top - 1;
              return data;
      }
      else{
          printf("Stack is Empty.")
      }
}
```