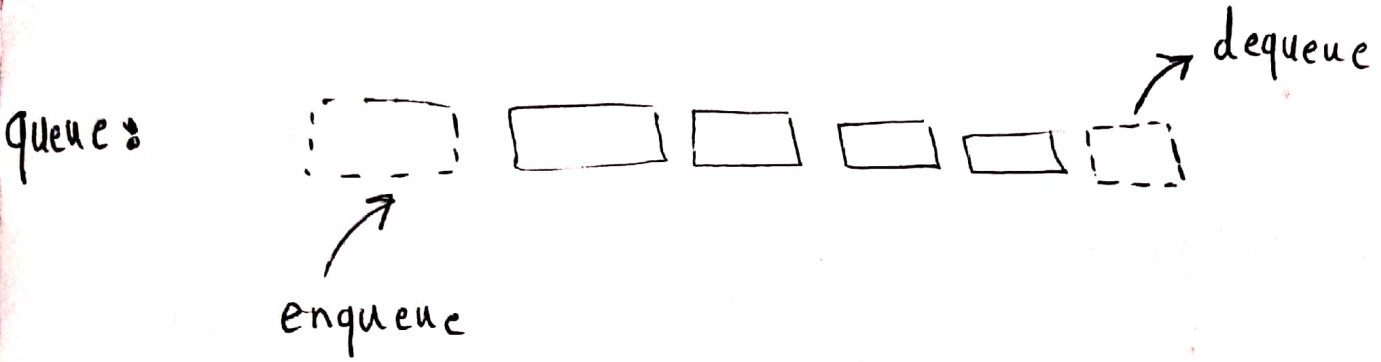
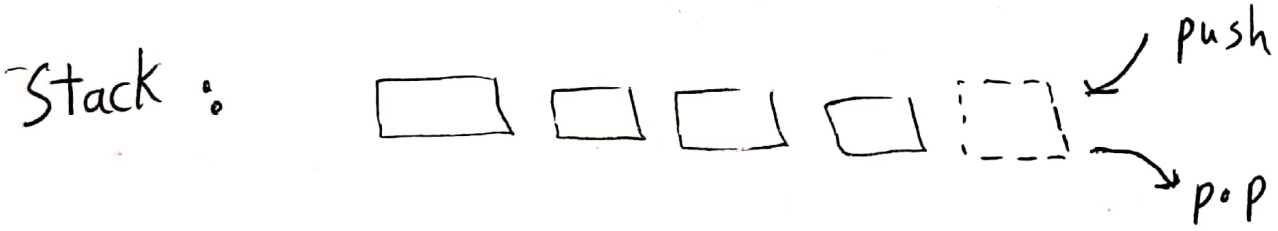
فرایند enqueue و dequeue در queue به شکل زیر است:




و فرایندهای pop و push در stack به شکل زیر است:




---

با توجه به ② شکل بالا برای پیاده‌سازی dequeue کافیست یکبار از stack اصلی خود که نقش صف را ایفا می‌کند یکبار pop() کنیم.

امّا برای enqueue کردن، ابتدا باید stack اصلی را خالی کرده (آن را در stack دوم بریزیم) سپس مقدار جدید را push کرده و دوباره مقادیر قبلی را به سرجای خود، انتقال دهیم.

حال اگر Stack اصلی را Stack ، Stack دوم که برای ذخیره سازی مقادیر Stack اول است را box بنامیم داریم :

```
enqueue (x):

    while(stack.top != -1):
        tmp = stack.pop
        box.push(tmp)
    stack.push(x)

    while(box.top != -1):
        tmp = box.pop
        stack.push(tmp)
```

ریختن مقادیر در box و اضافه کردن مقدار جدید به ابتدای Stack

ریختن مقادیر اولیه به Stack

---

```
dequeue :
    if(stack.top != -1):
        return stack.pop
```

آخرین عنصر Stack را حذف می‌کند و آن را برمی‌گرداند.

« کد این پیاده سازی نیز در صفحه بعد موجود است »

```cpp
struct Queue{
  stack <int> Stack;
  stack <int> box;
};
int Dequeue(Queue *q){
  if (!q->Stack.empty()){
    int tmp = q->Stack.top();
    q->Stack.pop();
    return tmp;
  }
  return 0;
}
void Enqueue(Queue *q, int value){
  while (!q->Stack.empty()){
    int tmp = q->Stack.top();
    q->Stack.pop();
    q->box.push(tmp);
  }
  q->box.push(value);
  while (!q->box.empty()){
    int tmp = q->box.top();
    q->box.pop();
    q->Stack.push(tmp);

  }
}
void printQueue(stack<int> s){
if (s.empty())
    return;
int x = s.top();
s.pop();
printQueue(s);
cout << x << " ";
s.push(x);
}
int main()
{
  Queue q;
  Enqueue(&q,2);
  Enqueue(&q,3);
  Enqueue(&q,4);
  printQueue(q.Stack);
  cout << "\n";
  Dequeue(&q);
  printQueue(q.Stack);
  cout << "\n";
}
```

```
armin@armin-Aspire-A715-71G: ~/Des...
(base) armin@armin-Aspire-A715-71G:~/Desktop/c++$ ./output
4 3 2
4 3
(base) armin@armin-Aspire-A715-71G:~/Desktop/c++$
```