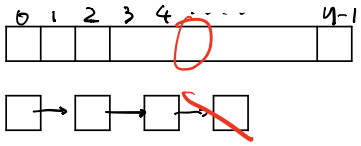
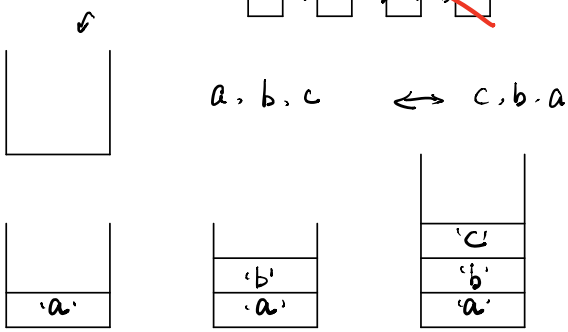
栈



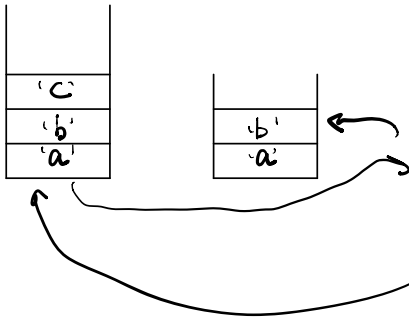
arr[i] = □

list.add(_)



First In Last Out

s.push('a') 压栈



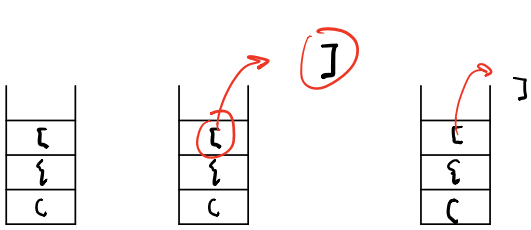
ArrayList<> _ = new ····

Stack<> s = new Stack···

s.pop() → c弹栈

s.peek() → c

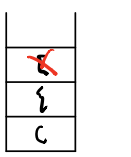
({[ ] [ ] })



({[ ] [ ] })

case '(' '{' '[' → stack.push( )

more right



return false

case ')' '}' ']' → stack.pop()

if(!stack.isEmpty()) return false

( )
{ }
[ ]

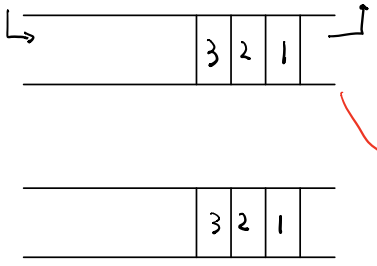
(({ ···

more left stack.isEmpty()

伪代码

```
boolean isValid (String brackets) {
  Stack <character> s = ----;
  // brackets element
  for (int i = ; i <  ; i++) {
      if (element == '(' '{' '[') {
          s.push(element);
      } else {
          if ( s.isEmpty()) {
              return false;
          } else {
              char c = s.pop();
              compare element , c
              if (not match) {
                  return false
              }
          }
      }
  }
  return s.isEmpty();
}
```

- add
- push
- offer
- remove
- pop
- poll



1, 2, 3

1, 2, 3

3, 2, 1

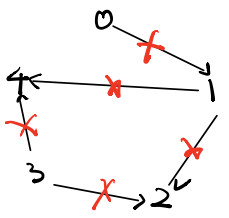
```
Stack ... = new Stack...

Queue<Integer> q =
    new PriorityQueue <>();

Queue<Integer> q =
    new LinkedList<>();
```

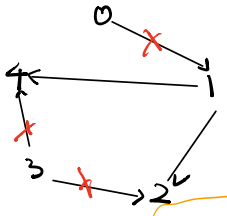
5 0, 1, 2, 3, 4

5 [[4,1], [2,3], [2,1], [4,3], [1,0]]



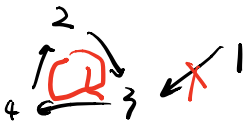
0, 1, 3, 2, 4

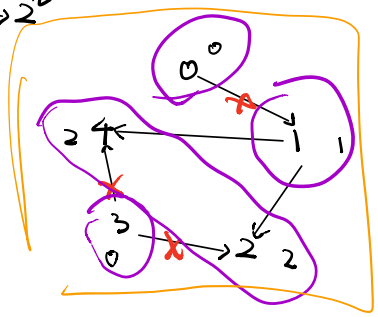
0, 1, 3, 4, 2



3, 0 1, 2, 4

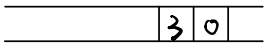
	0	1	2	3	4
0	0	0	0	0	0
1	1	0	0	0	0
2	0	1	0	1	0
3	0	0	1	0	0
4	0	1	0	0	0

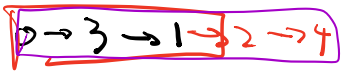


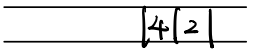


indegree

0	1	2	3	4
0	1	2	0	3
0	1		1	
	0		0	







while (!q.isEmpty()) {

q.poll() // take conse

modify indegree

scan indegree

==0?