stack&queue

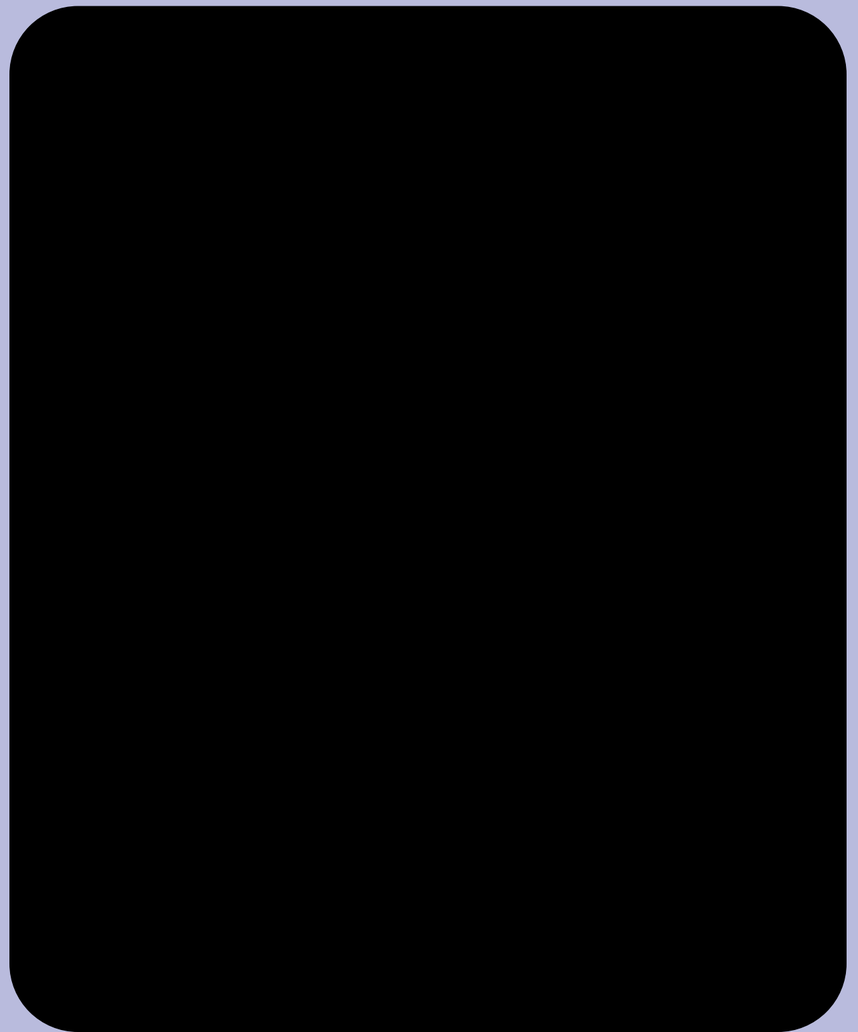
資研下學期第五次社課

進度

1. 什麼是stack&queue
2. 簡單的應用場景
3. 兩者的語法
4. 結論

先簡單的介紹一下stack(堆疊,堆棧)

可以想做是一個箱子要裝跟卸東西



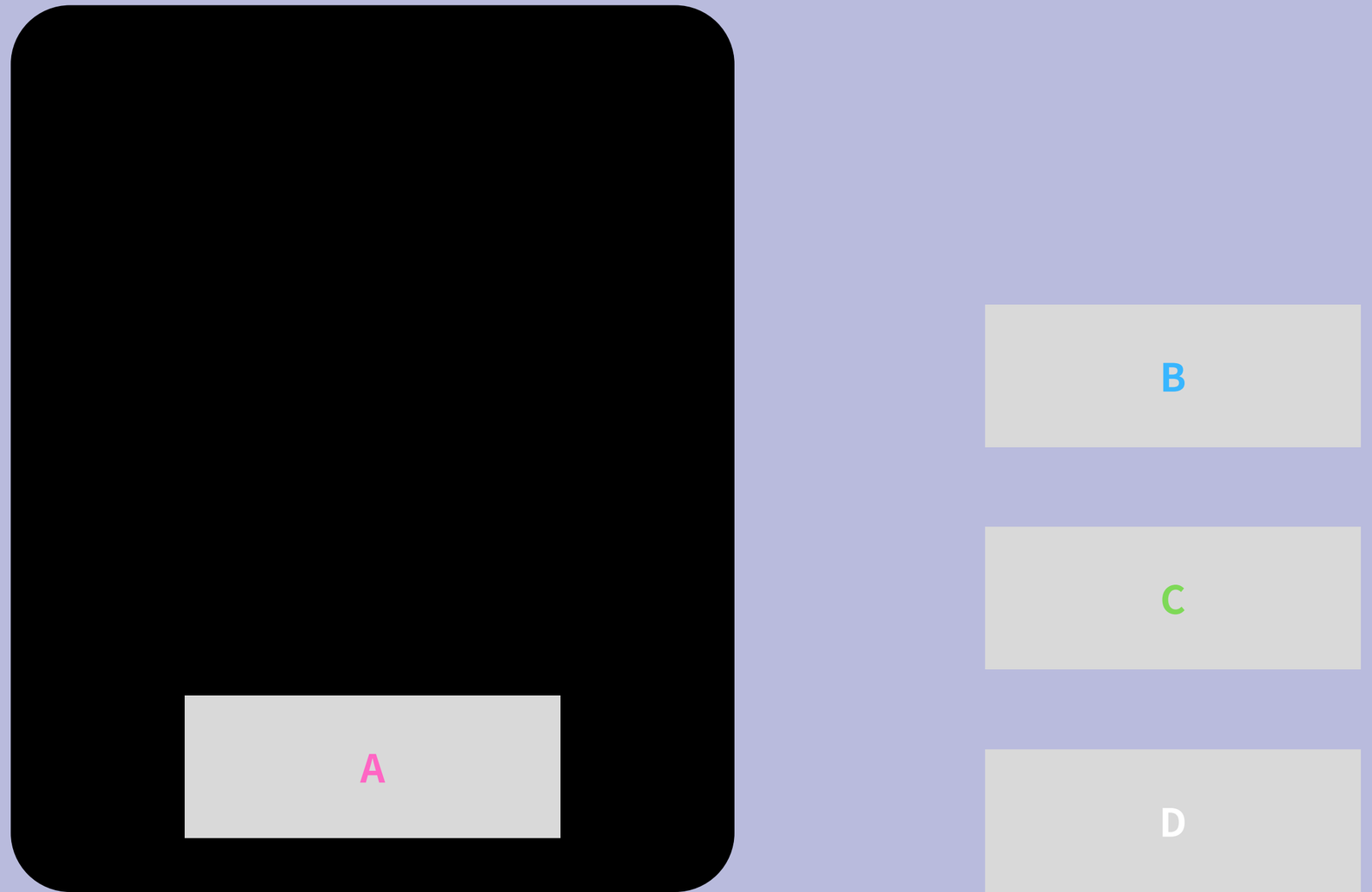
A

B

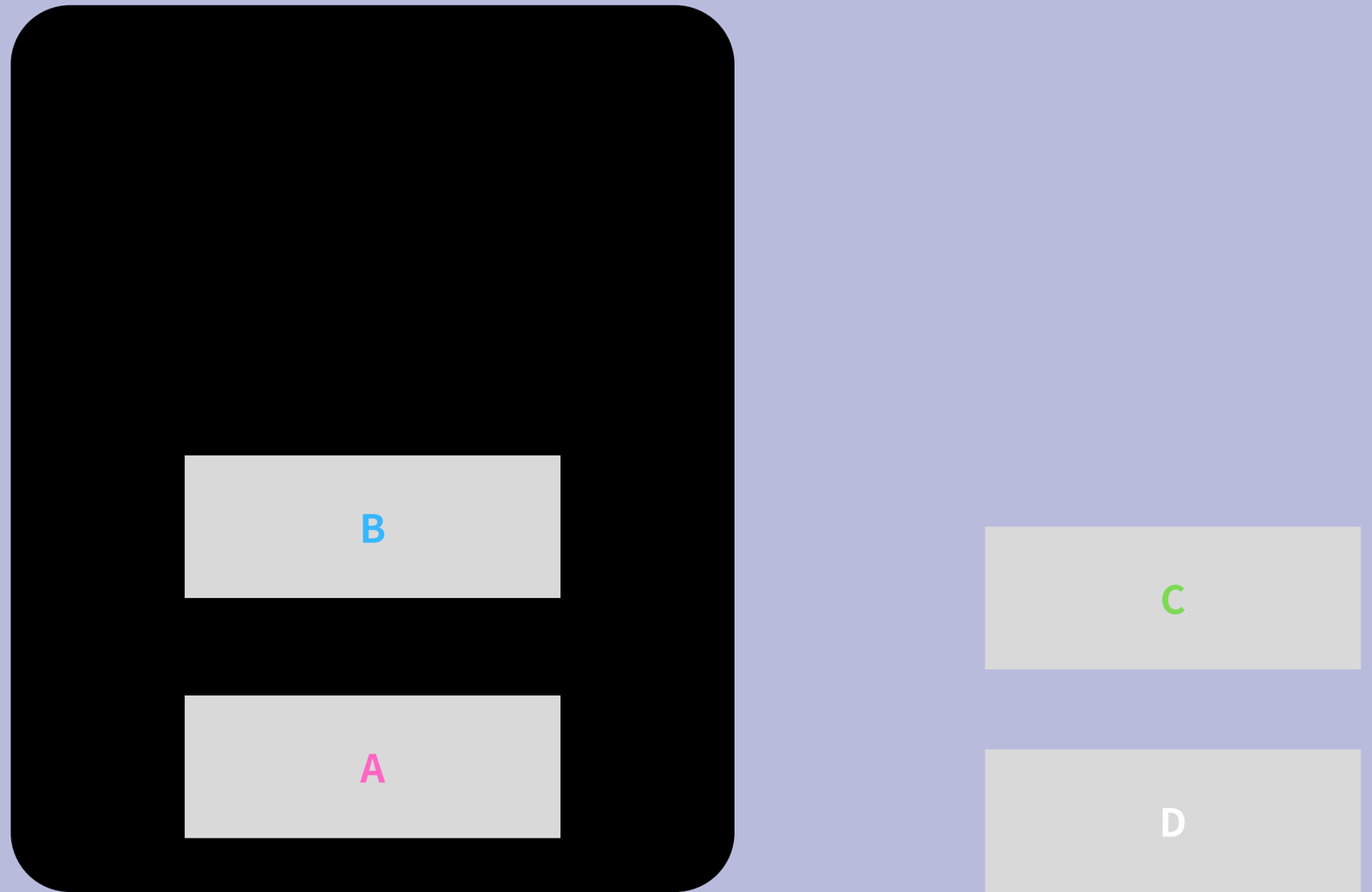
C

D

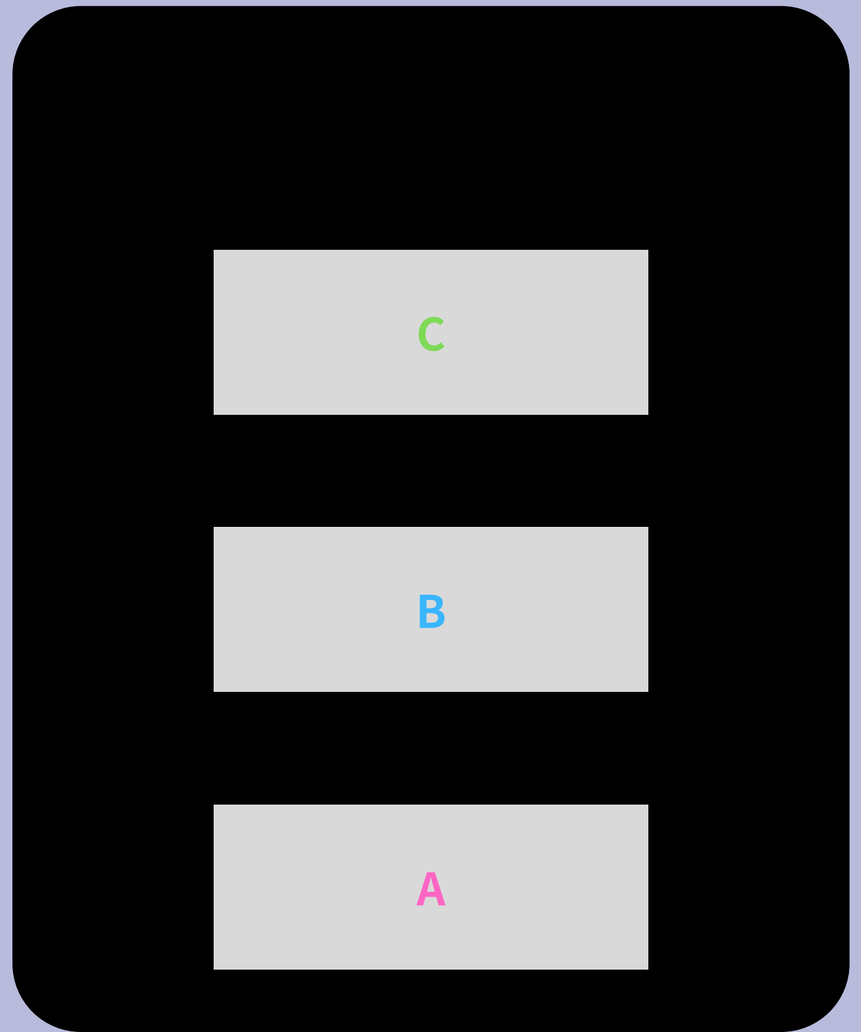
可以想做是一個箱子要裝跟卸東西



可以想做是一個箱子要裝跟卸東西

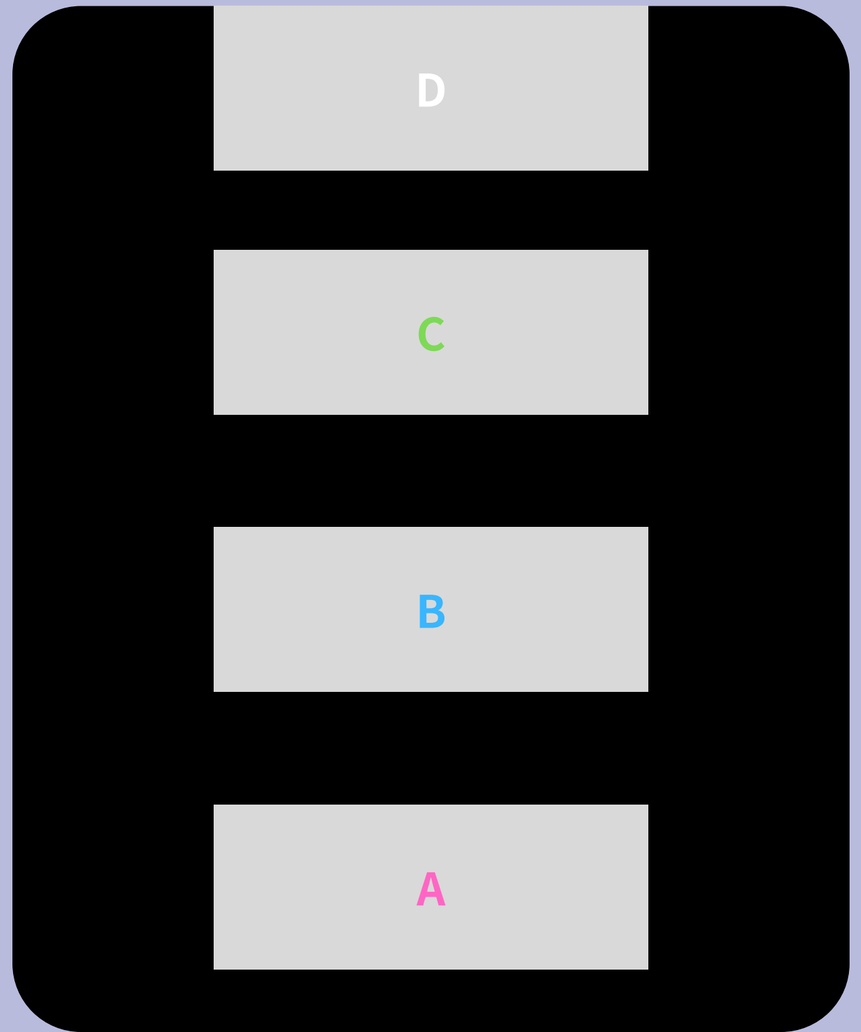


可以想做是一個箱子要裝跟卸東西

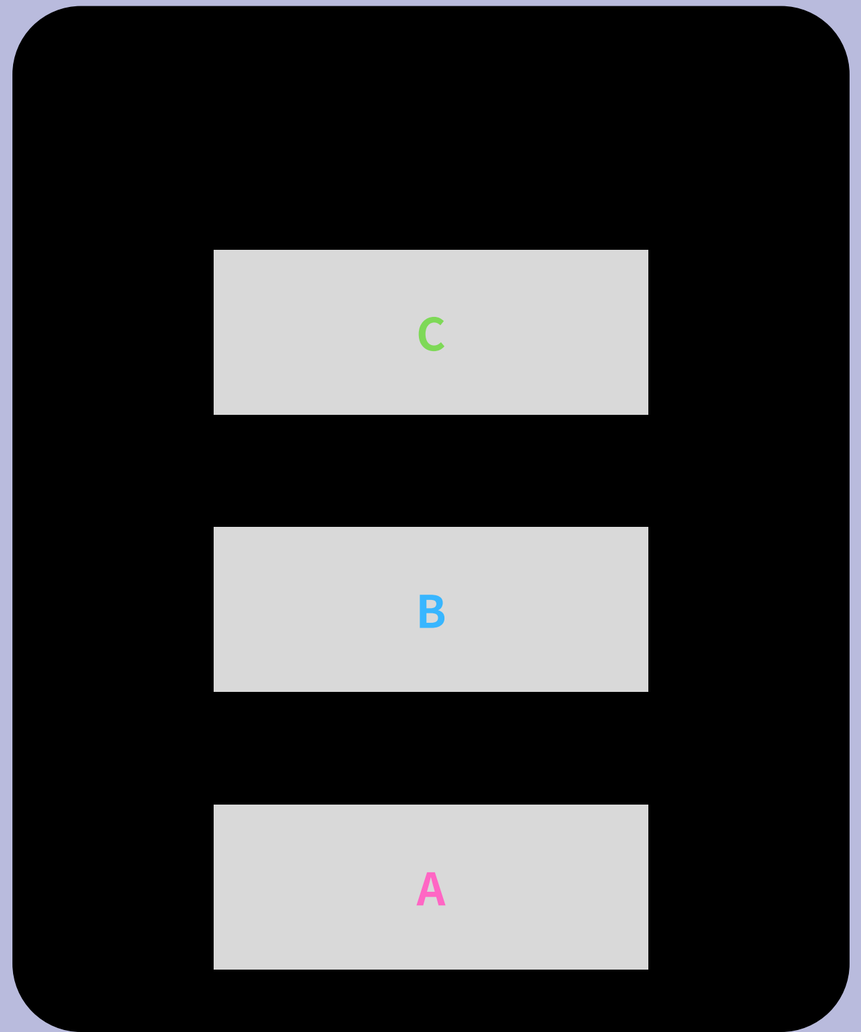


D

可以想做是一個箱子要裝跟卸東西

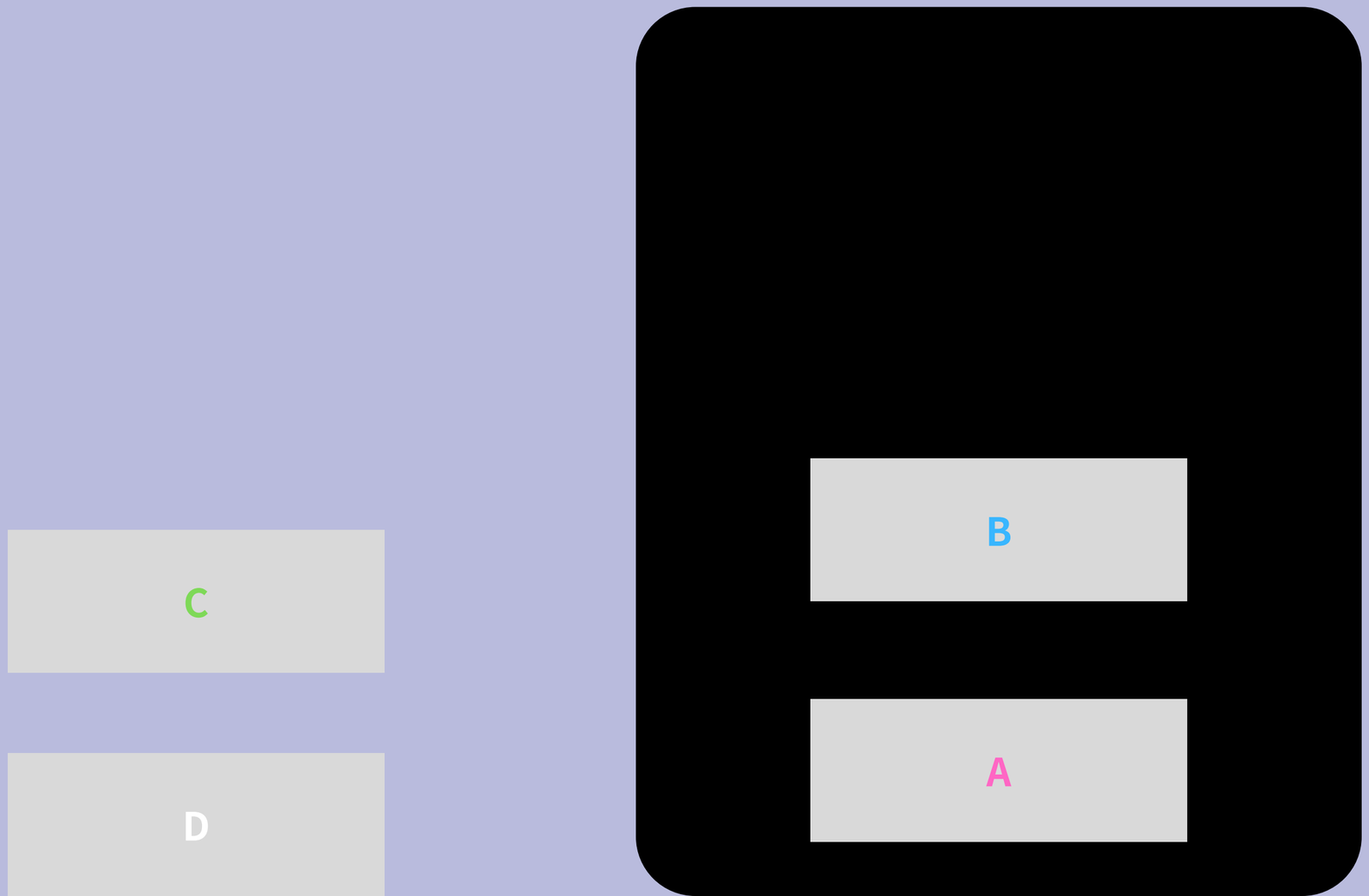


可以想做是一個箱子要裝跟卸東西

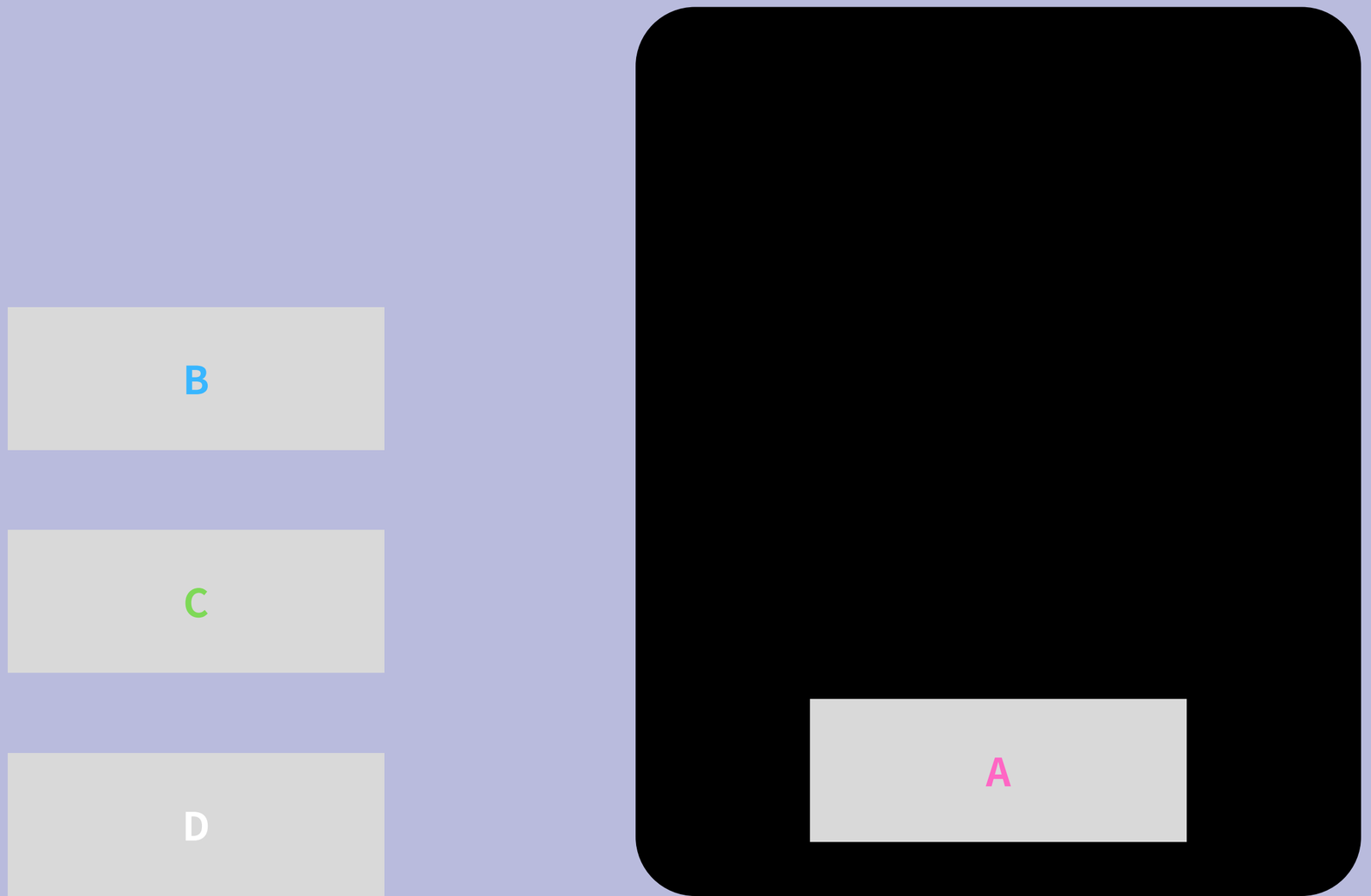


D

可以想做是一個箱子要裝跟卸東西



可以想做是一個箱子要裝跟卸東西



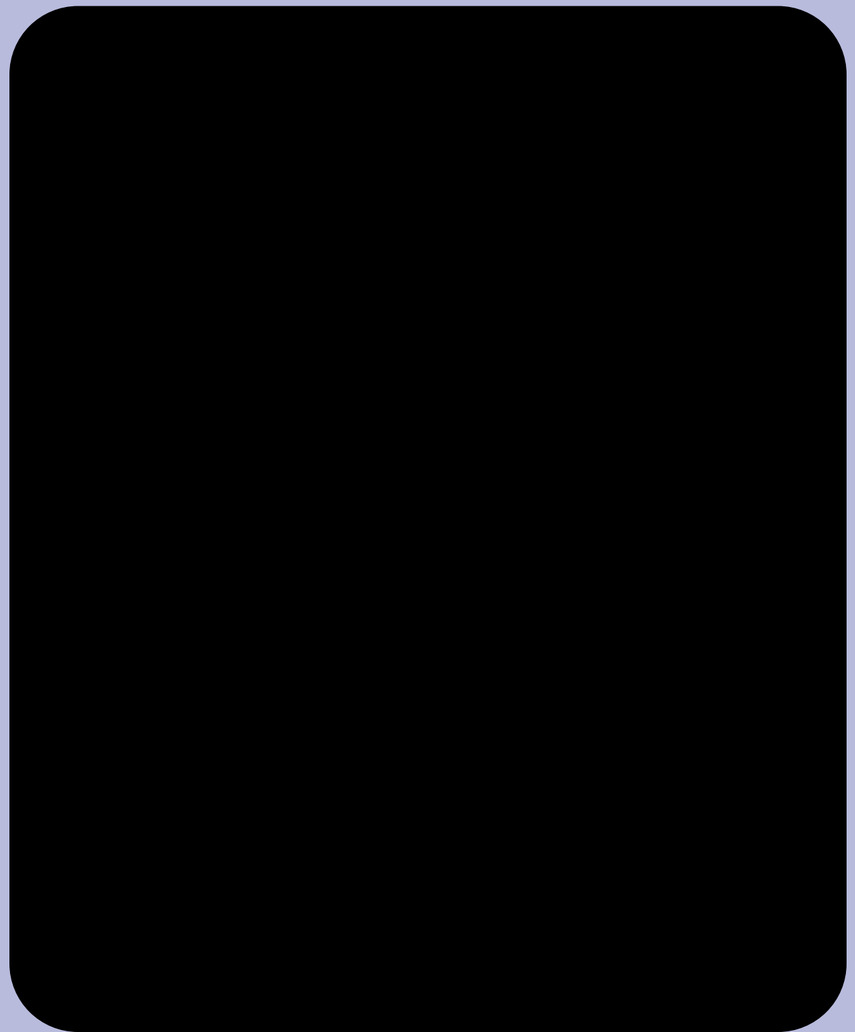
可以想做是一個箱子要裝跟卸東西

A

B

C

D



由此可知stack是先進後出

first in last out (FILO)

舉個現實中常見堆疊的例子

- 吃到飽的盤子,杯子
- 堆疊起來的書本
- 袋裝吐司

接著是queue(佇列)

可以想成是一條水管(兩邊都有開洞)

A

B

C

D

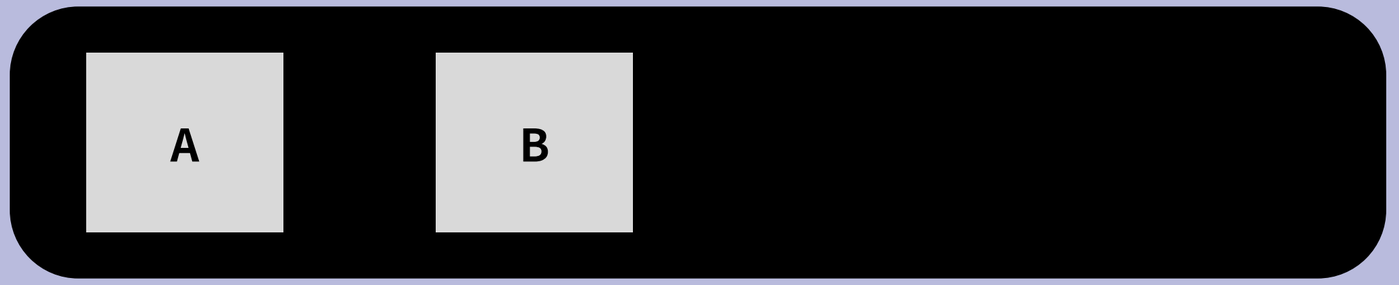
可以想成是一條水管(兩邊都有開洞)

B

C

D

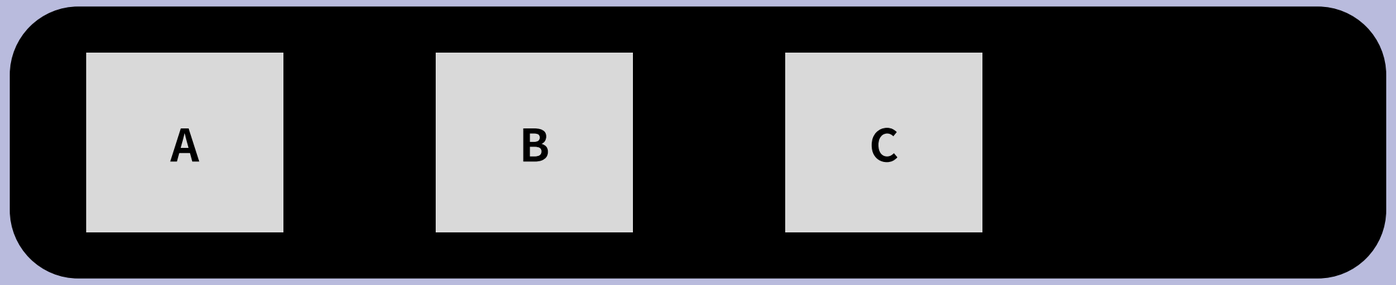
可以想成是一條水管(兩邊都有開洞)



C

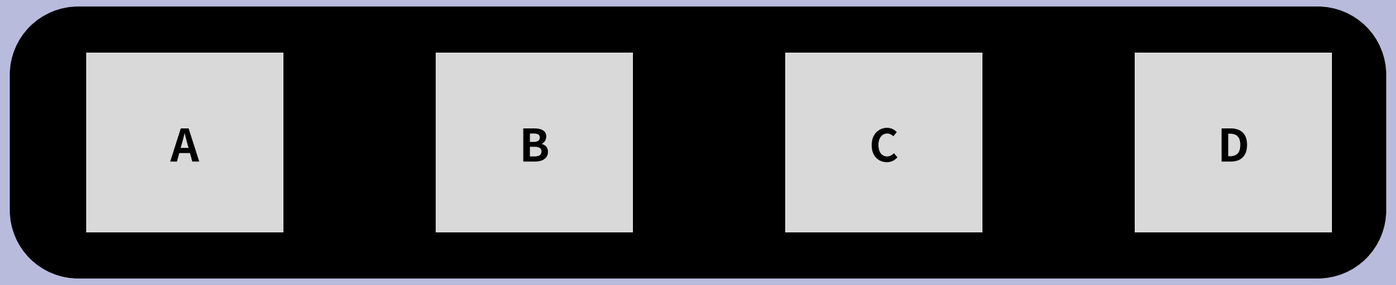
D

可以想成是一條水管(兩邊都有開洞)

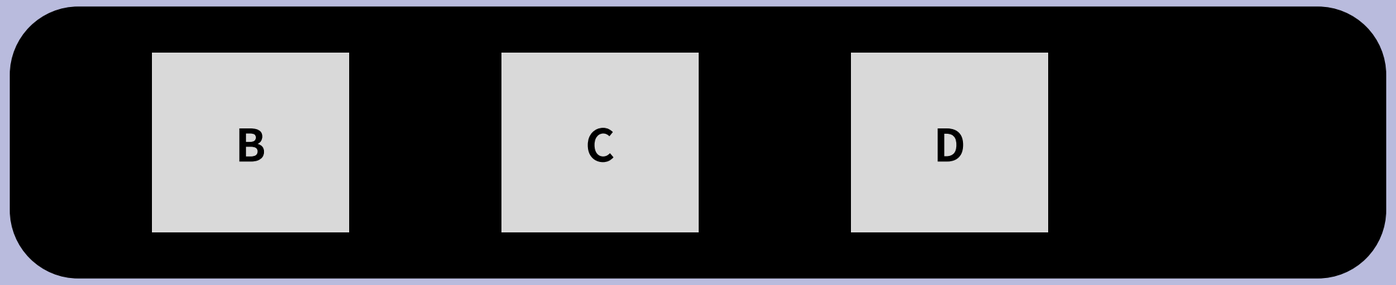


D

可以想成是一條水管(兩邊都有開洞)

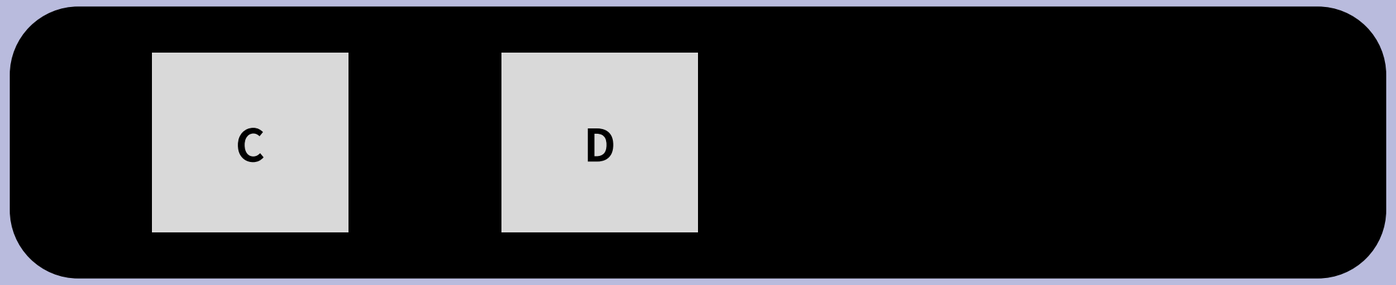


可以想成是一條水管(兩邊都有開洞)



A

可以想成是一條水管(兩邊都有開洞)



B

A

可以想成是一條水管(兩邊都有開洞)

C

B

A

可以想成是一條水管(兩邊都有開洞)

D

C

B

A

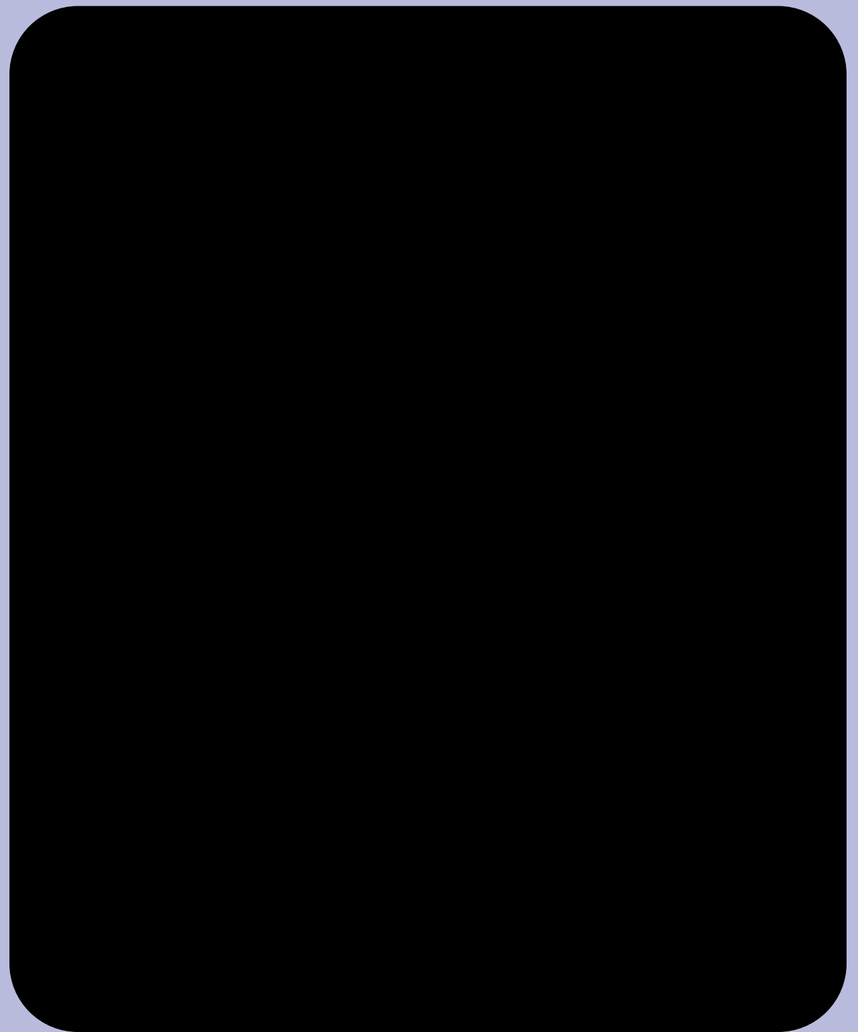
由此可知queue是先進先出

first in first out (FIFO)

語法

創建: stack<datatype>name

可以想做是一個箱子要裝跟卸東西



A

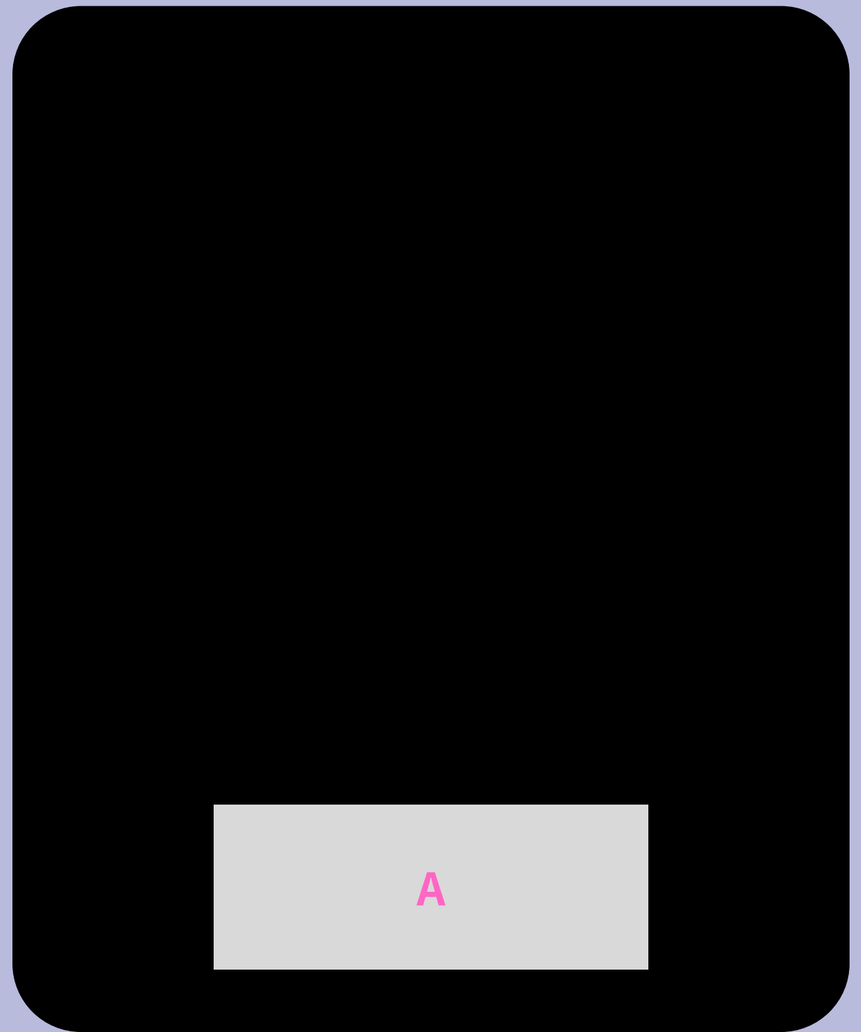
B

C

D

可以想做是一個箱子要裝跟卸東西

```cpp
#include<bits/stdc++.h>//萬能標頭檔
using namespace std;
int main(){
    /*宣告一個名叫box的stack容器
    用來放char類型的元素*/
    stack<char> box;
    //放A進去
    box.push('A');
    //輸出當前箱子的頂部元素
    cout<<box.top();
    return 0;
}
```



box.push(‘A’)

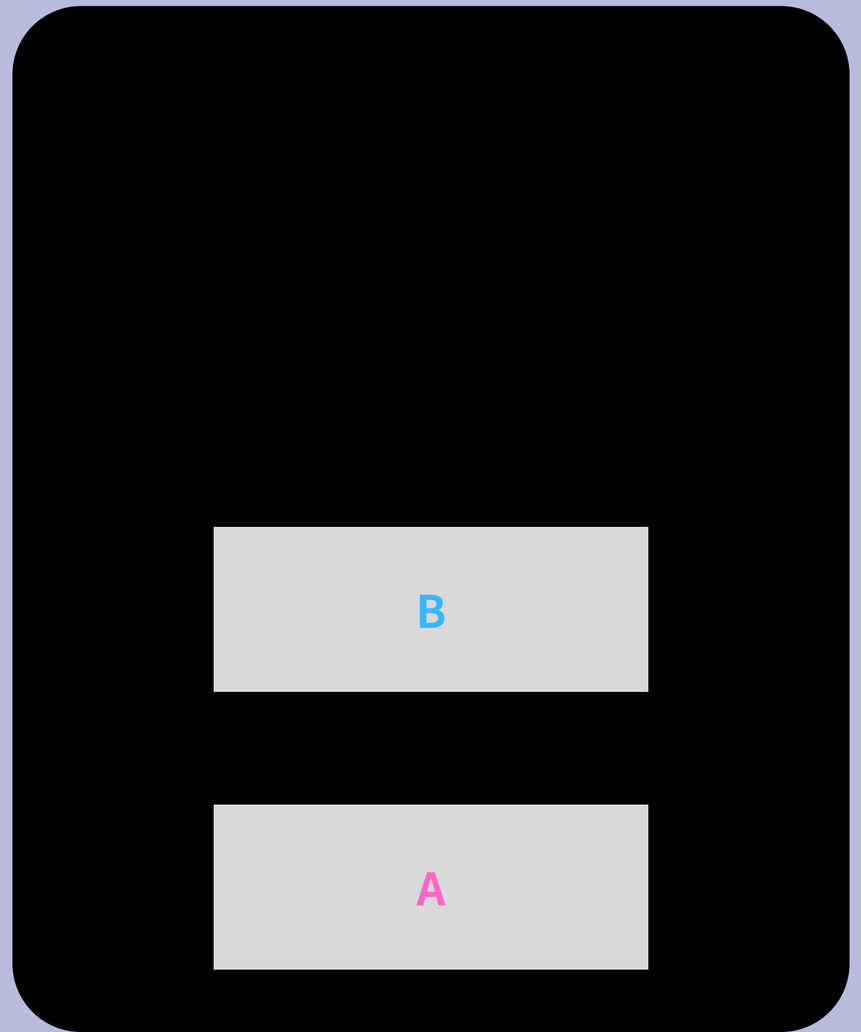
B

C

D

可以想做是一個箱子要裝跟卸東西

```cpp
#include<bits/stdc++.h>//萬能標頭檔
using namespace std;
int main(){
    /*宣告一個名叫box的stack容器
    用來放char類型的元素*/
    stack<char> box;
    //放A進去
    box.push('A');
    //放B進去
    box.push('B');
    //輸出當前箱子的頂部元素
    cout<<box.top();
    return 0;
}
```



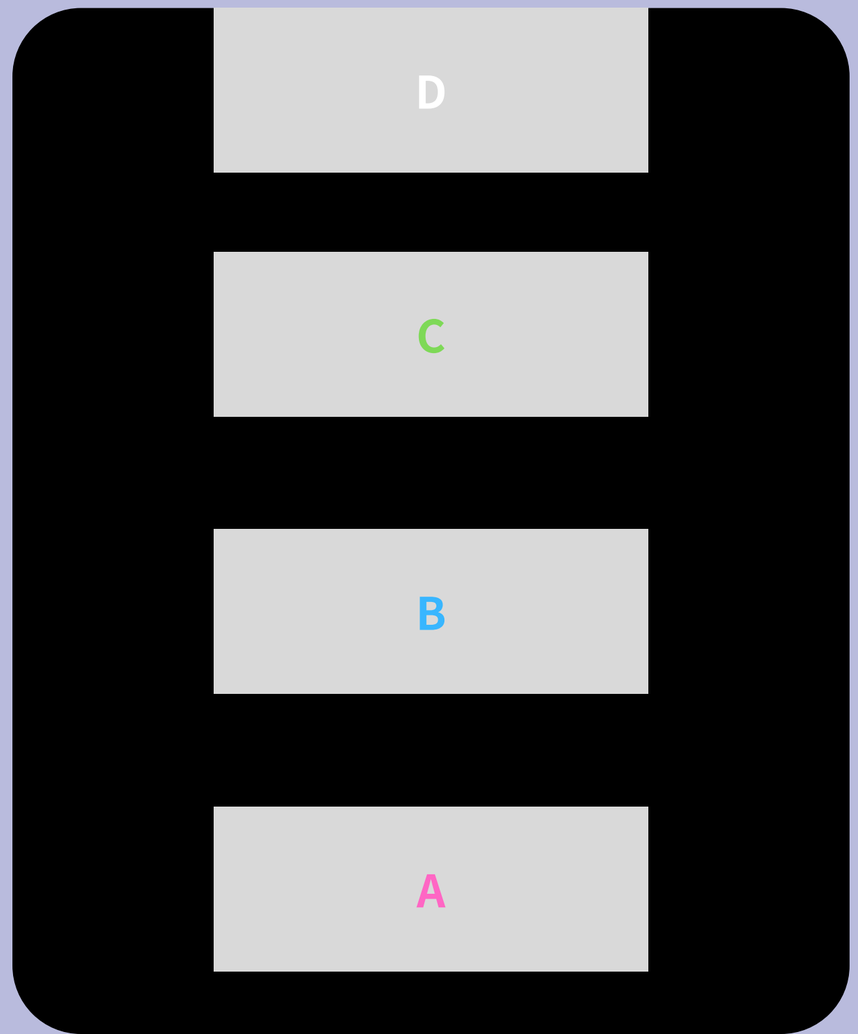
box.push(‘B’)

C

D

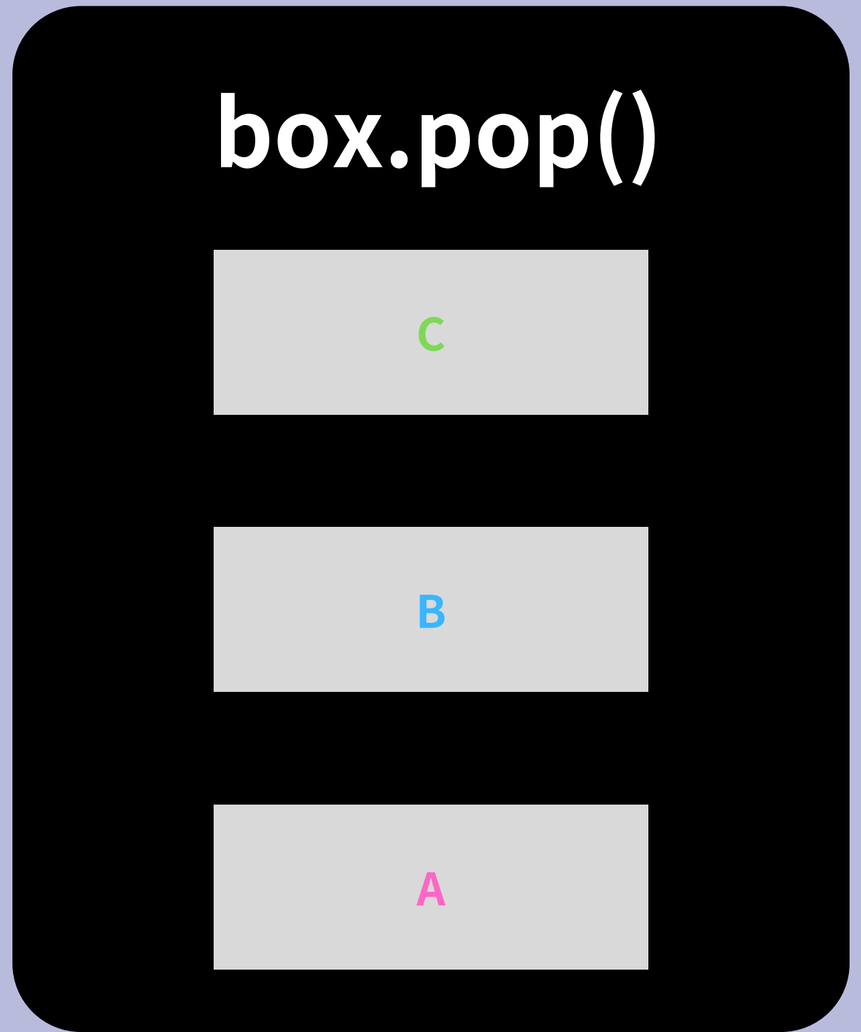
可以想做是一個箱子要裝跟卸東西

```cpp
#include<bits/stdc++.h>//萬能標頭檔
using namespace std;
int main(){
    /*宣告一個名叫box的stack容器
    用來放char類型的元素*/
    stack<char> box;
    //放A進去
    box.push('A');
    //放B進去
    box.push('B');
    //放C進去
    box.push('C');
    //放D進去
    box.push('D');
    //輸出當前箱子的頂部元素
    cout<<box.top();
    return 0;
}
```



box.push(‘D’)

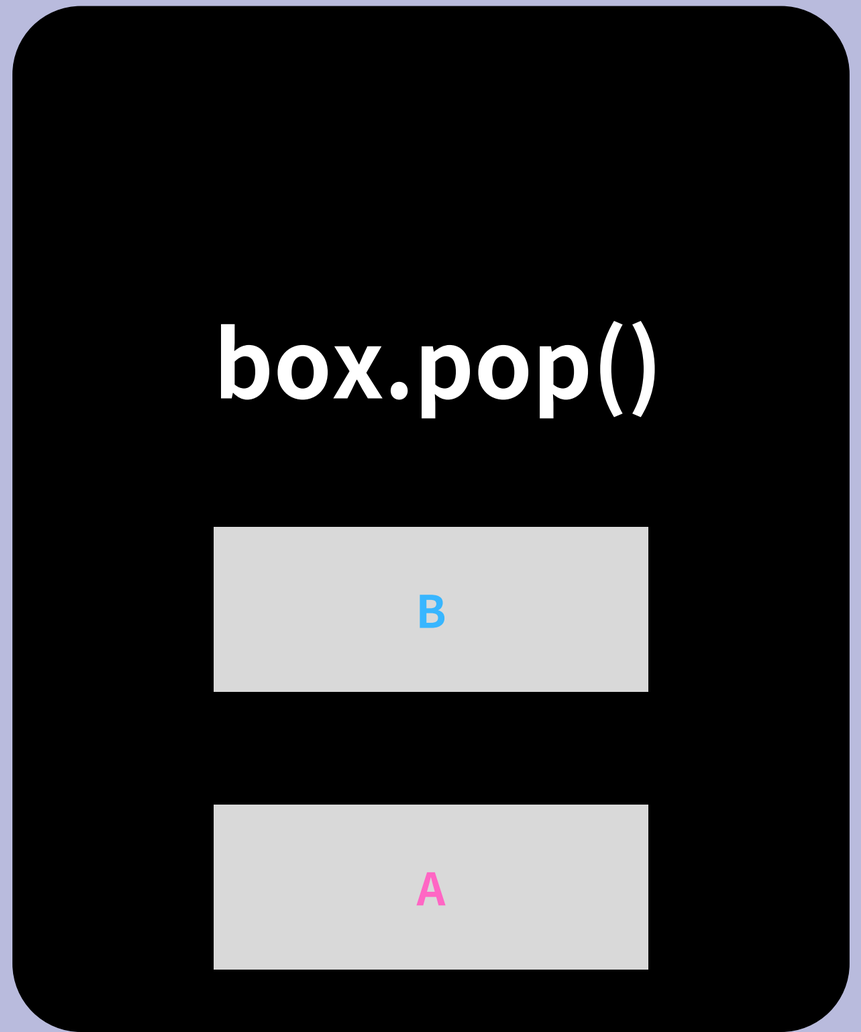
可以想做是一個箱子要裝跟卸東西



D

```
用來放char類型的元素*/
stack<char> box;
//放A進去
box.push('A');
//放B進去
box.push('B');
//放C進去
box.push('C');
//放D進去
box.push('D');
//丟出去當前第一個
box.pop();
```

可以想做是一個箱子要裝跟卸東西



C

D

```
box.push('C');
//放D進去
box.push('D');
//丟出去當前第一個
box.pop();
//丟出去當前第一個
box.pop();
//輸出當前箱子的頂部元素
cout<<box.top();
return 0;
```

可以想做是一個箱子要裝跟卸東西

A

B

C

D

box.pop()

```
    box.push('D');
    //丟出去當前第一個
    box.pop();
    //丟出去當前第一個
    box.pop();
    //丟出去當前第一個
    box.pop();
    //丟出去當前第一個
    box.pop();
    //輸出當前箱子的頂部元素
    cout<<box.top();
    return 0;
}
```

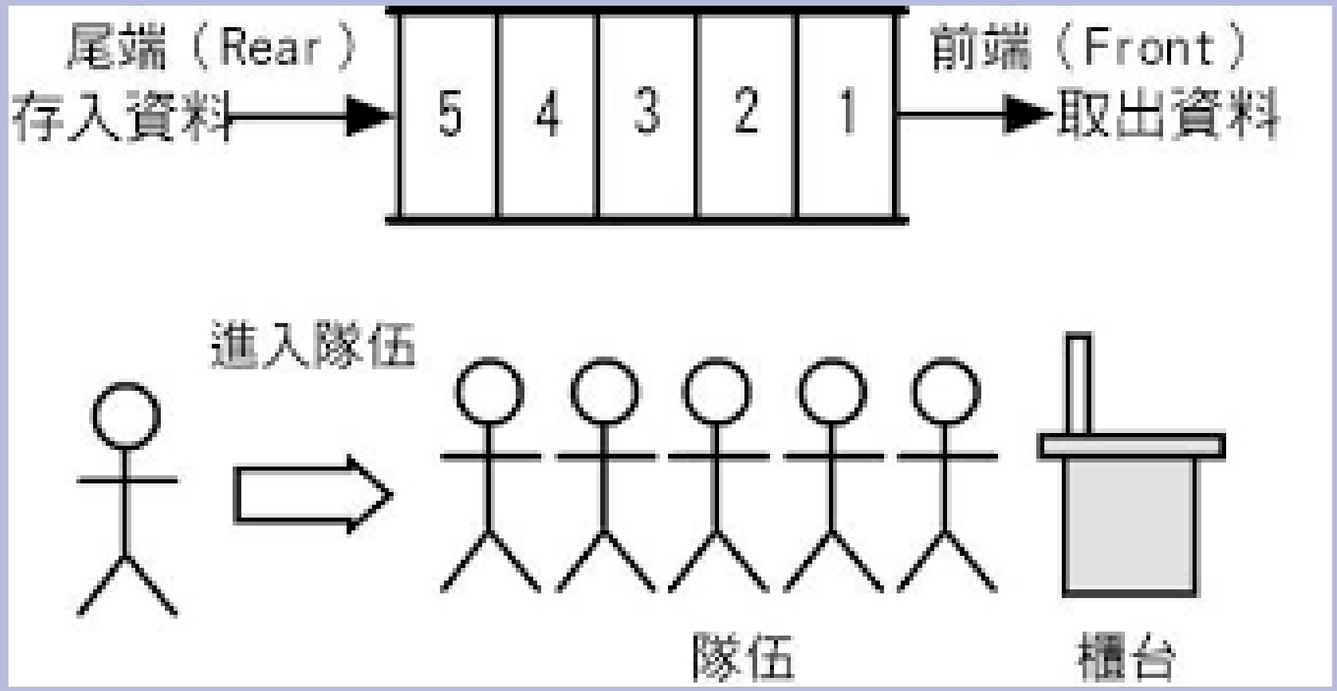
queue要知道最前面的資料要改成name.front喔別用成.top

queue跟這差不多(pop,push)
不過是先進先出

創建:queue<datatype>name

生活中的例子

- 輸送帶
- 排隊結帳



來個一題觀念題

以下是一個stack容器,順序為a,b,c,d,e,f,g
下列()不可能是合法的出棧順序

A： a , b , c , d , e , f , g
B： a , d , c , b , e , g , f
C： a , d , b , c , g , f , e
D： g , f , e , d , c , b , a

最後的一題實作題

使用佇列的資料結構，依序加入1、3、5、7等4個元素，移出其中2個元素後，再加入10、15、16等3個元素，再移出1個元素，最後再加入20，請問結果由前端至尾端之元素依序為何？

```cpp
#include<bits/stdc++.h>//萬能標頭檔
using namespace std;
int main(){
    queue<int>box;
    box.push(1);box.push(3);box.push(5);box.push(7);
    box.pop();box.pop();
    box.push(10);box.push(15);box.push(16);
    box.pop();
    box.push(20);
    while(box.size()>0){
        cout<<box.front()<<" ";
        box.pop();
    }
}
```

可以改成stack試試看會有啥不同