集训课-15：数据结构与STL模板

课程目标

01 STL栈

02 STL队列

03 STL优先队列

Part 1

STL栈

STL栈

数据结构是计算机存储、组织数据的方式。是指相互之间存在一种或多种特点关系的数据元素的集合。

数据结构可以存储一个或者一个以上的数据元素。

变量属于数据结构吗?

STL栈

下面属于数据结构的是（ ）

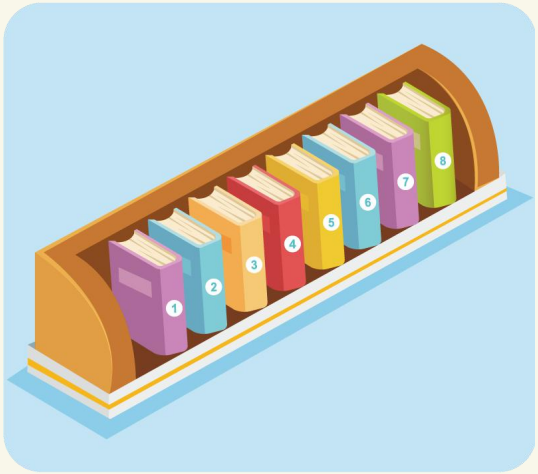
A.int a = 1;

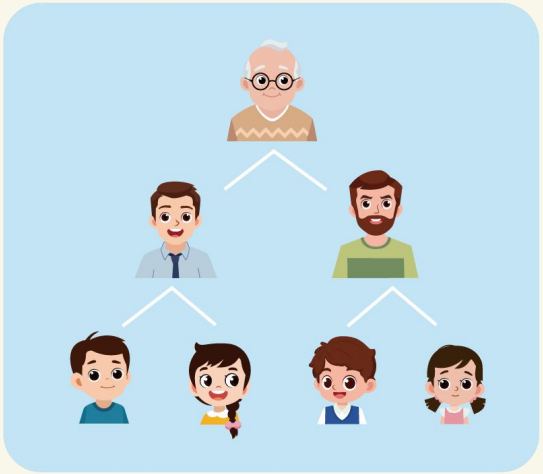
B.char b = 'a';

C.string name = "keduo";

D.int num[3]= {1,2,3};

STL栈





一对一

线性结构

一对多

树形结构

多对多

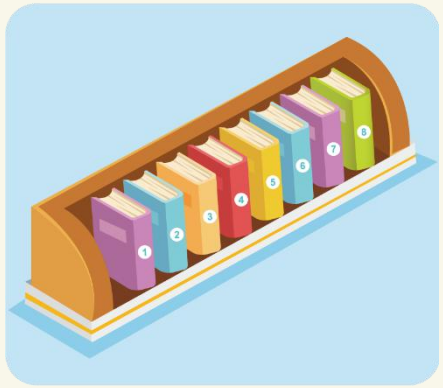
图形结构

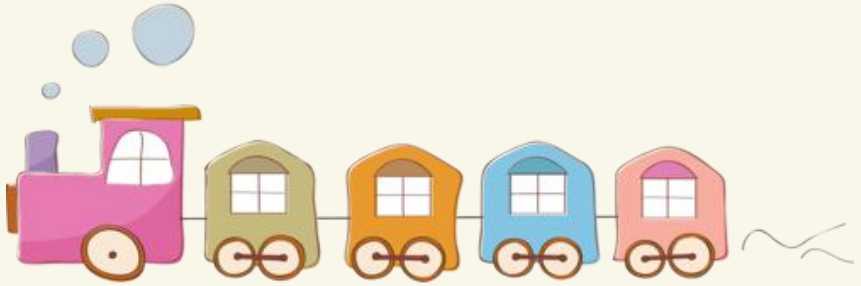
数据结构

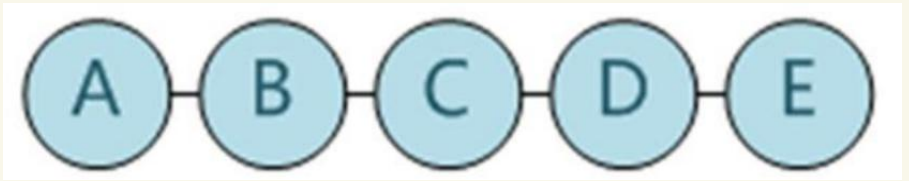
STL栈

一对一

线性结构





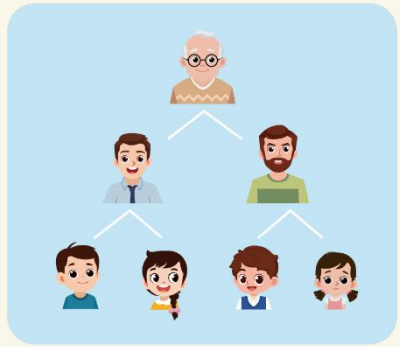


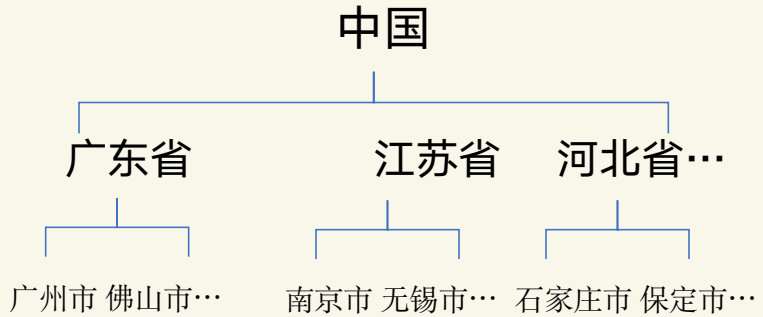
……

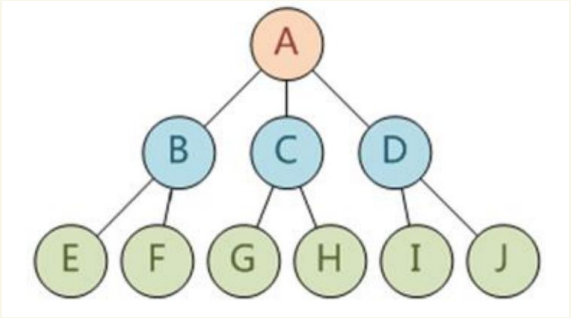
STL栈

一对多

树形结构





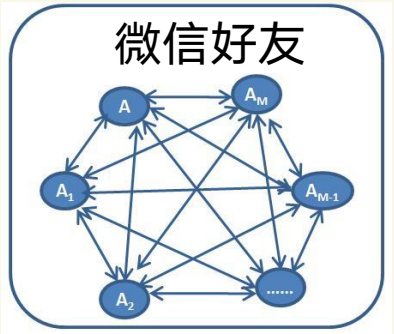


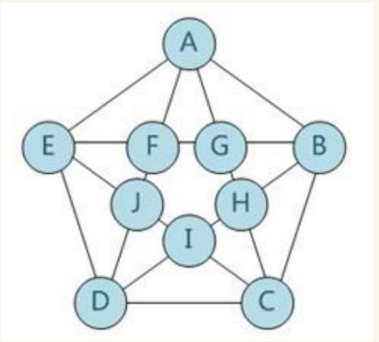
……

STL栈

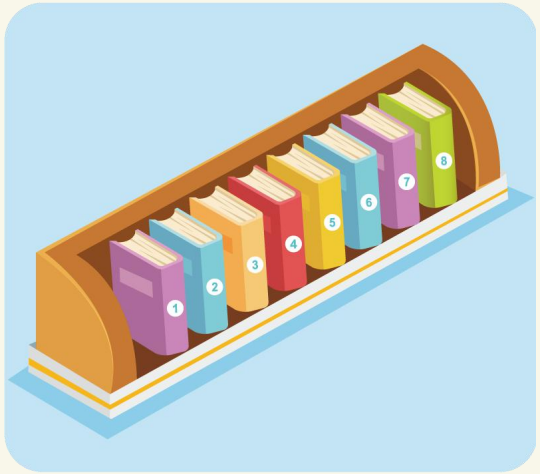
多对多

图形结构





STL栈



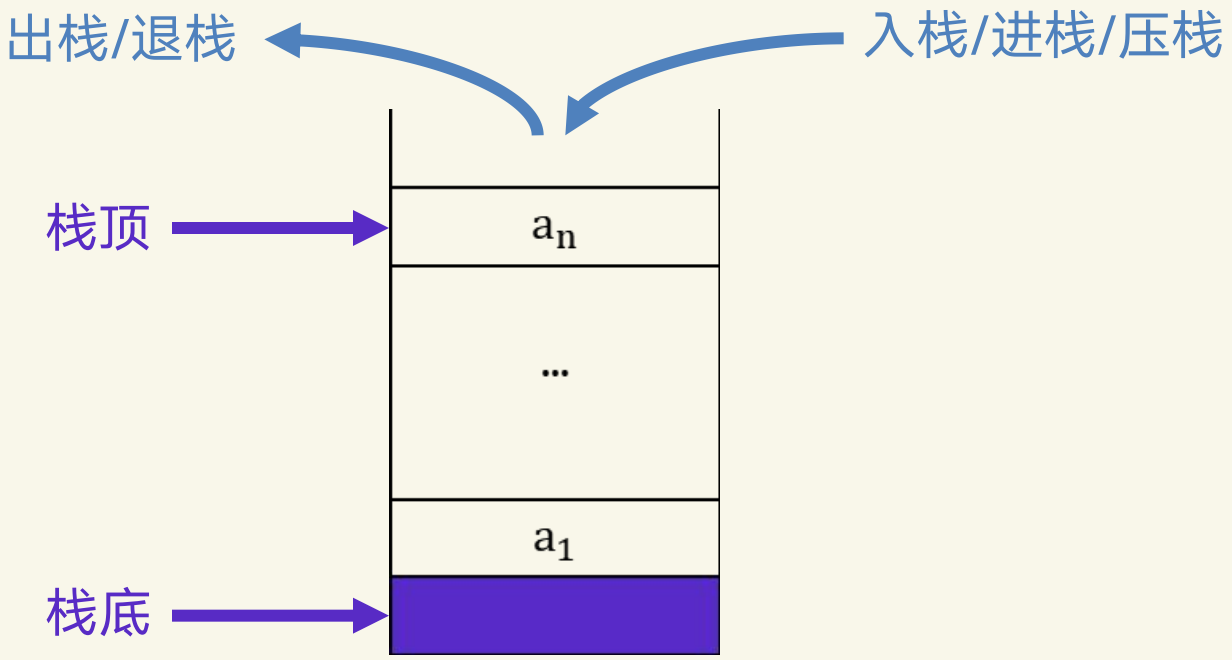
一对一

线性结构

- 数组
- 栈
- 队列
- 链表

STL栈

栈(stack)又名堆栈,是限定仅在栈顶进行插入和删除操作的线性结构,特点为:先进后出。



栈的模型

STL栈

栈的定义

头文件:

```
#include <stack>
```

格式:

```
stack<数据类型> 栈名;
```

举例:

```
stack<int> a;
stack<char> b;
stack<string> c;
```

STL栈

栈的操作

入栈

获取栈中元素个数

获取栈顶元素

出栈

判断栈是否为空

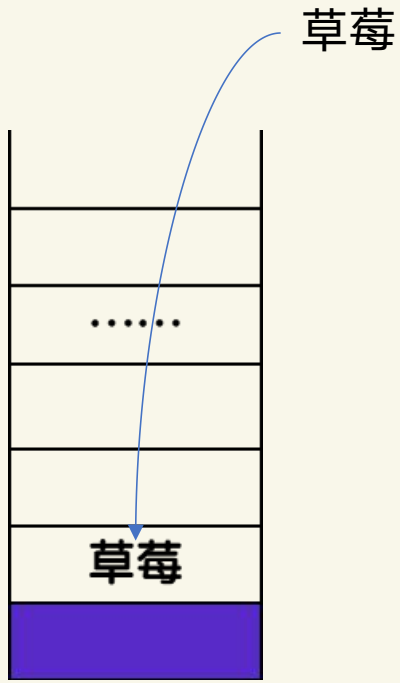
STL栈

入栈操作

栈名.push(数据); //将数据存入栈

举例：

```
stack<string> s;
s.push("草莓");
```



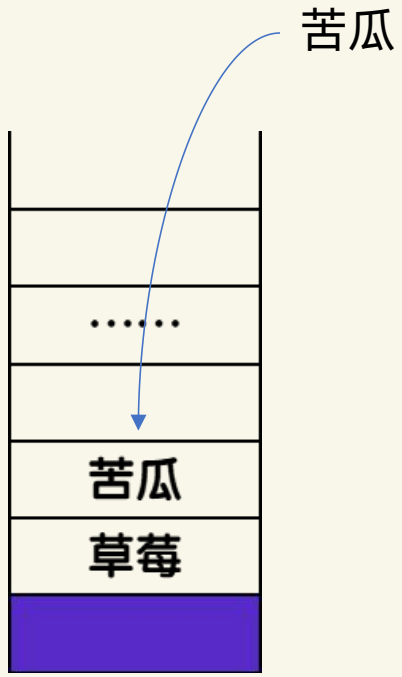
STL栈

入栈操作

栈名.push(数据); //将数据存入栈

举例：

```
stack<string> s;
s.push("草莓");
s.push("苦瓜");
```



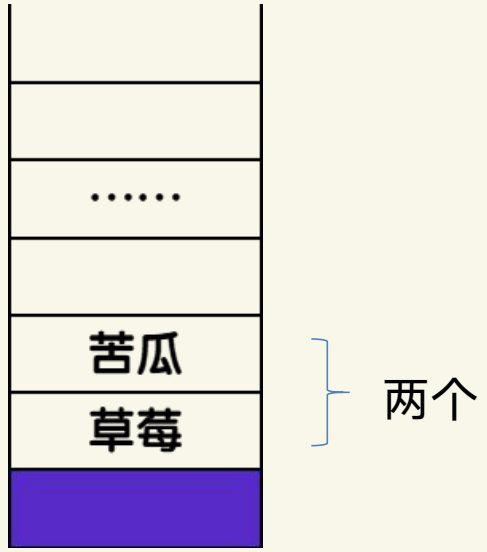
STL栈

获取栈中元素个数

栈名.size();

举例：

```
stack<string> s;
s.push("草莓");
s.push("苦瓜");
cout << s.size();
```



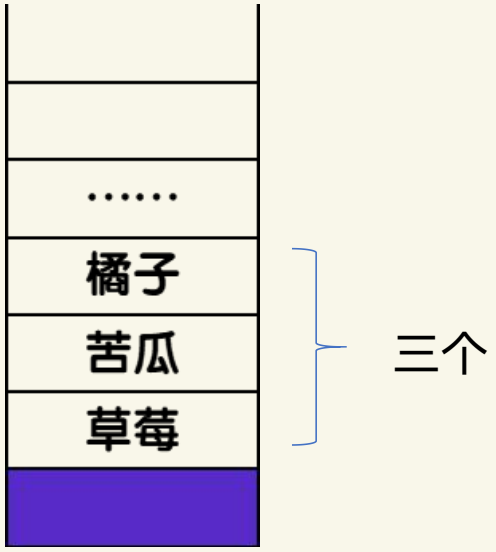
STL栈

获取栈中元素个数

栈名.size();

举例：

```
stack<string> s;
s.push("草莓");
s.push("苦瓜");
s.push("橘子");
cout << s.size();
```



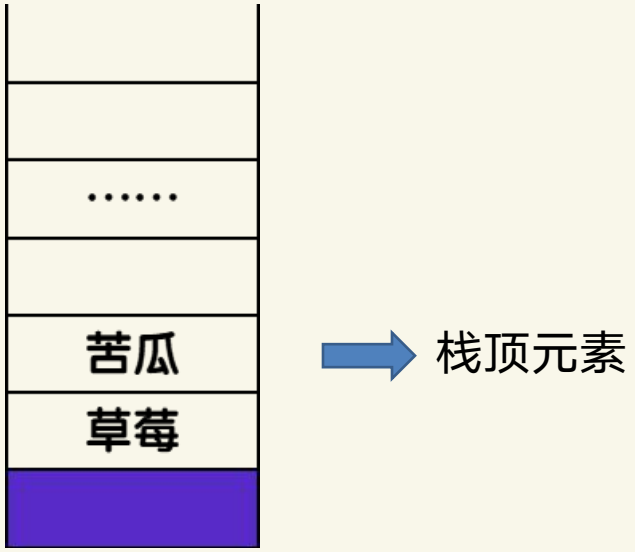
STL栈

获取栈顶元素

栈名.top();

举例：

```
stack<string> s;
s.push("草莓");
s.push("苦瓜");
cout << s.top();
```



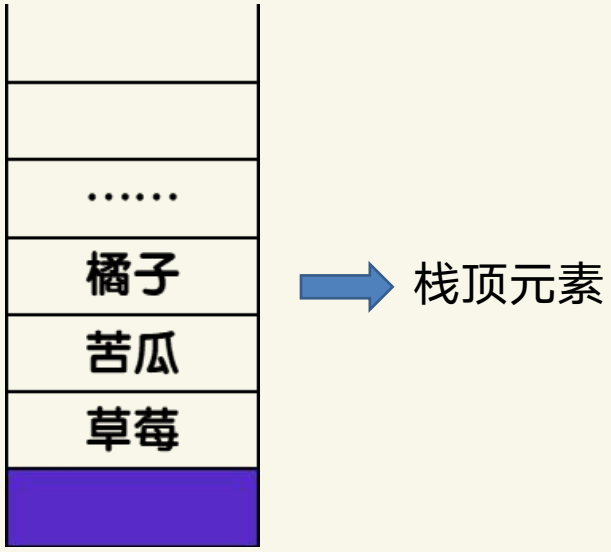
STL栈

获取栈顶元素

栈名.top();

举例：

```
stack<string> s;
s.push("草莓");
s.push("苦瓜");
s.push("橘子");
cout << s.top();
```



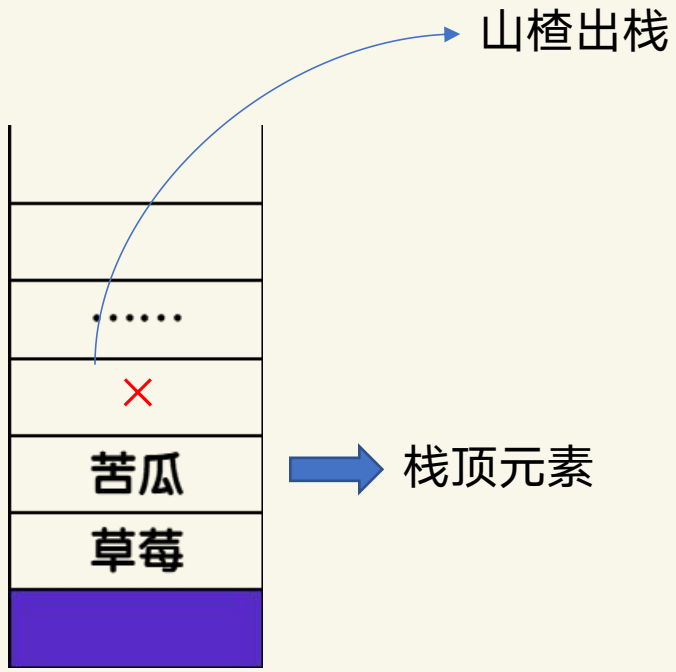
STL栈

出栈操作

栈名.pop(); //将栈顶元素移除

举例：

```
stack<string> s;
s.push("草莓");
s.push("苦瓜");
s.push("山楂");
s.pop();
```



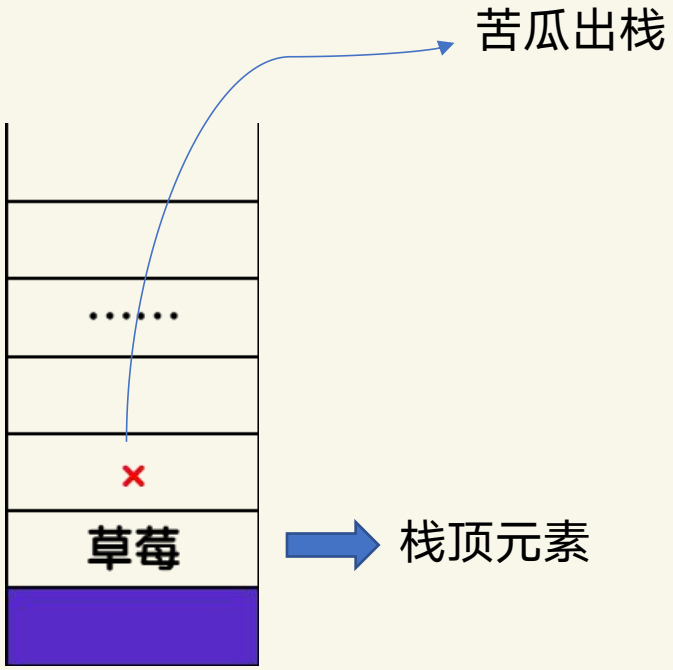
STL栈

出栈操作

栈名.pop(); //将栈顶元素移除

举例：

```
stack<string> s;
s.push("草莓");
s.push("苦瓜");
s.push("山楂");
s.pop();
s.pop();
```



STL栈

判断栈是否为空

```
栈名.empty(); //判断栈是否为空
```

是空,返回为1

否空,返回为0

STL栈

遍历栈

```
stack<string> s;
……
while (!s.empty())
{
    cout << s.top() << " ";
    s.pop();
}
```

STL栈

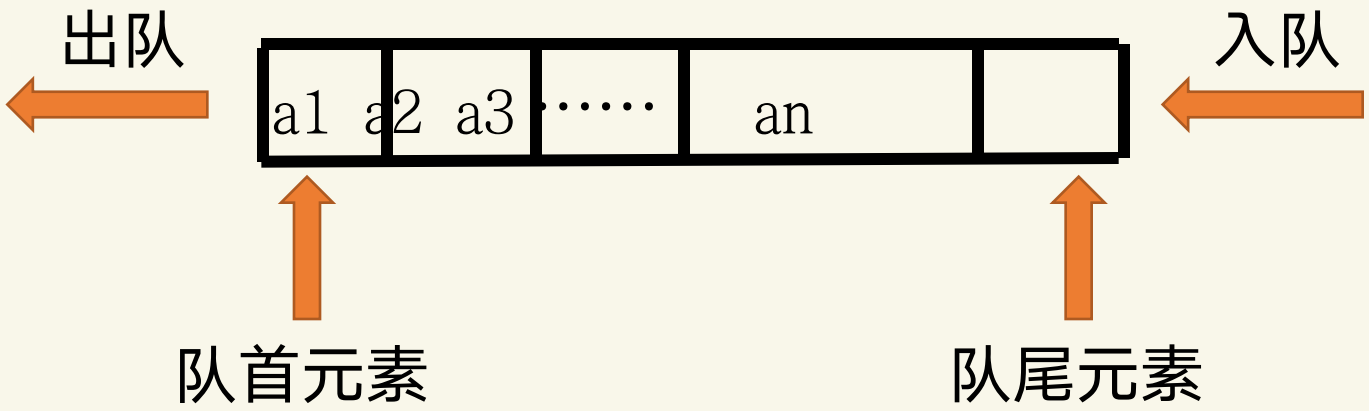
栈的操作	用法	举例
栈的定义	stack<数据类型> 栈名;	stack<string> s;
入栈操作	栈名.push(数据);	s.push("草莓");
栈中元素个数	栈名.size();	s.size();
栈顶元素	栈名.top();	s.top();
出栈操作	栈名.pop();	s.pop();
栈是否为空	栈名.empty();	s.empty();

Part 2

STL队列

STL队列

队列(queue)是存储数据的一种结构,特点为:先进先出。



STL队列

队列的定义

头文件:

```
#include <queue>
```

格式:

```
queue<数据类型> 队列名;
```

举例:

```
queue<int> a;
queue<char> b;
queue<string> c;
```

STL队列

队列的定义

举例：queue<string> q;

队列q示意图

队列中没有元素,此时处于空队列

STL队列

队列的操作

入队

获取队列中元素个数

获取队首元素

获取队尾元素

出队

判断队列是否为空

STL队列

入队操作

队列名.push(数据); //将数据存入队列

举例：

```
queue<string> q;
q.push("可多");
```

STL队列

入队操作

队列名.push(数据); //将数据存入队列

举例：

```
queue<string> q;
q.push("可多");
q.push("卡普");
```

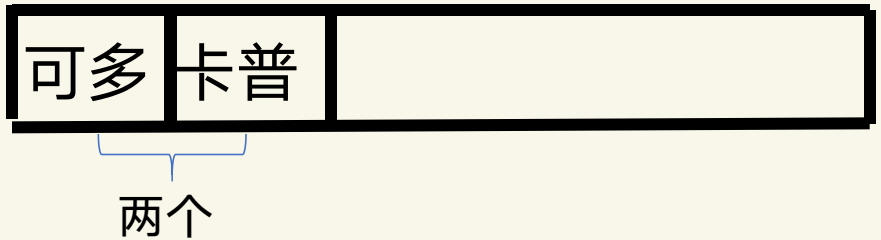
STL队列

获取队列元素个数

队列名.size(); //获取队列中元素个数

举例：

```
queue<string> q;
q.push("可多");
q.push("卡普");
cout << q.size();
```



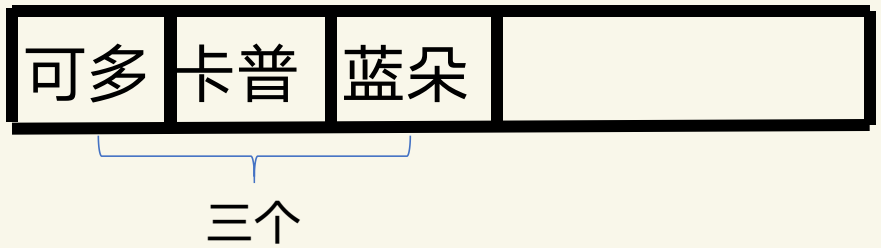
STL队列

获取队列元素个数

队列名.size(); //获取队列中元素个数

举例：

```
queue<string> q;
q.push("可多");
q.push("卡普");
q.push("蓝朵");
cout << q.size();
```



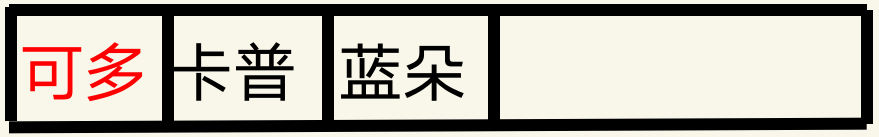
STL队列

获取队首元素

队列名.front(); //获取队首元素

举例：

```
queue<string> q;
q.push("可多");
q.push("卡普");
q.push("蓝朵");
cout << q.front();
```



STL队列

获取队尾元素

队列名.back(); //获取队尾元素

举例：

```
queue<string> q;
q.push("可多");
q.push("卡普");
cout << q.back();
```

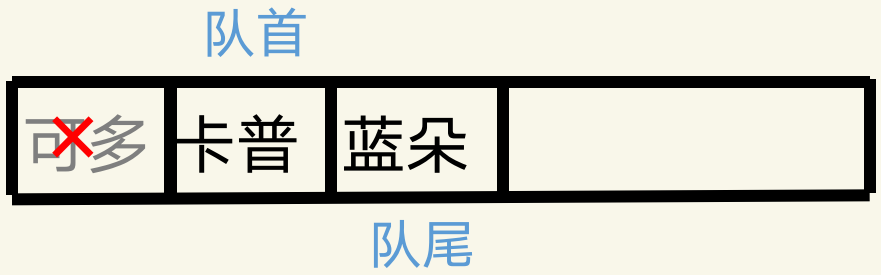
STL队列

出队操作

队列名.pop(); //将队首元素移除

举例：

```
queue<string> q;
q.push("可多");
q.push("卡普");
q.push("蓝朵");
q.pop();
```



STL队列

队列是否为空

队列名.empty(); //判断队列是否为空

是空,返回为1

否空,返回为0

STL队列

遍历队列

```
queue<string> q;
……
while (!q.empty())
{
    cout << q.front() << " ";
    q.pop();
}
```

STL队列

队列的操作	用法	举例
队列的定义	queue<数据类型> 队列名;	queue<string> q;
入队操作	队列名.push(数据);	q.push("可多");
获取队列中元素个数	队列名.size();	q.size();
获取队首元素	队列名.front();	q.front();
获取队尾元素	队列名.back();	q.back();
出队操作	队列名.pop();	q.pop();
判断队列是否为空	队列名.empty();	q.empty();

Part 3

STL优先队列

STL优先队列

普通的队列是一种先进先出的数据结构，元素在队尾追加，从队首删除。在优先队列中，元素被赋予优先级。当访问元素时，具有最高优先级的元素最先删除。优先队列具有最高级先出的特征。

STL优先队列

优先队列

头文件:

```
#include <queue>
```

格式:

```
priority_queue<数据类型> 队列名;
```

举例:

```
priority_queue<int> q;
```

STL优先队列

优先队列

数字越大优先级越高

方式一:priority_queue<int> q;

方式二:priority_queue<int,vector<int>,less<int> > q;

数字越小优先级越高

方式三:priority_queue<int,vector<int>,greater<int> > q;

STL优先队列

优先队列常用函数

①队列名.push(x):将x入队

②队列名.pop():队首元素出队

③队列名.top():获取队首元素

④队列名.empty():检测队列是否为空

⑤队列名.size():获取队列中元素个数

STL优先队列

优先队列普通队列对比

队列的操作	普通队列	优先队列
队列的定义	queue<string> q;	priority_queue<int> q;
入队操作	q.push("可多");	q.push(1);
获取队列中元素个数	q.size();	q.size();
获取队首元素	q.front();	q.top();
获取队尾元素	q.back();	无获取队尾元素函数
出队操作	q.pop();	q.pop();
判断队列是否为空	q.empty();	q.empty();

Class is over