C++
STANDARD TEMPLATE LIBRARY (STL)
(QUEUE)

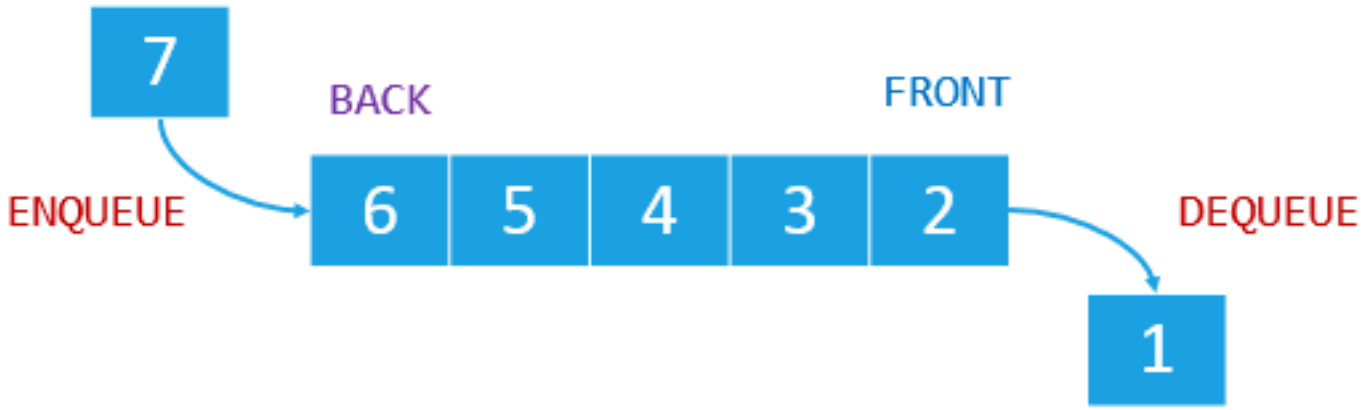
WASARA RODHETBHAI

DEPARTMENT OF COMPUTING • SILPAKORN UNIVERSITY

03/2024

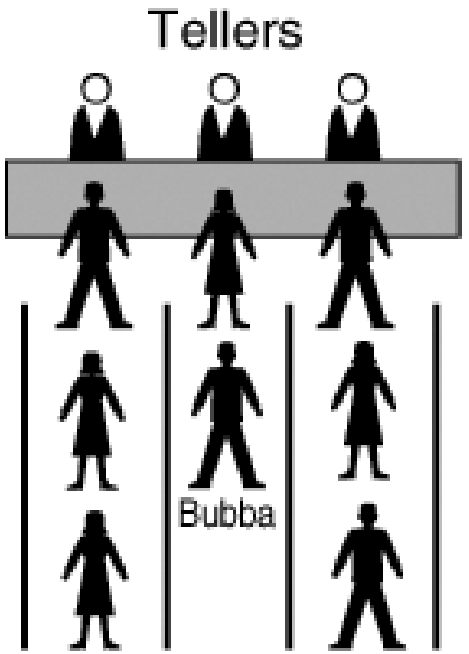
Queue (คิว)

โครงสร้างข้อมูลเชิงเส้นที่มีการจัดเก็บข้อมูลในรูปแบบของ First In First Out (FIFO)

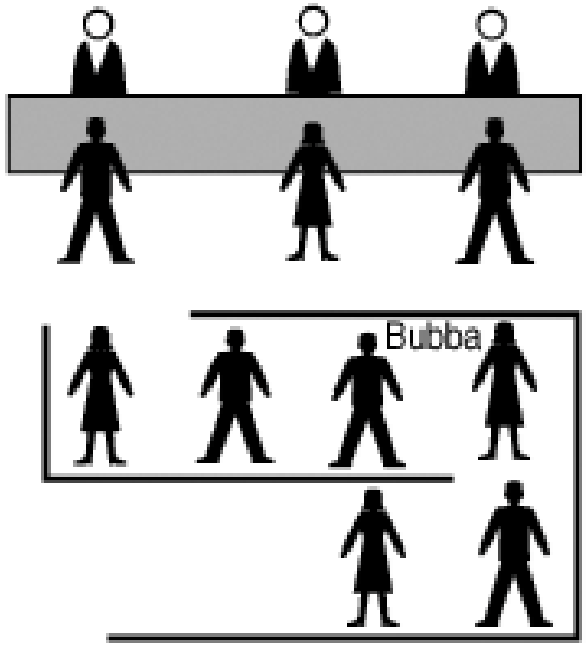


Queue ในชีวิตประจำวัน

Multiple Queues



Multiple Q's
Multiple Servers



Single Q
Multiple Servers

Queue ในระบบคอมพิวเตอร์

- Music Playlist
- Printer Queue

การสร้าง Queue Container

รูปแบบ

```
queue<data_type> queue_name;
```

ตัวอย่าง

```
queue<double> scores;
queue<string> names;
queue<int> ages;
```

การสร้าง Queue Container ของ Record (Struct)

ตัวอย่าง

```
struct Student {
        int no;
        string name;
        double score;
};
queue<Student> classroom;
```

การสร้าง Queue Container ของ Object

ตัวอย่าง

```
class Person {
    public:
        string name;
        int age;
        double income;
}
queue<Person> people;
```

Method ของ Queue Container

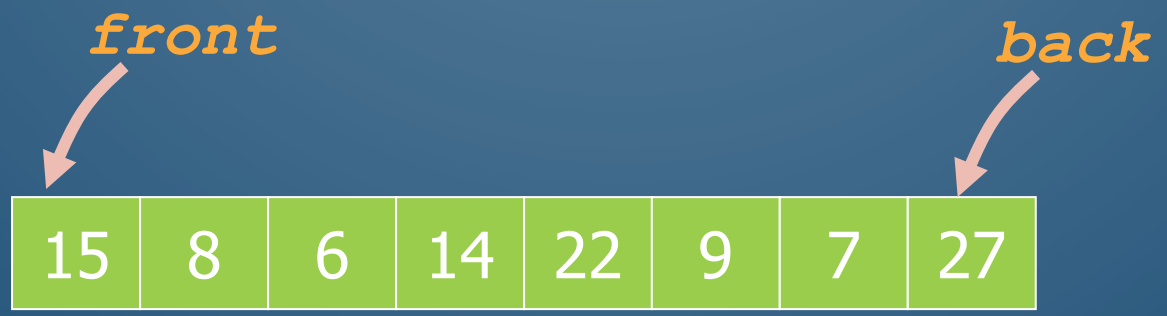
- เมธอดที่สำคัญของ Queue Container เช่น

```
push();          // นำข้อมูลเพิ่มเข้าต่อท้าย Queue
pop();           // ลบข้อมูลออกจากต้น Queue
front();         // คืนค่าข้อมูลในตำแหน่งต้น Queue
back();          // คืนค่าข้อมูลในตำแหน่งท้าย Queue
size();          // คืนค่าจำนวนข้อมูลใน Queue
empty();         // ตรวจสอบว่า Queue ว่างหรือไม่
```

การสร้าง Queue Container พร้อมค่าเริ่มต้น

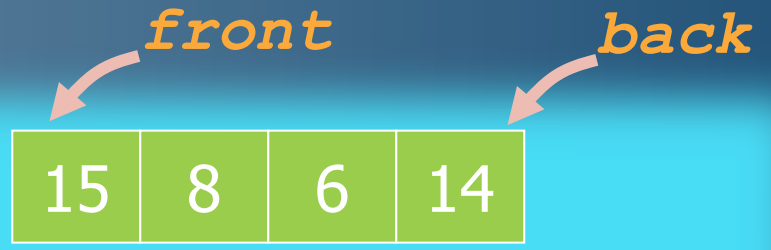
ตัวอย่าง

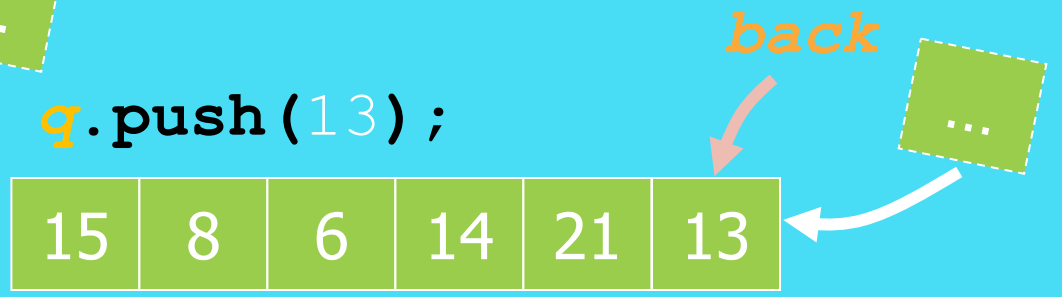
```
queue<int> q({15,8,6,14,22,9,7,27});
```

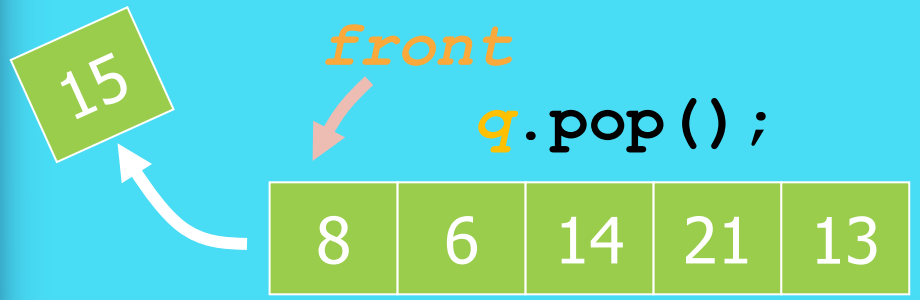


การนำข้อมูลเข้าและออกใน Stack Container

```
queue<int> q({15,8,6,14});
```



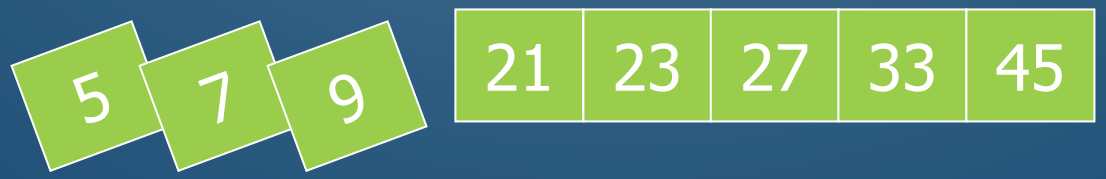




ข้อมูลใน Queue Container ณ ตำแหน่งต้น

ตัวอย่าง

```
queue<int> q({5,7,9,21,23,27,33,45});
int val=21;
while(!q.empty() && q.front()!=val) {
        cout << q.front() << " ";
        q.pop();
}
```



ข้อมูลใน Queue Container ณ ตำแหน่งท้ายคิว

ตัวอย่าง

```
list<int> lst{5,9,7,27,23,21,45,33,60};
queue<int> q;
for(int tmp : lst) {
        if(q.empty() || q.back()<tmp)
                q.push(tmp);
}
```

การแสดงข้อมูลใน Queue Container

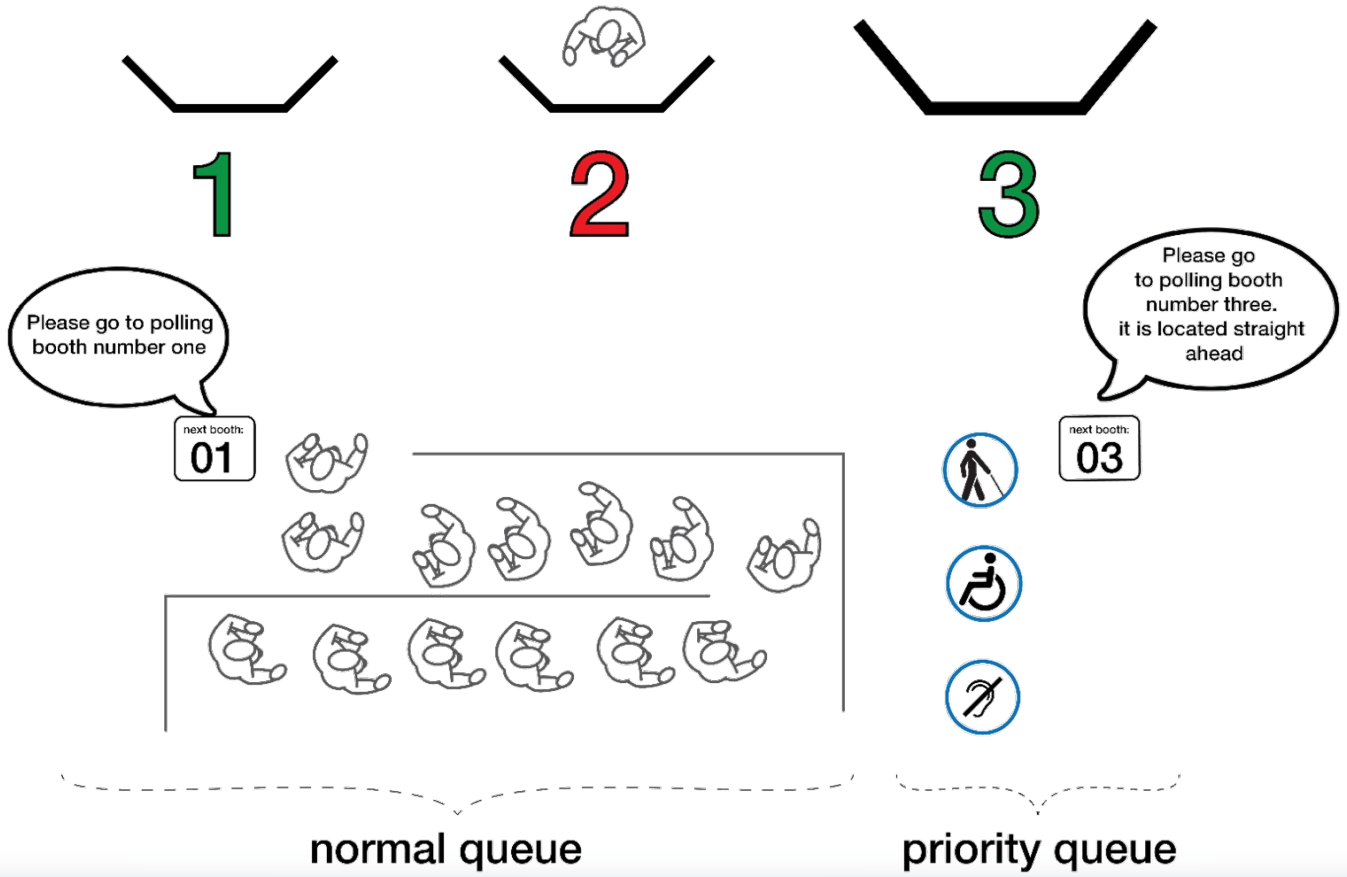
ตัวอย่าง

```
#include <iostream>
#include <queue>
using namespace std;

template <typename T>
void print(queue<T> q) {
    queue<T> temp;
    temp=q;
    while (!temp.empty()) {
        cout << temp.front() << " ";
        temp.pop();
    } cout << endl;
}
int main() {
    queue<int> q1({5,7,9,21,23,27,33,45});
    queue<string> q2({"CP","SC","SU","TH"});
    print(q1);
    print(q2);
}
```

Output:
5 7 9 21 23 27 33 45
CP SC SU TH

Priority Queues



การสร้าง Priority Queue Container

รูปแบบ

```
priority_queue<data_type> queue_name;
```

ตัวอย่าง

```
priority_queue<double> scores;
priority_queue<string> names;
priority_queue<int> ages;
```

ข้อมูลใน Priority Queue Container

ตัวอย่าง

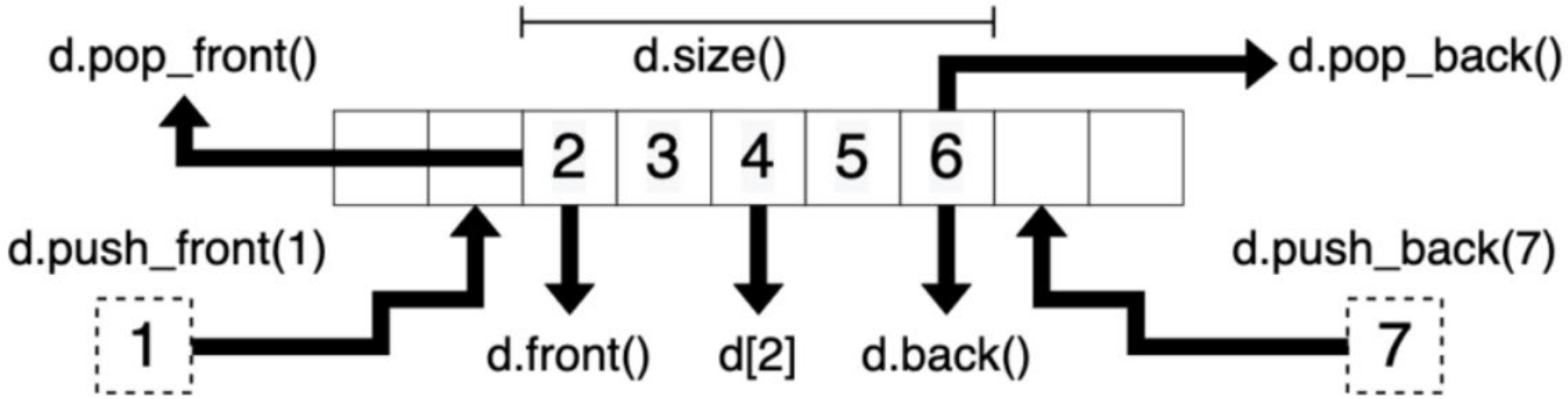
```
#include <iostream>
#include <queue>
#include <list>
using namespace std;

template <typename T>
void print(priority_queue<T> q) {
    priority_queue<T> temp;
    temp=q;
    while (!temp.empty()) {
        cout << temp.top() << " ";
        temp.pop();
    } cout << endl;
}

template <typename T>
priority_queue<T> add_queue(list<T> lst) {
    priority_queue<T> q;
    for(T t : lst)
        q.push(t);
    return q;
}
int main() {
    priority_queue<int> q1;  priority_queue<string> q2;
    list<int> lst1{5,7,21,9,21,27,27,45,33}; q1=add_queue(lst1);
    list<string> lst2{"CP","SC","SU","TH"}; q2=add_queue(lst2);
    print(q1); print(q2);
}
```

Output:
45 33 27 27 21 21 9 7 5
TH SU SC CP

Deque (Double Ended Queue)



Method ของ Deque Container

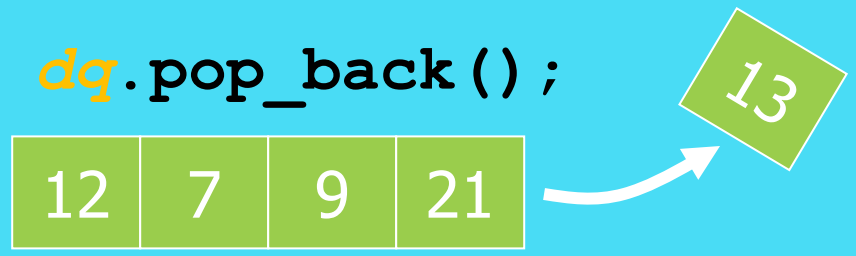
- เมธอดที่สำคัญของ Deque Container เช่น

```
push_back();          // นำข้อมูลเพิ่มเข้าต่อท้าย Deque
push_front ();        // นำข้อมูลเพิ่มเข้าต่อต้น Deque
pop_back();           // ลบข้อมูลออกจากท้าย Deque
pop_front();          // ลบข้อมูลออกจากต้น Deque
front();              // คืนค่าข้อมูลในตำแหน่งต้น Deque
back();               // คืนค่าข้อมูลในตำแหน่งท้าย Deque
size();               // คืนค่าจำนวนข้อมูลใน Deque
empty();              // ตรวจสอบว่า Deque ว่างหรือไม่
```

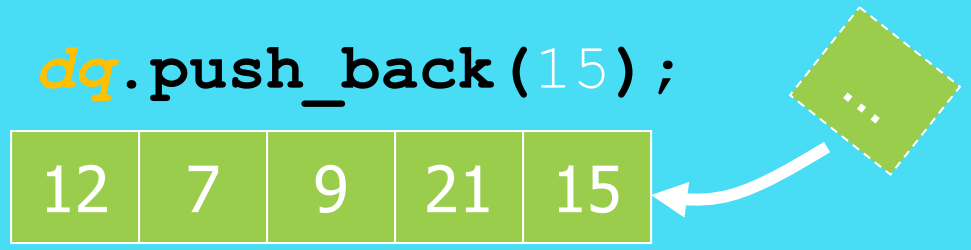
การเพิ่มและลบข้อมูลใน Deque Container

```
deque<int> dq({12,7,9,13});
```

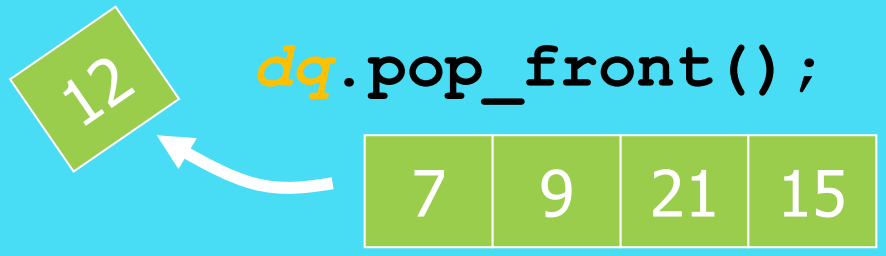

```
dq.pop_back();
```



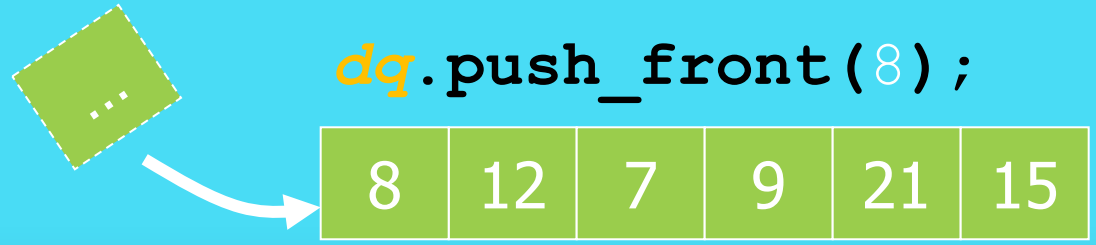

```
dq.push_back(15);
```




```
dq.pop_front();
```




```
dq.push_front(8);
```



การแสดงข้อมูลใน Deque Container

ตัวอย่าง

```
#include <iostream>
#include <queue>
using namespace std;

template <typename T>
void print(deque<T> q) {
    deque<T> temp;
    temp=q;
    while (!temp.empty()) {
        cout << temp.front() << " ";
        temp.pop_front();
    } cout << endl;
}
int main() {
    deque<int> q1({3,7,1,33,9,23,45,31});
    deque<string> q2({"CP","SC","SU","TH"});
    print(q1);
    print(q2);
}
```

Output:
3 7 1 33 9 23 45 31
CP SC SU TH

การแสดงข้อมูลใน Deque Container

ตัวอย่าง

```
#include <iostream>
#include <queue>
#include <algorithm>
using namespace std;

template <typename T>
void print(deque<T> q) {
    deque<T> temp;
    temp=q;
    while (!temp.empty()) {
        cout << temp.front() << " ";
        temp.pop_front();
    } cout << endl;
}
int main() {
    deque<int> q({3,7,1,33,9,23,45,31}); print(q);
    sort(q.begin()+1,q.begin()+5); print(q);
    sort(q.begin(),q.end()); print(q);
    q.erase(q.begin()+1); q.erase(q.end()); print(q);
    q.erase(q.begin()+2,q.begin()+4); print(q);
}
```

Output:

```
3 7 1 33 9 23 45 31
3 1 7 9 33 23 45 31
1 3 7 9 23 31 33 45
1 7 9 23 31 33
1 7 31 33
```