QUEUES

Problem Solving with Computers-II

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

int main(){
   cout<<"Hola Facebook\n";
   return 0;
}
```

How is PA01 going?

A. Done!

B. On track to finish.

C. Struggling.

D. Haven't started

The Queue Operations

Store keys in a queue

- A queue is like a line of people waiting for a bank teller.
- The queue has a **front** and a **rear**.

push()
insert

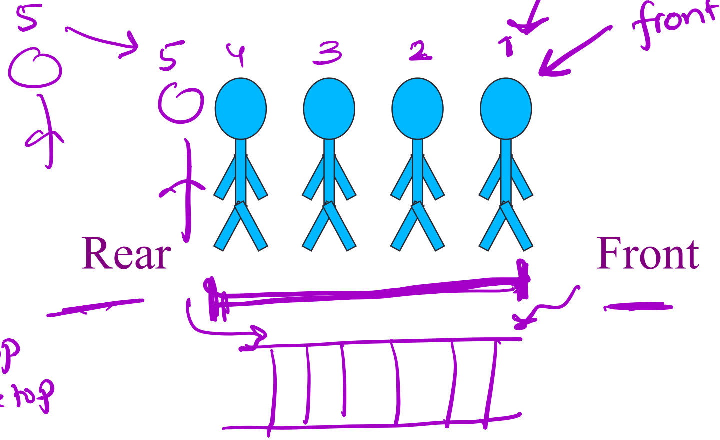
pop()
delete the front element

front()

Hawaii!
plane

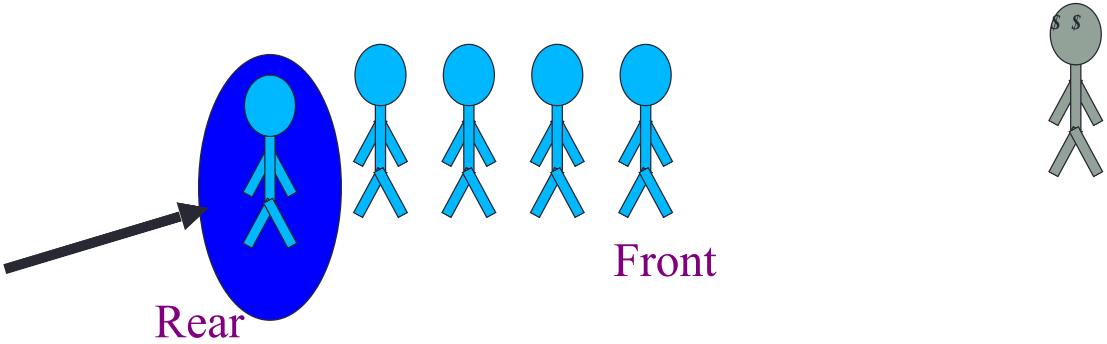
Stack

pop() // delete top
top() // check the top
push() // insert to the top



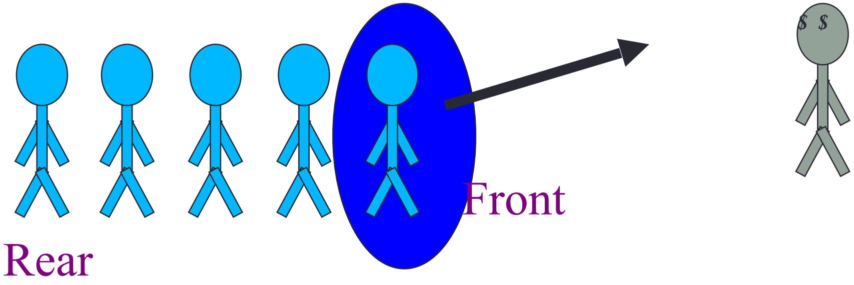
The Queue Operations

- New people must enter the queue at the rear. The C++ queue class calls this a **push**, although it is usually called an **enqueue** operation.



The Queue Operations

- When an item is taken from the queue, it always comes from the front. The C++ queue calls this a **pop**, although it is usually called a **dequeue** operation.



The Queue Class

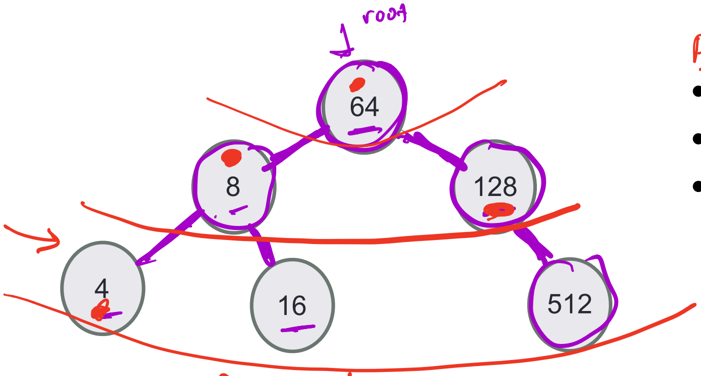
- The C++ standard template library has a queue template class.
- The template parameter is the type of the items that can be put in the queue.

```
template <class Item>
class queue<Item>
{
public:
        queue( );
        void push(const Item& entry);
        void pop( );
        bool empty( ) const;
        Item front( ) const;
        …
```

Breadth first traversal

BOA: (shortest path on unweighted graphe.)
minimum spanning

bread th first traversal



Algorithm:

- Take an empty Queue.
- Start from the root, insert the root into the Queue.
- Now while Queue is not empty,
  - ○ Extract the node from the Queue and insert left child all its children into the Queue. insert right child
  - ○ Print the extracted node.

Expected output

BFS: 64, 8, 128, 4, 16, 512

Preorder: 64, 8, 4, 16, 128, 512

Actual output

64, 8, 128, 4, 16, 512

rear

front

~~512~~, ~~16~~, ~~4~~, ~~128~~, ~~8~~, ~~64~~