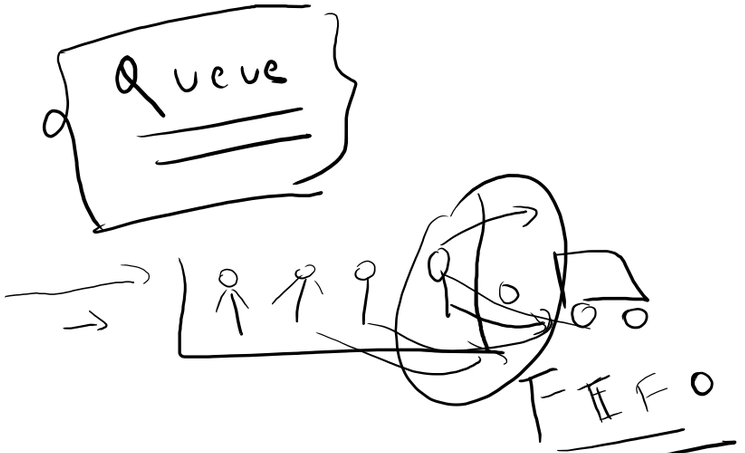
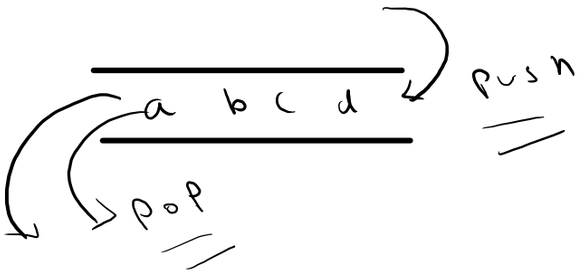
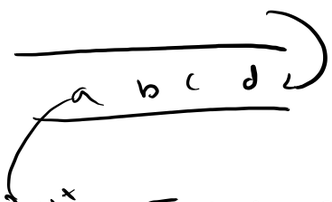
Queue



FIFO



Stores a list of items in which an item can be inserted at one end & removed from end.



Types of operation in queue:

enqueue

- Push() → from end

dequeue

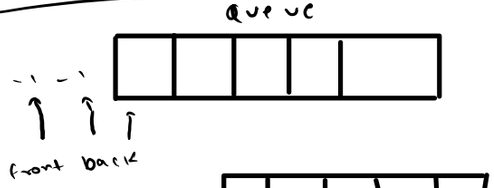
- Pop() → front
- front() → Peek()

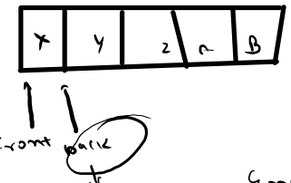
```
int main() {
    queue<string> q;
    q.push("abc");
    q.push("cde");
    while (!q.empty()) {
        cout << q.front() << endl;
        q.pop();
    }
}
```

output: 4 3 2 1

1 2 3 4

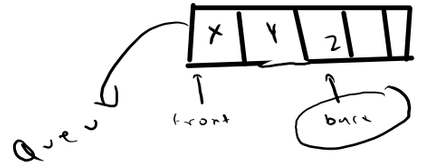
Operation in a queue:-





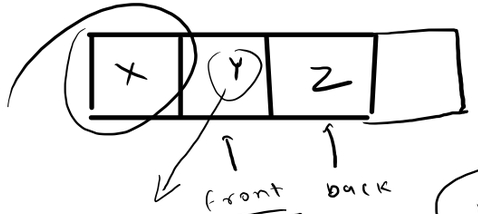
back++

Enqueue



Deque

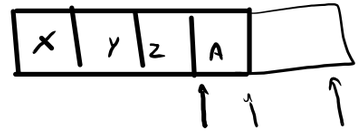
FIFO



Peek()

Peek() will output the element which is on the front

empty()



if (front == -1

front

if (front == -1 || front > back)

cout << "Queue empty"

overflow

Operations:

- enqueue()
- dequeue()
- Peek()
- empty()

Pointer Array implement Queue

C++ STL

STL

array [3]

{ C++ vectors }

```
#include<bits/stdc++.h>
using namespace std;
int main() int n
vector<int> V(n);
for (int i=0; i<n; i++)
{
cin >> v[i];
}
```

```
int main(){
vector<int> v1;
for(int i=1; i<=5; i++)
v1.push_back(i);
cout << " Print begin to end" << endl;
for( auto i= v1.begin(); i != v1.end();++i)
{
cout << *i << " ";
return 0;
}
```

array

(-, -, -7)

reverse(s.begin(), s.end())

palindrome

string s = " abba ";

string rev;

reverse(rev.begin(), rev.end)

if (s == rev)

yes palr