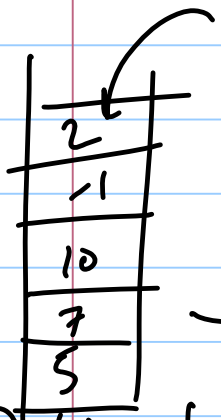
Note Title 11-10-2018

Oct 11, 2018

Stacks :- ADT (Abstract Data Type)



~~FL~~ :-

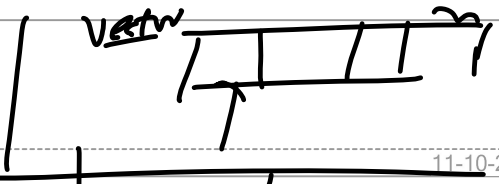
LIFO :-

means F1 → F2 → F3

Operations:

- push (value) → void
- pop () → value
- size() → size of stack
- isEmpty() → true/false
- peek() → value

O(1)
O(1)
O(1)
O(1)*
O(1)



Logistics :-

Friday :-

Extra Class :-

↳ ~~7:45 - 9:00~~

~~5:00 - 6:15~~

LIFO :-

Last in first out

start

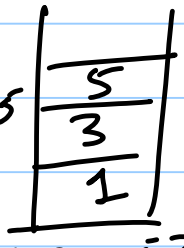
peek & pop on empty stack → ERROR

STACKS

```
#include "stack.h"

int main() {
    Stack<int> nums;   // {}
    nums.push(1);
    nums.push(3);
    nums.push(5);  // {1,3,5}
    cout << nums.peek() << endl; // 5
    cout << nums << endl;
    nums.pop(); // {1,3}
    cout << nums.isEmpty();
    int size = nums.size();
```

push/pop (top)



// {1,3,5}

if (nums.isEmpty())
{
}

Stack limitations / characteristics

```
Stack<int> S;
for(int i=0; i<S.size(); i++) {
    access S[i];
}
```

```
while( !S.isEmpty() ) {
    int value = S.pop();
}
```

```
int size = S.size();
for(int i=0; i<size; i++) {
    int val = S.pop();
}
```

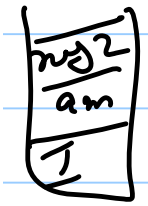
Exercise- "zyx are I" using stacks

Exercise- Reverse a sentence-

Input:- "I am xyz" (string)

prints ~~output~~:- "xyz am I"

```
void printSentenceReverse(const string & sentence){
    stack<string> wordStack;
    string word = "";
    for(char c: sentence){
        if(c == ' '){
            if(word.empty() != "")
                wordStack.push(word);
            word = "";
        }else{
            word = word + c;
        }
    }
}
```



"I am my2 ___"

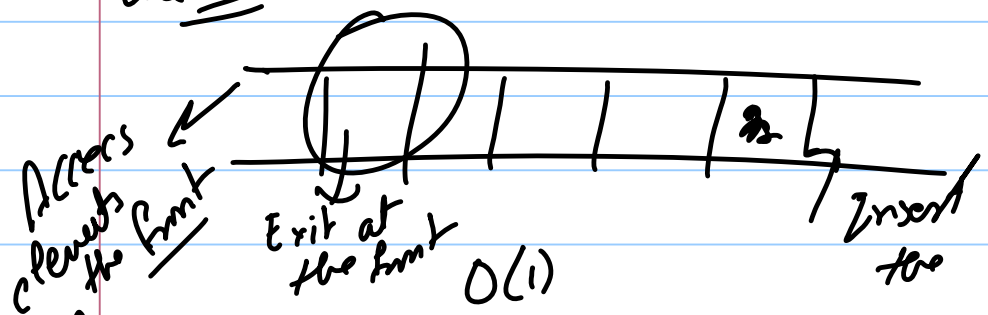
if

```
if(word.trim() != "") {
    wordStack.push(word);
}
}
while( !isEmpty() ! wordstack.isEmpty()) {
    counter wordstack.pop() <=
    <c" ";>
}
```

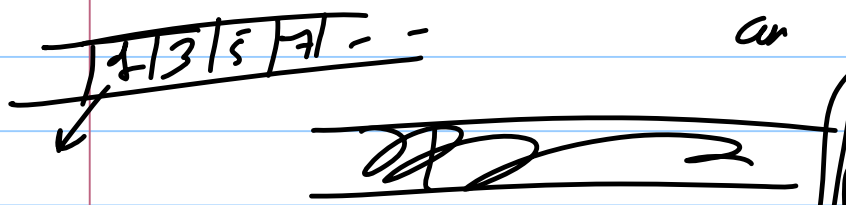
Exercise:- Do the same program using string streams

2- Data: Grid
1 . Data: Vectors Stacks()

Queues :- FIFO :- "First In First Out"



Note :- there is no index of an element as such



queue<int> nums;

O(1)

q.enqueue(value);
q.dequeue(); // Removes
q.peek(); // Doesn't remove
q.isEmpty(); // If empty
q.size();

```
queue<int> nums;
nums.enqueue(42);      // {42}
nums.enqueue(-3);      // {42, -3}
nums.enqueue(7);       // {42, -3, 7}

cout << nums.dequeue() << endl; // 42. {-3, 7}
cout << nums.peek() << endl;    // -3. {-3, 7}
cout << nums.dequeue() << endl; // -3. {7}
while (!nums.isEmpty()) {
    cout << nums.size() << endl; // 1.
```

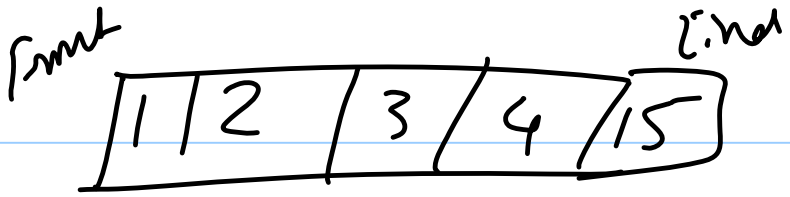
i=0 | 1
i=1 | 1 2
i=2 | 1 2 3

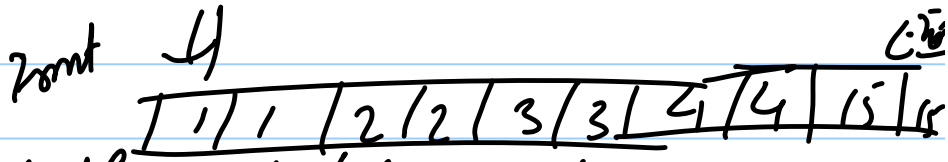
{1 2 3 4 5 6}
{X X 3 4 5 6}
{X X X 4 5 6}

```
Queue<int> queue;
for(int i=1; i<=6; i++) {
    queue.enqueue(i);
}                                   // {1,2,3,4,5,6}
int size = queue.size();
for(int i=0; i<size; i++) {         while(!queue.isEmpty()) {
    cout<<queue.dequeue()<<" ";
}
cout<<"queue size = "<<queue.size()<<endl;
```

(A) 1 2 3 4 5 6 {3 size=0 (B) 1 2 3 {4 5 6} size=3

(C) 1 2 3 4 5 6 {2 2 3 4 5 6} size=6 (D) NONE






```
void doubleElements(Queue<int> nums) {
~~for(int i=~~

}
```