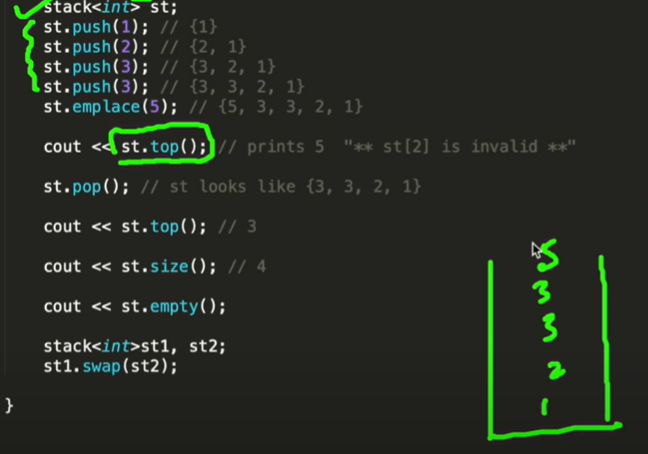
**Stack: it follows the concept of last in first out concept which mas that the element that enters last will come out first**

**Syntax: stack<data\_type>v\_name; or stack<data\_type>v\_name={elements};(not)**

****

**A STACK HAS 3 MAJOR FUNTIONS IN IT:**

**A] push() / emplace() :**

**Like the the push\_front() / emplace\_front() elements enters from the front side an easier understanding is done from the diagram above.**

**Syntax: st.push(); or st.emplace();**

**B] pop():**

**Removes element from the element but in stack the top elements gets erased first as we see in the diagram.**

**Syntax: st.pop();**

**C] top(): it shows element on the top**

**D] size(): it tells size of the element**

**E] empty(): tells if the stack is empty or not helps during the print the stack**

**NOTE:**

**Stack, unlike others, doesn’t have index value therefore we cannot do various operations that we were able to do in vector,list,deque,etc.**

**For example**

**We cannot do insert() erase() begin()/end()/rbegin()/rend() as these all are iterators that require reference to index values**

**In vector, list,deque we use for(auto it: v/s/dq/lst){ cout<<i<<””} to print the values of the corresponding.**

**This is not possible in stack as absence of index value as well as iterator.**

**The way to do it is create a temp stack, assign it to the original and then use pop() and top() to print the values.**

