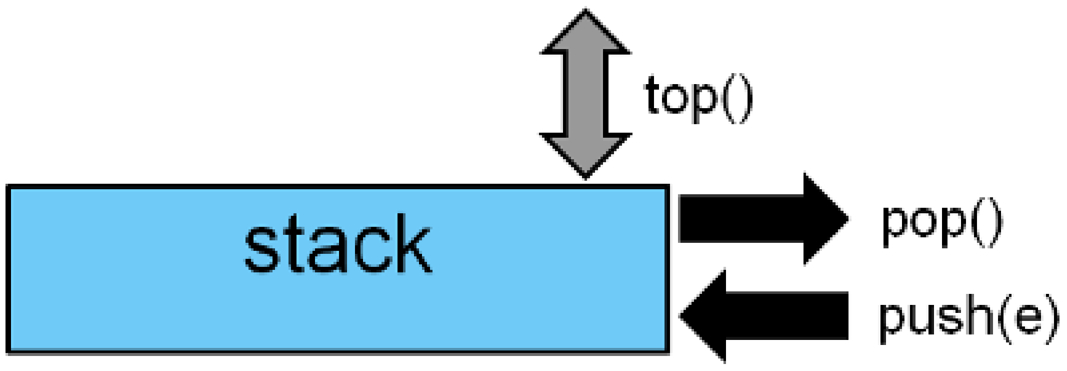
Stack

It's time to study the behavior of this popular data structure.



The std::stack follows the LIFO principle (**L**ast **I**n **F**irst **O**ut). The stack `sta`, which needs the header `<stack>`, has three special methods.

With `sta.push(e)` you can insert a new element `e` at the top of the stack, remove it from the top with `sta.pop()` and reference it with `sta.top()`. The stack supports the comparison operators and knows its size. The operations of the stack have constant complexity.

```
// stack.cpp
#include <iostream>
#include <stack>

int main(){

  std::stack<int> myStack;

  std::cout << myStack.empty() << std::endl;   // true
  std::cout << myStack.size() << std::endl;    // 0

  myStack.push(1);
  myStack.push(2);
  myStack.push(3);
  std::cout << myStack.top() << std::endl;     // 3

  while (!myStack.empty()){
    std::cout << myStack.top() << " ";
    myStack.pop();
  }                                            // 3 2 1
  std::cout << std::endl << myStack.empty() << std::endl;   // 1 (denotes true)
  std::cout << myStack.size() << std::endl;    // 0


  return 0;
}
```

std::stack