

Comparació d'eficiència de contenidors

Mètode	pro2::Stack	pro2::Queue	pro2::Heap	std::vector	std::deque	std::list	std::map	std::set
push(x)	$O(1)$	$O(1)$	$O(\log n)$					
pop()	$O(1)$	$O(1)$	$O(\log n)$					
top()	$O(1)$		$O(1)$					
front()		$O(1)$		$O(1)$	$O(1)$	$O(1)$		
back()				$O(1)$	$O(1)$	$O(1)$		
push_back(x)				$O(1)^*$	$O(1)^*$	$O(1)$		
push_front(x)					$O(1)^*$	$O(1)$		
pop_back()				$O(1)$	$O(1)$	$O(1)$		
pop_front()					$O(1)$	$O(1)$		
operator[]($\cdot$)				$O(1)$	$O(1)$		$O(\log n)$	
find($\cdot$)							$O(\log n)$	$O(\log n)$
insert(x)							$O(\log n)$	$O(\log n)$
insert(pos, x)				$O(n)$	$O(n)$	$O(1)^+$		
erase(x)							$O(\log n)$	$O(\log n)$
erase(pos)				$O(n)$	$O(n)$	$O(1)^+$		

Llegenda

- * Cost amortitzat: de tant en tant cal redimensionar el vector intern ($O(n)$), però en mitjana cada operació és $O(1)$.
- † Amb iterador: si ja tenim l'iterador a la posició, inserir/eliminar és $O(1)$. Trobar la posició és $O(n)$.

Notes

- Tots els contenidors tenen `size()` i `empty()` en $O(1)$.
- Per a `deque`, `vector` i `list`, `insert(pos, x)` i `erase(pos)` utilitzen un iterador com a posició.
- Per a `map` i `set`, `insert(x)` i `erase(x)` operen per clau/valor.