Java Wrapper Class

byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean
char	Character

Array

	C++	Java	Python3
new	int x = [0];	int[] x = new int[]{0};	x = [0]
length	sizeof(x)	x.length	len(x)

String

	C++	Java	Python3
new	char* s = "string";	String s = "string";	s = "string"
get	char c = s[0];	char c = s.charAt(0);	c = s[0];
length	strlen(s)	s.length()	len(s)
sub string	char* sub = malloc(10); strncpy(sub, s+begin, end-begin);	String sub = s.substring(begin, end);	s[begin, end]
check substring	if (strstr(sub, s))	if (s.contains(sub))	if sub in s:

Stack

	C++	Java	Python3
include	#include <stack>	import java.util.*;	
new	std::stack<int> s;	Stack<Integer> s = new Stack<Integer>();	s = []
push	s.push(i)	s.push(i);	s.append(i)
pop	s.pop();	popped = s.pop();	popped = s.pop()
top	s.top()	s.peak()	s[-1]
size	s.size()	s.size()	len(s)
check empty	if (s.empty())	if (s.empty())	if not s:

Queue

	C++	Java	Python3
include	#include <queue>	import java.util.*;	
new	std::queue<int> q;	Queue<Integer> q = new LinkedList<>();	q = []
push	q.push(i);	q.add(i);	q.append(i)
pop	q.pop()	popped = q.remove();	popped = q.pop(0)
front	q.front()	q.peek()	q[0]
size	q.size()	q.size()	len(q)
check empty	if (q.empty())	if (q.isEmpty())	if not q:

Heap

	C++	Java	Python3
include	#include <queue>	import java.util.*;	from queue import PriorityQueue
new min heap	std::priority_queue<int, std::vector<int>, std::greater<int>> h;	Queue<Integer> h = new PriorityQueue<>();	h = PriorityQueue()
new max heap	std::priority_queue<int> h;	Queue<Integer> h = new PriorityQueue<>(Collections.reverseOrder());	
push	h.push(i);	h.add(i);	h.put((1, "Harry"))
pop	h.pop();	Integer popped = h.poll();	popped = h.get()
top	h.top()	h.peek()	popped = h.get() h.put(popped)
size	h.size()	h.size()	h.qsize()
check empty	if (h.empty())	if (h.isEmpty())	if h.empty():

HashSet

	C++	Java	Python3
include	#include <unordered_set>	import java.util.*;	
new	std::unordered_set<int> s;	Set<Integer> s = new HashSet<>();	s = set()
add	s.insert(x);	s.add(x);	s.add(x)
delete	s.erase(x);	s.remove(x);	s.discard(x)
check in set	std::unordered_set<int>::const_iterator it = s.find(x); if (it != s.end())	if (s.contains(x))	if x in s:
size	s.size()	s.size()	len(s)
check empty	if (s.empty())	if (s.isEmpty())	if not s:

HashMap			
	C++	Java	Python3
include	#include <unordered_map>	import java.util.*;	
new	std::unordered_map<int, int> t;	Hashtable<Integer, Integer> t = new Hashtable<>();	d = {}
add	t.insert(std::make_pair<int, int>(key, val));	t.put(key, val);	d[key] = val
get	int val = t.at(key);	Integer val = t.get(key);	val = d[key]
delete	t.erase(key);	t.remove(key);	del d[key]

iterate		for (Map.Entry<Integer, Integer> set : t.entrySet()) {
		key=set.getKey();
		val=set.getValue();
		}	

iterate		t.forEach((key, val)->{...})	for key in d:
check in table	std::unordered_map<int, int>::const_iterator it = t.find(key); if (it != t.end())	if (t.containsKey(key))	if key in d.keys():
size	s.size()	s.size()	len(s)
check empty	if (s.empty())	if (s.isEmpty())	if not s: