Java Coding Interview Cheat Sheet

by Ptero via cheatography.com/190551/cs/39699/

Wrapper Class

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

Min Max

int min	Integer.MIN_VALUE // -2^31
int max	Integer.MAX_VALUE // 2^31-1
min	int min = Math.min(a, b);
max	int max = Math.max(a, b);
array min	int min = Collections.min(Arrays.asList(arr));
array max	int max = Collections.max(Arrays.asList(arr));

Math

abs	int abs = Math.abs(num);

Debug

print array	System.out.println(Arrays.toString(arr));
print 2D array	System.out.println(Arrays.deepToString(arr));
print map	System.out.println(Arrays.asList(map));

String

new	String str = "string";
get	char c = str.charAt(0);
length	int len = str.length();
check emty	if (str.isEmpty()) {...}

compare	if (s1.equals(s2)) {...}
	// or
	if (s1.compareTo(s2) == 0) {...}

String (cont)

sub string	String sub = str.substring(begin, end);
check substring	if (str.contains(sub)) {...}
split	String[] arr = str.split("[\\]+");
split char	import java.util.regex.Pattern; String[] arr = word.split(Pattern.quote(Character.toString('.')));
to char array	char[] arr = str.toCharArray();
from char array	char[] arr = {'s', 't', 'r'}; String str = new String(arr);
join string array	String[] arr = new String[] {"a","b","c"}; String joined = String.join(",", arr);
to string array	int[] arr = {1, 2, 3}; String[] strArr = new String[arr.length]; for (int i = 0; i < arr.length; ++i) { strArr[i] = String.valueOf(arr[i]); }
char to string	String str = Character.toString(',');

Array

new	int[] arr = new int[]{0};
	// or
	int[] arr = {1,2,3};
	// or
	int[] arr = new int[3];

Array (cont)

new 2D	int[][] arr = new int[height][width];
length	int len = arr.length;
fill	Arrays.fill(arr, val);
fill 2D	Arrays.stream(arr).forEach(a -> Arrays.fill(a, val));
compare	if (Arrays.equals(arr1, arr2)) {...}
traverse	for (int item : arr) {...}
sort ascending, inplace	import java.util.Arrays; Arrays.sort(arr);
sort descending, inplace	import java.util.*; Arrays.sort(arr, Collections.reverseOrder());
sum	int sum = Arrays.stream(arr).sum();
find index	int index = Arrays.asList(arr).indexOf(num);

Array List

include	import java.util.*;
new	List<Integer> list = new ArrayList<Integer>();
new 2D	List<List<Integer>> list = new ArrayList<List<Integer>>(size);
check empty	if (list.isEmpty()) {...}
compare	if (list1.equals(list2)) {...}
add to tail	list.add(num);
get	int num = list.get(index);
size	int size = list.size();
to array	Integer[] arr = new Integer[list.size()]; list.toArray(arr);

By **Ptero**
cheatography.com/ptero/

Not published yet.
Last updated 1st August, 2023.

Linked List

include	import java.util.*;
new	LinkedList<Integer> list = new LinkedList<Integer>();
new 2D	List<List<Integer>> list = new ArrayList<List<Integer>>(size);
add to head	list.addFirst(num);

add to tail	list.add(num);
	// or
	list.addLast(num);

get head	int num = list.getFirst();
get tail	int num = list.getLast();

delete head	list.remove();
	// or
	list.removeFirst();

delete tail	list.removeLast();
size	int size = list.size();

Stack

include	import java.util.*;
new	Stack<Integer> stack = new Stack<>();
push	stack.push(num);
pop	int num = stack.pop();
top	int num = stack.peak();
size	int size = stack.size();
check empty	if (stack.empty()) {...}
to array	Integer[] arr = new Integer[stack.size()]; stack.toArray(arr);

Queue

include	import java.util.*;
new	Queue<Integer> queue = new LinkedList<Integer>();
push	queue.add(num);
pop	int num = queue.poll();
get front	int num = queue.peek();

Queue (cont)

size	int size = queue.size();
check empty	if (queue.isEmpty()) {...}

Heap

include	import java.util.*;
new min heap	Queue<Integer> minHeap = new PriorityQueue<Integer>();
new max heap	Queue<Integer> maxHeap = new PriorityQueue<Integer>(Collections.reverseOrder());
push	heap.add(num);
pop	int num = heap.remove();
get top	int num = heap.peek();
size	int size = heap.size();
check empty	if (heap.isEmpty()) {...}
add array	heap.addAll(Arrays.asList(arr));

Hash Set

include	import java.util.*;
new	Set<Integer> set = new HashSet<>();
add	set.add(num);
delete	set.remove(num);
check in set	if (set.contains(num)) {...}
iterate	for (int num : set) {...}
size	int size = set.size();
check empty	if (set.isEmpty()) {...}
to array	String arr = new String[set.size()]; set.toArray(arr);

Hash Table

include	import java.util.*;
new	Hashtable<Integer, Integer> table = new Hashtable<>();
set	table.put(key, val);
get	int val = table.get(key);
get default	int val = table.getOrDefault(key, default);
delete	table.remove(key);
check key in table	if (table.containsKey(key)) {...}
check val in table	if (table.containsValue(val)) {...}
iterate	for (Integer key : table.keySet()) {...}
size	int size = table.size();
check empty	if (table.isEmpty()) {...}
keys to array	String[] arr = new String[map.size()]; map.keySet().toArray(arr);
values to array	Integer[] arr = new Integer[map.size()]; map.values().toArray(arr);