Java sheet sheet

Array

- sort: `Arrays.sort(array)`
- sort with comparator:
    - return 1 means l1 > l2

```
Comparator<int[]> selfDefineComparator = new Comparator<int[]>() {
        public int compare(int[] l1, int[] l2) {
                if(l1[0]==l2[0]){
                    return l1[1]>l2[1]?1:-1;
                }else{
                    return l1[0]>l2[0]?1:-1;
                }
            }
        };
Arrays.sort(list, selfDefineComparator);
```

- `arrays.clone()`

String

- `result.toLowerCase()`
- `result.charAt(i)`
- Start include, end exclude. `result.substring(start, end)`
- StringBuilder
    - constructor: `StringBuilder builder = new StringBuilder()`
    - `builder.append('a')`
    - `builder.toString()`
- `word.toCharArray()` <=> `String word = new String(wordArray)`

Integer

- `Integer.toString(i)` <=> `Integer.parseInt(string)`
- `Integer.MAX_VALUE`
-

HashMap

- `containsKey(Object key)`
- `containsValue(Object key)`
- `get(Object key)`
- `size()`
- `put(K key, V value)`
- `remove(Object key)`
- List to array: `result.toArray(new String[result.size()])`

HashSet

- `set.iterator().next();`
- `set.add(num)`
- `set.remove(num)`
-

ArrayList

- concat `array1.addAll(array2);`
- copy `List<Integer> newList = new ArrayList<>(oldList);`

LinkedList

- add(E e)
- add(int index, E element)
- addAll(Collection<> c)
- addFirst(E e)
- addLast(E e)
-

Stack

- `stack.empty()`
- `stack.peek()`
- `stack.pop()`
- `stack.push()`

Interface Queue

- FIFO: `LinkedList`
    - `Queue<Integer> fifo = new LinkedList<>()`
    - add to end `fifo.offer(e)`
    - retrive and remove the head, return null if empty `fifo.poll()`
    - `fifo.peek()`

Priority Queue

- `PriorityQueue<Map.Entry<Integer, Integer>> pq = new PriorityQueue<> ((a,b)->(b.getValue()-a.getValue()));`
- `pq.offer(e)`
- `pq.poll()`
- `pq.peek()`

Collection

- `collections.sort(list)`

Random

- `Random rand = new Random()`

- `rand.nextInt(bount)` [0,bound)
- `rand.nextDouble()` [0,1)

easy error

- Variable not defined
- string double quote
- duplicate define
- corner case
- bonary search overflow