public LRU(int cacheSize) {

this.cacheSize = cacheSize;

int capacity = (int) Math.ceil(cacheSize / hashLoadFactory) + 1;

map = new LinkedHashMap<K, V>(capacity, hashLoadFactory, true) {

private static final long serialVersionUID = -1;

@Override

protected boolean removeEldestEntry(Map.Entry<K, V> eldest) {

return size() > LRU.this.cacheSize;

}

};

}

// offer、poll、peek

PriorityQueue<Integer> minHeap1 = new PriorityQueue<>(3, Integer::compareTo);

PriorityQueue<Integer> minHeap2 = new PriorityQueue<>(3, Comparator.naturalOrder());

PriorityQueue<Integer> maxHeap = new PriorityQueue<>(3, Comparator.reverseOrder());

Stack：push、pop、peek、empty

Queue：add offer remove poll peek

ArrayBlockingQueue：add offer put remove,poll,take 【异常、bool、阻塞】