1. (최대이진힙 구현, binary max-heap) 다음은 최대이진힙의 삽입, 삭제 동작을 구현한 코드이다. 아래 코드를 입력하고 실행하면서 이진힙의 구현법을 익히시오. (Reference: https://en.wikipedia.org/wiki/Binary_heap, CC-BY-SA, https://algs4.cs.princeton.edu/24pq/MaxPQ.java.html, GPLv3)

```
public class Test {
	public static void main(String[] args) {
		int		v[]={4, 2, 9, 5, 7, 5, 8, 10, 15};
		SimpleHeap		heap=new SimpleHeap();
		for (int i = 0; i < v.length; i++) {
			heap.add(v[i]);
			System.out.println(heap);
		}
		for (int i = 0; i < v.length; i++) {
			System.out.println(heap.remove()+"=>"+heap);
		}
	}
}
public class SimpleHeap {
	int		last=-1, MaxHeapSize=4;
	int		heap[]=new int[MaxHeapSize];
	private void resize() {
		MaxHeapSize*=2;
		heap=Arrays.copyOf(heap, MaxHeapSize);
	}
	private void swap(int m, int n) {
		int		temp=heap[m];
		heap[m]=heap[n];
		heap[n]=temp;
	}
	public void add(Integer data) {
		if(last+1==MaxHeapSize) resize();
		heap[++last]=data; // heap의 마지막 노드 다음 위치에 새 자료 삽입
		for (int child=last; child>0; ) { // 마지막 노드를 heapify-up
			int		parent=(child-1)/2;
			if(heap[child]<=heap[parent]) break; // 최대힙조건 검사
			swap(child, parent); // 조건 불만족 시 교환
			child=parent;
		}
	}
	public int remove() {
		if(last<0) throw new RuntimeException("heap empty");
		int		data=heap[0]; // root 노드 자료 추출
		heap[0]=heap[last--]; // 마지막 노드 root로 이동 & 크기 1 감소
		for (int parent=0, child=2*parent+1; child<=last; parent=child, child=2*parent+1) { // root를 heapify-down
			if(child<last && heap[child]<heap[child+1]) child++;
			if(heap[child]<=heap[parent]) break;
			swap(child,parent);
		}
		return data;
	}
	@Override
	public String toString() {
		return Arrays.toString(heap);
	}
}
```

2. (우선순위큐, 자바클래스 활용) 다음은 자바클래스 PriorityQueue를 이용하여 배열 내 정수 모음에 대해 최소힙을 생성한 후 우선순위가 높은 순으로 추출하여 출력하는 코드이다. PriorityQueue는 priority heap을 구현한 자바클래스이다. 아래 코드를 입력하고 실행하면서 자바에서 우선순위큐의 사용법을 익히시오.

```
public class Test {
	public static void main(String[] args) {
		Integer	v[]={4, 2, 9, 5, 7, 5, 8, 10, 15};
		List<Integer> list=Arrays.asList(v);
		PriorityQueue<Integer>		pq=new PriorityQueue<>(list); //  최소힙  생성
		for (int i = 0; i<v.length; i++) {
			System.out.println(pq.remove()+"=>"+pq);
		}
	}
}
```

3. (우선순위큐, 자바클래스 활용) 다음은 자바클래스 PriorityQueue를 이용하여 배열 내 정수들을 우선순위큐에 삽입하는 방식으로 최소힙을 생성한 후 우선순위가 높은 순으로 추출하여 출력하는 코드이다. PriorityQueue는 priority heap을 구현한 자바클래스이다. 아래 코드를 입력하고 실행하면서 자바에서 우선순위큐의 사용법을 익히시오.

```
public class Test {
	public static void main(String[] args) {
		int		v[]={4, 2, 9, 5, 7, 5, 8, 10, 15};
		// PriorityQueue<Integer>		pq=new PriorityQueue<>(); // min-heap
		PriorityQueue<Integer>		pq=new PriorityQueue<>(Collections.reverseOrder()); // max-heap
		for (int i = 0; i < v.length; i++) {
			pq.add(v[i]);
			System.out.println(pq);
		}
		for (int i = 0; i<v.length; i++) {
			System.out.println(pq.remove()+"=>"+pq);
		}
	}
}
```

4. (이진힙, 균형이진탐색트리, 정렬 기반 방법 비교) 다음은 배열 v에 저장된 N개 임의 정수(int) 중 크기 순으로 최하위 10개 정수를 출력하는 코드로, 동일한 작업을 이진힙, 균형이진탐색트리, 배열정렬을 사용한 각 경우로 나누어 소요 시간을 함께 출력한다.

```
public class Test {
	public static void main(String[] args) {
		int		N=1000000;
		ArrayList<Integer>	list=new ArrayList<>();
		for (int i = 0; i < N; i++) list.add(i);
		Collections.shuffle(list);
		Integer	v[]=new Integer[N];
		for (int i = 0; i < list.size(); i++) v[i]=list.get(i);
		System.out.println("Start ...");
		long		start;

		start=System.currentTimeMillis();
//		PriorityQueue<Integer>		pg=new PriorityQueue<>();
//		for (int i = 0; i < v.length; i++) pg.add(i);
		PriorityQueue<Integer>		pg=new PriorityQueue<>(Arrays.asList(v));
		for (int i = 0; i < 10; i++) System.out.print(pg.remove()+" ");
		System.out.println(" ... "+(System.currentTimeMillis()-start)+" ms (PriorityQueue)");

		start=System.currentTimeMillis();
		TreeSet<Integer>	map=new TreeSet<>();
		for (int i = 0; i < v.length; i++) map.add(v[i]);
		for (int i = 0; i < 10; i++){
			Integer	n=map.first();
			System.out.print(n+" ");
			map.remove(n);
		}
		System.out.println(" ... "+(System.currentTimeMillis()-start)+" ms (TreeMap)");

		start=System.currentTimeMillis();
		Arrays.sort(v);
		for (int i = 0; i < 10; i++) System.out.print(v[i]+" ");
		System.out.println(" ... "+(System.currentTimeMillis()-start)+" ms (Arrays.sort())");
	}
}
```

References

- C로 쓴 자료구조론 (Fundamentals of Data Structures in C, Horowitz et al.). 이석호 역. 사이텍 미디어. 1993.
- 쉽게 배우는 알고리즘: 관계 중심의 사고법. 문병로. 한빛아카데미. 2013.
- C언어로 쉽게 풀어 쓴 자료구조. 천인국 외 2인. 생능출판사. 2017.
- 김윤명. (2008). 뇌를 자극하는 Java 프로그래밍. 한빛미디어.
- 남궁성. 자바의 정석. 도우출판.
- 김윤명. (2010). 뇌를 자극하는 JSP & Servlet. 한빛미디어.