Heap

Implementasi Max Heap

Pada max heap, setiap node kecuali root, memenuhi property: ***A[parent(i)]>=A[i]***. Pada bagian ini dicontohkan implementasi dari max heap.

1. Buatlah kelas MaxHeap dengan atribut-atributnya seperti gambar di bawah ini:

```
public class MaxHeap{
    private int maxHeap[];
    private int heapSize;
    private int length;
```

2. Tambahkan dua buah consructor:
   a. Constructor pertama dengan parameter *length*, yaitu ukuran maksimum dari max heap.

```
public MaxHeap(int length){
    this.heapSize=0;
    this.length=length;
    this.maxHeap=new int[length+1];
}
```

   b. Constructor kedua dengan parameter sebuah array bertipe int.

```
public MaxHeap(int[] arr){
    this.heapSize=arr.length;
    this.length=arr.length;
    this.maxHeap=new int[length+1];
    for(int i=0; i<length;i++){
        this.maxHeap[i+1]=arr[i];
    }
}
```

3. Tambahkan method getLeft, getRight, dan getParent yang masing-masing berfungsi untuk mendapatkan indeks dari anak kiri, anak kanan, dan parent dari node *i* pada heap.

```
private int getLeft(int i){
    return i<<1;
}

private int getRight(int i){
    return (i<<1)|1;
}

private int getParent(int i){
    return i>>1;
}
```

Note: Pada implementasi di atas, indeks dari max heap dimulai dari 1. Oleh karena itu, untuk membuat max heap berukuran *length*, perlu disediakan sebuah array dengan ukuran *length+1*, sehingga dapat diisi sebanyak *length* buah elemen pada array tersebut (dari indeks 1,2,3...,length). Dampak lainnya adalah *insert* sebuah key hanya bisa dilakukan selama ***heapSize<=length***, dan element baru disimpan di lokasi ke-*heapsize+1*.

Tambahkanlah method-method berikut ini:

a. maxHeapify(int i)
b. buildMaxHeap()
c. insertKey(int key)
d. increaseKey(int i, int key)
e. extractMax()

HeapSort

Implementasikanlah heapsort! Anda hanya perlu menambahkan sebuah method lagi di dalam kelas MaxHeap yang telah dibuat pada soal nomor 1.