Praktikum Desain dan Analisis Algoritma: Priority Queue

A. Implementasi Max Priority Queue dengan Unordered Array

1. Buatlah sebuah kelas bernama **PriorityQueue** dengan atribut:
   - Integer array[]
   - int length sebagai kapasitas maksimum dari priority queue.
   - int size sebagai ukuran dari priority queue.

   Pada kelas PriorityQueue, tambahkan constructor dengan parameter **int length** seperti pada contoh di bawah ini:

```
1  public class PriorityQueue{
2      private Integer array[];
3      private int size;
4      private int length;
5
6      PriorityQueue(int length){
7          this.length=length;
8          this.size=0;
9          this.array=new Integer[length];
10     }
```

2. Tambahkan method **insert(Integer x)**. Method ini memasukkan key *x* ke dalam priority queue. Karena priority queue pada modul ini diimplementasikan dengan unordered array, maka *x* dimasukkan ke dalam priority queue tanpa perlu mengurutkannya. Key *x* ditambahkan pada indeks ke-*size*. Key *x* dapat ditambahkan ke dalam priority queue selama priority queue belum penuh (ukurannya tidak sama dengan length). Perhatikan contoh implementasi di bawah ini:

```
public boolean insert(Integer x){
    if(size<length){
        array[size]=x;
        size++;
        return true;
    }
    else return false;
}
```

3. Method berikutnya yang perlu ditambahkan adalah method **max()**. Method ini mengembalikan elemen maksimum di dalam priority queue tanpa menghapus elemen maksimum tersebut.

```
19      public Integer max(){
20          int i,max;
21          if(this.size==0){
22            return null;
23          }
24
25          else{
26            max=this.array[0];
27            for(i=1;i<this.size;i++){
28                if(this.array[i]>max){
29                    max=this.array[i];
30                }
31            }
32            return max;
33          }
34      }
```

4. Selanjutnya, tambahkan method **extractMax()**. Berbeda dengan method **max()**, method **extractMax()** akan mengembalikan elemen maksimum di dalam priority queue dan menghapusnya dari priority queue. Untuk menghapus sebuah elemen dari unordered priority queue, kita cukup me-*replace* elemen tersebut dengan elemen terakhir pada priority queue, kemudian *size*-nya dikurangi 1.

```
36      public Integer extractMax(){
37          int i,max,temp;
38          if(this.size==0){
39             return null;
40          }
41          else{
42            max=0;
43            for(i=1;i<this.size;i++){
44               if(this.array[i]>this.array[max]){
45                  max=i;
46               }
47            }
48            temp=this.array[max];
49            this.array[max]=this.array[size-1];
50            size--;
51            return temp;
52          }
53      }
```

5. Method terakhir yang perlu ditambahkan adalah method **increaseKey(int i, int key)**. Method ini mengubah isi dari priority queue pada indeks ke-*i* dengan key yang baru. Karena implementasi yang digunakan adalah unordered array, maka penggantian key tidak mempengaruhi letak elemen pada array.

```
55      public void increaseKey(int i, int key){
56          this.array[i]=key;
57      }
58  }
```

B. Latihan

Implementasikanlah priority queue dengan **ordered linked list.** Di dalam kelas priority queue yang anda buat, harus terdapat method insert, max dan extractMax. Anda tidak perlu membuat method increaseKey. **Tidak boleh menggunakan LinkedList yang disediakan oleh Java.**