```
void swim(int k) {
```

```
void sink(int k) {

        int x;
        int temp = heap[k];
        while(getChild(k, 1) <= size-1) {
            x = getMaxChild(k);
            if(heap[x] > temp) {
                heap[k] = heap[x];
            }
            else break;
            k = x;
        }
        heap[k] = temp;
}
```

The worst case is O(d log $_2$ N
This is where x = the number of children
This causes the time complexity

```
DaryHeap heap = new DaryHeap(3);
        for(int k = 0; k<20; k++) {
            heap.insert(k);
        }


        int[] sorted = heap.daryHeapsort();
        for(int i = 0; i < sorted.length; i++) {
                System.out.print(sorted[i]+", ");
```

Worst case for dary-heapsort is O (N log N)

It has to go through the recursive call for the array.