```
/**
 *
 */
package taskk1.heap.pack;

/**
 * @author SAMSUNG
 *
 */

public class PowerHeap {

	private int[] heap;
	private int size;
	//Constructor for size
	public PowerHeap(int maxSize) {
		this.size=maxSize;
		this.heap = new int[this.size+1];
		this.heap[0] = Integer.MAX_VALUE;
		this.size = 0;
	}
	//Swapping Nodes
	private void Swap(int i, int j) {
		int temp;
		temp = heap[i];
		heap[i] = heap[j];
		heap[j] = temp;
	}
	//Removing top element by swapping and, the code for
downHeapify function is
	private void DownHeapify(int k) {
		int greatest = k;
		int leftIdx = 2*k;
		int rightIdx = 2*k + 1;
		int exp = 2;

		if (leftIdx <= heap.length && heap[leftIdx] >
heap[greatest]){
			greatest = leftIdx;
			leftIdx = (int)(Math.pow(leftIdx, exp));
		}

		if (rightIdx <= heap.length && heap[rightIdx] >
heap[greatest]) {
			greatest = rightIdx;
			rightIdx = (int)(Math.pow(leftIdx, exp));
		}

		if (greatest != k) {
			Swap(k, greatest);
			DownHeapify(greatest);
		}
	}

	//Function to swap nodes until it reaches root or
```

```
satisfies heap property
            private void heapifUp(int k) {
                  while (heap[k] > heap[k/2]) {
                        Swap(k , k/2);
                        k = k/2;
                  }
            }


            //Function to print and display the heap
            private void print(){
                  for (int i = 1; i <= size/2; i++) {
                        System.out.printf("Root: %d  ::: LC: %d, RC:
%d %s", heap[i], heap[i*2], heap[i*2+1], System.lineSeparator());
                        System.out.printf("Root: %d ::: LC: %d, RC: %d
%s", (int)(Math.pow(heap[i], 2)), (int)(Math.pow(heap[i*2], 2)),
(int)(Math.pow((heap[i*2+1]), 2)), System.lineSeparator());
                  }
            }

            //Function to Adding new element
            public void insert(int x) {
                  heap[++size] = x;
                  heapifUp(size);
            }

            //Function to delete the top element
            public int pop() {
                  int max = heap[1];
                  heap[1] = heap[size--];
                  DownHeapify(1);

                  return max;
            }

            public int peek() {
                  return heap[1];
            }

            //Main Method
            public static void main(String[] args) {
                  PowerHeap maxHeap = new PowerHeap(5);
                  maxHeap.insert(1);
                  maxHeap.insert(4);
                  maxHeap.insert(2);
                  maxHeap.insert(5);
                  maxHeap.insert(6);


                  maxHeap.print();
                  System.out.println("The max is " + maxHeap.pop());
            }
      }
```

```
//   Output of the PowerHeap.java is

   /*    Root: 6  ::: LC: 5, RC: 2
         Root: 36 ::: LC: 25, RC: 4
         Root: 5  ::: LC: 1, RC: 4
         Root: 25 ::: LC: 1, RC: 16
         The max is 6     */
```