class PowerOfTwoMaxHeap {

constructor(powerExponent) {

if (powerExponent < 0 || powerExponent > 10) {

throw new Error("powerExponent must be between 0 and 10 for performance.");

}

this.powerExponent = powerExponent;

this.numChildren = 1 << powerExponent; // 2^powerExponent

this.heap = [];

}

insert(value) {

this.heap.push(value);

this.\_heapifyUp(this.heap.length - 1);

}

popMax() {

if (this.heap.length === 0) {

throw new Error("Heap is empty.");

}

const maxValue = this.heap[0];

const lastValue = this.heap.pop();

if (this.heap.length > 0) {

this.heap[0] = lastValue;

this.\_heapifyDown(0);

}

return maxValue;

}

\_heapifyUp(index) {

let current = index;

while (current > 0) {

const parentIndex = Math.floor((current - 1) / this.numChildren);

if (this.heap[current] > this.heap[parentIndex]) {

this.\_swap(current, parentIndex);

current = parentIndex;

} else {

break;

}

}

}

\_heapifyDown(index) {

let current = index;

const size = this.heap.length;

while (true) {

let maxIndex = current;

for (let i = 1; i <= this.numChildren; i++) {

const childIndex = this.numChildren \* current + i;

if (

childIndex < size &&

this.heap[childIndex] > this.heap[maxIndex]

) {

maxIndex = childIndex;

}

}

if (maxIndex !== current) {

this.\_swap(current, maxIndex);

current = maxIndex;

} else {

break;

}

}

}

\_swap(i, j) {

const temp = this.heap[i];

this.heap[i] = this.heap[j];

this.heap[j] = temp;

}

isEmpty() {

return this.heap.length === 0;

}

printHeap() {

console.log(this.heap);

}

}