Este es un ejemplo de un Heap especial llamado máximum Heap o maxheap, el cual cumple con la condición de que todo nodo almacena un valor mayor o igual al de sus hijos. La definición es equivalente para un minheap.

El tiempo de inserción es logarítmico (se recorren log n nodos para llegar de la raíz a una hoja, la altura del árbol es logarítmica porque es binario y árbol completo) , es decir, si existen n elementos en la estructura la inserción se realiza en a lo mas k log n pasos (cuando decimos orden log n estamos asumiendo que hay una constante que multiplica por log n, generalmente pequeño).

Peor de los casos

17, 42, 89, 23, 67, 8, 95, 34, 51, 76

La extracción se hace en tiempo constante y siempre se extrae el elemento de mejor prioridad. Luego de extraer hay que dejar la estructura ordenada, lo cual toma nuevamente tiempo logarítmico, esto se denota como O(log n) y se lee: “el tiempo es orden log n”.

1. Extraemos siempre el elemento de mayor prioridad de nuestro Heap X[1].
2. Como debemos de dejar el Heap ordenado, ponemos el ultimo elemento del Heap, u = X[n], en la primera posición: X[1] = u, luego se decrementa n en uno y “hundimos” u intercambiándolo con el hijo de mayor prioridad hasta que quede bien ubicado.

La operación que “hunde” X[1] hasta dejarlo bien ubicado se llama heapify.

Maxheapify

Raíz = x[i]

M = el hijo de la raíz. El primer hijo esta en la posición 2

I = para recorrer

X = como un ordenante para ir intercambiando