|  |
| --- |
| 0hnQGAuNPo4wojnr77c2x8cWiaT70FGHBZ2h-3__FYQ.png |
| Heap Sort |
| Data Structures Made Easy |
|  |

Dublin City University

# 1. *Heap Sort*

class heap\_Sort{

private static int left\_Sort(int element){

return 2 \* element + 1;

}

private static int right\_Sort(int element){

return 2 \* element + 2;

}

private static void swap(int [] array, int a, int b){

int temp = array[a];

array[a] = array[b];

array[b] = temp;

}

public static void max\_Heap(int [] heap\_Array, int heap\_Integer, int heap\_Size){

int left;

int right;

int maximum;

int temporary;

left = left\_Sort(heap\_Integer);

right = right\_Sort(heap\_Integer);

if(left < heap\_Size && heap\_Array[left] > heap\_Array[heap\_Integer])

maximum = left;

else

maximum = heap\_Integer;

if(right < heap\_Size && heap\_Array[right] > heap\_Array[maximum])

maximum = right;

if(maximum != heap\_Integer){

swap(heap\_Array, heap\_Integer, maximum);

max\_Heap(heap\_Array, maximum, heap\_Size);

}

}

public static void build\_Heap(int [] heap\_Array, int heap\_Size){

for(int index = heap\_Size / 2; index >= 0; index--)

max\_Heap(heap\_Array, index, heap\_Size);

}

public static void heap\_Sort(int [] heap\_Array, int heap\_Size){

build\_Heap(heap\_Array, heap\_Size);

for(int index = heap\_Size - 1; index > 0; index--){

swap(heap\_Array, index, 0);

heap\_Size--;

max\_Heap(heap\_Array, 0, heap\_Size);

}

}

public static void main(String [] args){

System.out.print("Enter the number of elements: ");

int size = Console.readInt();

int [] heap = new int[size];

System.out.print('\n' + "Enter the elements: ");

for(int index\_1 = 0; index\_1 < size; index\_1++){

int element = Console.readInt();

heap[index\_1] = element;

}

heap\_Sort(heap, size);

System.out.print('\n' + "The sorted list is: ");

for(int index\_2 = 0; index\_2 < size; index\_2++)

System.out.print(heap[index\_2] + " ");

}

}