```
HEAPIFY(A, i)
1 l ← LEFT(i)
2 r ← RIGHT(i)
3 if l < heap-size[A] and A[l] > A[i]
4     then largest ← l
5     else largest ← i
6 if r < heap-size[A] and A[r] > A[largest]
7     then largest ← r
8 if largest != i
9     then exchange A[i] ←→ A[largest]
10         HEAPIFY(A,largest)
```

```
BUILD-HEAP(A)
1  heap-size[A] ← length[A]
2  for i ← length[A]/2 downto 1
3      do HEAPIFY(A, i)
```

```
HEAPSORT(A)
1  BUILD-HEAP(A)
2  for i ← length[A] downto 2
3      do exchange A[1] ←→ A[i]
4         heap-size[A] ← heap-size[A] -1
5         HEAPIFY(A, 1)
```

```
HEAP-EXTRACT-MAX(A)
1  if heap-size[A] < 1
2     then error "heap underflow"
3  max ← A[1]
4  A[1] ← A[heap-size[A]]
5  heap-size[A] ← heap-size[A] - 1
6  HEAPIFY(A, 1)
7  return max
```

```
HEAP-INSERT(A,key)
1  heap-size[A] ← heap-size[A] + 1
2  i ← heap-size[A]
3  while i > 1 and A[PARENT(i)] < key
4     do A[i] ← A[PARENT(i)]
5        i ← PARENT(i)
6  A[i] ← key
```