Sorting

Data and File Structures Laboratory

`http://www.isical.ac.in/~dfslab/2018/index.html`

Heap sort

Reference: Sedgewick and Wayne, Section 2.4

```
void heapsort(void *a, int N, size_t element_size,
              int (*comparator)(void *, int, int))) {
  int k;
  HEAP h;

  h.element_size = element_size;
  h.num_allocated = h.num_used = N;
  h.array = a;
  h.comparator = comparator;
  /* Make heap out of array */
  for (k = N/2; k >= 1; k--)
      swapDown(&h, k);
  /* Sort by successive deleteMax */
  while (h.num_used > 1) {
      swap(&h, 1, h.num_used); // move max to end
      h.num_used--;
      swapDown(&h, 1);
  }
}
```

NOTE: Indexing from 1!

Insertion sort

```
for (i = 1; i < n; i++) {
    key = A[i];
    /* Find the right place for A[i] in A[0 .. i-1] */
    for (j = i-1; j >= 0 && A[j] > key; j--)
        A[j+1] = A[j];
    A[j+1] = key;
}
```

Insertion sort

```
for (i = 1; i < n; i++) {
    key = A[i];
    /* Find the right place for A[i] in A[0 .. i-1] */
    for (j = i-1; j >= 0 && A[j] > key; j--)
        A[j+1] = A[j];
    A[j+1] = key;
}
```

Use `memcpy()` here.

Bubble sort

```
for (i = 0; i < n-1; i++)
    for (j = 0; j < n-i-1; j++)
        if (A[j+1] < A[j])
            swap(A, j, j+1);
```

Merge sort

```
void msort(int A, int beginning, int end) {
     if (beginning < end) {
        middle = (beginning + end) / 2;
        msort(A, beginning, middle);
        msort(A, middle+1, end);
        merge(A, beginning, middle, end);
     }
}
```

Merge sort

```
void merge(A, b, m, e) {
    int i, j, k;
    /* allocate space for auxiliary array B */
    for (i = b, j = m+1, k = 0; i <= m && j <= e; )
        if (A[i] < A[j])
           B[k++] = A[i++];
        else if (A[i] > A[j])
           B[k++] = A[j++];
        else
           B[k++] = A[i++], B[k++] = A[j++];
    while (i <= m) B[k++] = A[i++];
    while (j <= e) B[k++] = A[j++];
    assert(...);
}
```

Quick sort

Partition:

```
i = -1; j = n-1; v = a[n-1];
while (1) {
    do i = i+1; while (a[i] < v);
    do j = j-1; while (a[j] > v);
    if (i >= j) break;
    x = a[i]; a[i] = a[j]; a[j] = x;
}
x = a[i]; a[i] = a[n-1]; a[n-1] = x;
```