**public** **class** SequentialSearchST<Key, Value> {

**int** N; // number of key-value pairs

Node first; // the linked list of key-value pairs

**public** Value get(Key key) {

// Nodes now store (key and value)

**class** Node {

Key key;

Value value;

Node next;

}

Node n = first;

**while** (n != **null**) {

**if** (key.equals (n.key)) {

**return** n.value;

}

n = n.next;

}

**return** **null**; // not present

}

**public** **void** put(Key key, Value val) {

Node n = first;

**while** (n != **null**) {

**if** (key.equals (n.key)) {

n.value = val;

**return**;

}

n = n.next;

}

// add as new node at beginning

first = **new** Node (key, val, first);

N++;

}

**public** **void** delete(Key key) {

Node prev = **null**;

Node n = first;

**while** (n != **null**) {

**if** (key.equals (n.key)) {

**if** (prev == **null**) { // no previous? Must have been first

first = n.next;

} **else** {

prev.next = n.next; // have previous one link around

}

**return**;

}

prev = n; // don't forget to update!

n = n.next;

}

}

}

/\*\* I can’t believe I’m using this Bubble Sort as example. \*GASP\*. \*/

**public** **static** **void** sort(Comparable[] a) {

**int** N = a.length;

**boolean** swapped;

**do** {

swapped = **false**;

**for** (**int** idx = 1; idx < N; idx++) {

**if** (*less*(a[idx], a[idx-1])) {

*exch*(a, idx, idx-1);

swapped = **true**;

}

}

N--;

} **while** (swapped);

}