Algorithm 1 Insertion Process {API}

```
1: function INSERT(key, value)
2:
3: Input:   key → String format key
4:          value → String format value
5: Output: Status → Status of operation
6: procedure
7:     blocks ← new Arraylist[bCOUNTER]
8:     counters ← new int[bCOUNTER]
9:
10:    //Memory blocks management part
11:    for (i ← 0; i < bCOUNTER; i++) do
12:        order ← 0
13:        while value ≠ "" do
14:            order ← order + 1
15:            if (value.length() <= BLOCKS[i]) then
16:                if (value.length() > BLOCKS_TINY) then
17:                    break
18:                chunk ← value
19:                value ← ""
20:            else
21:                chunk ← value.substring(0, BLOCKS[i])
22:                value ← value.substring(BLOCKS[i])
23:            blocks[i].add(new Item(order, key, chunk))
24:        counters[i] ← order
25:
26:    //DB management part
27:    meta ← new Item(key, counters)
28:    metadb.insert(meta)
29:    if (using_one_sub_db) then
30:        for (i ← 0; i < bCOUNTER; i++) do
31:            for (j ← 0; j < blocks[i].size(); j++) do
32:                blocksdb.insert(blocks[i].get(j))
33:    else
34:        for (j ← 0; j < blocks[0].size(); j++) do
35:            mdb.insert(blocks[0].get(j))
36:        for (j ← 0; j < blocks[1].size(); j++) do
37:            bdb.insert(blocks[1].get(j))
38:        for (j ← 0; j < blocks[2].size(); j++) do
39:            tdb.insert(blocks[2].get(j))
40:    retrun Status.OK
```