5. String with non-contiguous duplicate, e.g.: abababa

Algorithm—With Additional Memory of Constant Size

```
1   public static void removeDuplicatesEff(char[] str) {
2       if (str == null) return;
3       int len = str.length;
4       if (len < 2) return;
5       boolean[] hit = new boolean[256];
6       for (int i = 0; i < 256; ++i) {
7           hit[i] = false;
8       }
9       hit[str[0]] = true;
10          int tail = 1;
11      for (int i = 1; i < len; ++i) {
12          if (!hit[str[i]]) {
13              str[tail] = str[i];
14              ++tail;
15              hit[str[i]] = true;
16          }
17      }
18      str[tail] = 0;
19  }
```

Test Cases:

1. String does not contain any duplicates, e.g.: abcd
2. String contains all duplicates, e.g.: aaaa
3. Null string
4. Empty string
5. String with all continuous duplicates, e.g.: aaabbb
6. String with non-contiguous duplicates, e.g.: abababa