Solutions

Marking: **M1** a correct method, **A1** the correct answer, **B1** an independent mark.

Each answer shows the steps, then the **result**.

2.1

`lo 0`, `hi 7`, `mid 3`: $21 < 40$, so `lo = 4`

`mid 5`: $33 < 40$, so `lo = 6`

`mid 6`: found 40, after 3 probes

2.2

$\lceil \log_2(10^6 + 1) \rceil$

$=$ **20**

2.3

when `hi = lo + 1`, `mid = lo`

e.g. `a = [1, 5]`, `x = 5`: `a[0] < 5` sets `lo = 0` again, for ever

fix: `lo = mid + 1`

2.4

5 linear searches: about 5×10^6 steps; sort then search: about $2 \times 10^7 + 5 \times 20$

no: sorting costs more than it saves for 5 searches

3.1

(a)

```
def lower_bound(a, x):
    lo, hi = 0, len(a)
    while lo < hi:
        mid = (lo + hi) // 2
        if a[mid] < x:
            lo = mid + 1
        else:
            hi = mid
    return lo
```

the range `[0, len(a))`; `lo = mid + 1` when `a[mid] < x`; `hi = mid` otherwise, keeping a possible answer; returns `lo`

(b) `1`; `4`

(c) $4 - 1 =$ **3**

3.2

(a) 7, as $49 \le 50 < 64$; the test $k^2 \le n$ is true for every k up to the answer and false after it, so each test can throw away half of the range

(b) $\lceil \log_2(10^6 + 1) \rceil =$ **20**

(c) about 10^6

3.3

(a) $\lceil \log_2(10^7 + 1) \rceil =$ **24**; hash table: about 1

(b) sorted array: `lower_bound("Zha")` finds the start, and the matches follow in order; a hash table scatters similar names, so it must scan all 10^7

(c) each insert shifts about $n/2 = 5 \times 10^6$ names; $1000 \times 5 \times 10^6 = 5 \times 10^9$ moves a day, a few seconds of work

(d) a sorted array with binary search: 24 probes is fast enough for 10^4 lookups a second, it gives prefix search, and new names can be added in one nightly batch merge, in $O(n)$; a balanced tree would also work if names must appear at once

exercises.lol exercises.com.cn