

ATTENTION: This problem is too difficult and open-ended to be an exam problem.

A hint given in the tutorial period was that powers of two are

1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, ...

It's also useful to know that the array element values are as follows

```
a[0] == value < 77
a[1] == value < 77
a[2] == 77
a[3] == 79
a[4], a[5], ..., a[1024] all have values > 79
```

The first call to `index_of` will have `low==0` and `high==1024`. `mid` will get a value of 512, and the test `key <= a[mid]` will be true, so a new call will be made with `low==0` and `high==512`. This pattern will repeat with `(low, high)` pairs of `(0, 256)`, `(0, 128)`, and so on until `low==0` and `high==4`. At that point `key <= a[mid]` will be false, so the outgoing `(low, high)` pair will be `(3, 4)`. From there it's a short trip to the base case of `low==3` and `high==3`, which will send -1 back as a return value. The output will be

```
low == 0; high == 1024.
low == 0; high == 512.
low == 0; high == 256.
low == 0; high == 128.
low == 0; high == 64.
low == 0; high == 32.
low == 0; high == 16.
low == 0; high == 8.
low == 0; high == 4.
low == 3; high == 4.
low == 3; high == 3.
low == 3; high == 3; result == -1.
low == 3; high == 4; result == -1.
low == 0; high == 4; result == -1.
low == 0; high == 8; result == -1.
low == 0; high == 16; result == -1.
low == 0; high == 32; result == -1.
low == 0; high == 64; result == -1.
low == 0; high == 128; result == -1.
low == 0; high == 256; result == -1.
low == 0; high == 512; result == -1.
low == 0; high == 1024; result == -1.
```

The number of `key <= a[mid]` comparisons is the number of calls to `index_of` that are not base cases, which in this case is 10.

The `key = a[mid]` comparison is done once, in the base case.