Goal

- Today we will cover a very important algorithm - Binary Search.

Resources

Binary Search

- Quick Birds Eye View
  - https://www.youtube.com/watch?v=MFhxShGxHWc
- Detailed Explanation
  - by Errichto
  - This guy is an LGM on Codeforces.
  - So, he kinda knows what he is talking about.
  - https://www.youtube.com/watch?v=GU7DpgHINWQ

Questions

Binary Search

1. Given an integers n and q.
   - Followed by a list of n integers in ascending order.
   - Now you will be given q integers.
     - For each integer output the index, if the integer exists in the list given above.
     - If the integer doesn't exist in the list, print -1.
     - You checking should be performed using binary search.
   - sample_input

```
10 5
1 2 3 4 5 6 7 8 9 10
7
3
11
1
10
```

   - sample_output

```
6
2
-1
0
9
```

- Solution

```cpp
#include <bits/stdc++.h>
using namespace std;

int binary_search(int x, vector<int> &a) {
    int l = 0, r = a.size() - 1;
    while (l <= r) {
        int m = l + (r - l) / 2;
        if (a[m] == x)
            return m;
        if (a[m] > x)
            r = m - 1;
        else
            l = m + 1;
    }
    return -1;
}

int main() {
    int n, q; cin >> n >> q;
    vector<int> a(n);
    for (int i = 0; i < n; i++)
        cin >> a[i];
    while (q--) {
        int x; cin >> x;
        cout << binary_search(x, a) << "\n";
    }
    return 0;
}
```

2. Given an integer x print its square root if it is a perfect square otherwise print -1.
    - Use binary search to find the square root.
    - sample_input_1

    ```
    100000000
    ```

    - sample_output_1

    ```
    10000
    ```

    - sample_input_2

    ```
    1000
    ```

    - sample_output_2

    ```
    -1
    ```

- Solution

```
#include <bits/stdc++.h>
using namespace std;

typedef long long ll;

int sqrt(int x) {
    ll l = 0, r = x;
    while (l <= r) {
        ll m = l + (r - l) / 2;
        if (m * m == x)
            return m;
        if (m * m > x)
            r = m - 1;
        else
            l = m + 1;
    }
    return -1;
}

int main() {
    int x; cin >> x;
    cout << sqrt(x);
    return 0;
}
```

3. Given an integer $n \geq 2$ followed by a list of n integers which are first strictly increasing and then strictly decreasing.
   - The length of the strictly increasing or the strictly decreasing part may be 0.
   - Find the index of the largest element using binary search.
   - sample_input

```
10
1 2 3 4 5 6 7 3 2 1
```

   - sample_output

```
6
```

- Solution

```
#include <bits/stdc++.h>
using namespace std;

int find_max(vector<int> &a) {
    int n = a.size();
    int l = 0, r = n - 1;
    while (l < r) {
```

```cpp
        int m = l + (r - l) / 2;
        if (a[m] > a[m + 1])
            r = m;
        else
            l = m + 1;
    }
    return l;
}

int main() {
    int n; cin >> n;
    vector<int> a(n);
    for (int i = 0; i < n; i++)
        cin >> a[i];
    cout << find_max(a);
    return 0;
}
```