Minimum element in a sorted and rotated array

> 求旋转数组中的最小元素

采用二分查找

对于mid，有四种情况：

1. mid < right && arr[mid] > arr[mid + 1] ，此时peak为arr[mid]，则最小元素为arr[mid + 1]
2. mid > left && arr[mid - 1] > arr[mid]，此时peak为arr[mid - 1]，则最小元素为arr[mid]
3. arr[right] > arr[mid] 则说明mid ~ right部分为未旋转数组的较小的那部分，所以应该将right左移
4. 以上情况都不满足，则将 left 右移

```
#include <bits/stdc++.h>

using namespace std;


typedef long long ll;

int minOfSortedRotatedArr(vector<int> &arr, int &N) {
    int left = 0, right = N - 1;
    while (left < right) {
        int mid = left + (right - left) / 2;
        if (mid < right && arr[mid + 1] < arr[mid]) return arr[mid + 1];
        if (mid > left && arr[mid] < arr[mid - 1]) return arr[mid];
        if (arr[right] > arr[mid]) right = mid - 1;
        else left = mid + 1;
    }
    if (left == right) return arr[left];
    if (left > right) return arr[0];
}


int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        int N, num;
        vector<int> arr;
        scanf("%d", &N);
```

```
        for (int i = 0; i < N; ++i) {
            scanf("%d", &num);
            arr.push_back(num);
        }
        printf("%d\n", minOfSortedRotatedArr(arr, N));
    }
    return 0;
}
```