```
void sqrtUpd(int id, int l, int r, int x) {
 int cl = l/K, cr = r/K;
 if(cl == cr) {
        for(int i=l; i<=r; i++) {
                p[id][i] += x;
        }
        return
 }
 for(int i=l, j=(cl+1)*K; i<j; i++) {
 p[id][i] += x
 }for(int i=cl+1; i<cr; i++) {
 a[id][i] += x;
 }for(int i=cr*K; i<=r; i++) {
 p[id][i] += x;
 }
}
}
 }
 void add(int x) {
 int prv = 0;
 for(int pos: where[x]) {
 update_max(prv+1, pos);
 prv = pos;
 }
 index_lim = min(index_lim, prv);
 }
 int main() {
 ios::sync_with_stdio(0);
 cin.tie(0);
 cin >> n;
 for(int i=1; i<=n; i++) {
 cin >> a[i];
 where[a[i]].pb(i);
 }index_lim = n;
```

```
// sqrtGet(int l, int r) {
int cl = l/K, cr = r/K
// ret = 0;
if(cl == cr) {
        for(int i=l; i<=r; i++) {
 ret += S[i];
 return ret
}
```

```
c) for(int x=0; x<=n-1; x++) {
add(x);
int j=1, ans=0;
while(j<=n) {
int i;
if(j > index_lim) {
i = n+1;
} else {
i = prof_max(j);
}
ans++;
if(i == j-1)
break
}
j = i;
}cout << ans << "\n"
x == n];
}// cerr << double(clock
() / CLOCKS_PER_SEC <<
endl;  return 0;
}
```