**985. Sum of Even Numbers After Queries**

We have an array A of integers, and an array queries of queries.

For the i-th query val = queries[i][0], index = queries[i][1], we add val to A[index].  Then, the answer to the i-th query is the sum of the even values of A.

(Here, the given *index = queries[i][1]* is a 0-based index, and each query permanently modifies the array *A*.)

Return the answer to all queries.  Your answerarray should have answer[i] as the answer to the i-th query.

我的解法：

|  |
| --- |
| vector<int> sumEvenAfterQueries(vector<int>& A, vector<vector<int>>& queries) {  int sum = 0;  vector<int> B(A.size(),0);  for(auto i = A.begin(); i < A.end(); i++) {  if(\*i % 2 == 0)  sum += \*i;  }  int val,index;  for(int i = 0; i < queries.size(); i++) {  index = queries[i][1];  val = queries[i][0];  if(A[index] % 2 == 0 && val % 2 == 0)  sum += val;  else if(A[index] % 2 == 0 && val % 2 != 0)  sum -= A[index];  else if(A[index] % 2 != 0 && val % 2 != 0)  sum += A[index] + val;  A[index] += val;  B[i] = sum;  }  return B;  } |
| 思路正确是但是代码不够优美；下面为两个比较好的C++代码 |
| vector<int> sumEvenAfterQueries(vector<int> &A, vector<vector<int>> &queries) {  vector<int> res;  int sum = 0;  for (auto a: A)  if (a % 2 == 0) sum += a;  for (auto q: queries) {  int pre = A[q[1]];  int cur = A[q[1]] + q[0];  A[q[1]] = cur;  sum -= (pre % 2 == 0) ? pre : 0;  sum += (cur % 2 == 0) ? cur : 0;  res.push\_back(sum);  }  return res;  } |
| vector<int> sumEvenAfterQueries(vector<int>& A, vector<vector<int>>& qs, vector<int> res = {}) {  int sum = accumulate(begin(A), end(A), 0, [](int s, int a) { return s + (a % 2 == 0 ? a : 0); });  for (auto &q : qs) {  if (A[q[1]] % 2 == 0) sum -= A[q[1]];  A[q[1]] += q[0];  if (A[q[1]] % 2 == 0) sum += A[q[1]];  res.push\_back(sum);  }  return res;  } |