Problem
Anna has a row of N blocks, each with exactly one letter from A to Z written on it. The blocks are numbered 1, 2, ..., N from left to right.

Today, she is learning about palindromes. A palindrome is a string that is the same written forwards and backwards. or example, ANNA, RACECAR, AAA and X are all palindromes, while AB, FROG and YOYO are not.

Bob wants to test how well Anna understands palindromes, and will ask her Q questions. The i-th question is: can A na use all of the blocks numbered from Li to Ri, inclusive, rearranging them if necessary, to form a palindrome? Aft r each question, Anna puts the blocks back in their original positions.

Please help Anna by finding out how many of Bob's questions she can answer "yes" to.

Input
The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with a line conta ning the two integers N and Q, the number of blocks and the number of questions, respectively. Then, another line f llows, containing a string of N uppercase characters (A to Z). Then, Q lines follow. The i-th line contains the two int gers Li to Ri, describing the i-th question.

Output
For each test case, output one line containing Case #x: y, where x is the test case number (starting from 1) and y is th number of questions Anna can answer "yes" to.

Limits
Time limit: 30 seconds per test set.
Memory limit: 1GB.
$1 \leq T \leq 100$. $1 \leq Li \leq Ri \leq N$.

Test set 1 (Visible)
$1 \leq N \leq 20$.
$1 \leq Q \leq 20$.

Test set 2 (Hidden)
$1 \leq N \leq 105$.
$1 \leq Q \leq 105$.

Sample

Input

Output

2
7 5
ABAACCA
3 6
4 4
2 5
6 7
3 7
3 5
XYZ

1 3
1 3
1 3
1 3
1 3

Case #1: 3
Case #2: 0

In Sample Case #1, N = 7 and Q = 5.
For the first question, Anna must use the blocks AACC. She can rearrange these blocks into the palindrome ACCA (r CAAC).
For the second question, Anna must use the blocks A. This is already a palindrome, so she does not need to rearrang them.
For the third question, Anna must use the blocks BAAC. These blocks cannot be rearranged into a palindrome.
For the fourth question, Anna must use the blocks CA. These blocks cannot be rearranged into a palindrome.
For the fifth question, Anna must use the blocks AACCA. She can rearrange these blocks to form the palindrome A ACA (or CAAAC).
In total, she is able to answer "yes" to 3 of Bob's questions, so the answer is 3.

In Sample Case #2, N = 3 and Q = 5. For the first question, Anna must use the blocks XYZ to create a palindrome. T is is impossible, and since the rest of Bob's questions are the same as the first one, the answer is 0.

```
// In the name of God

#include <iostream>
#include <algorithm>
#include <fstream>
#include <vector>
#include <deque>
#include <assert.h>
#include <queue>
#include <stack>
#include <set>
#include <map>
#include <stdio.h>
#include <string.h>
#include <utility>
#include <math.h>
#include <bitset>
#include <iomanip>
#include <complex>

using namespace std;

#define rep(i, a, b) for (int i = (a), i##_end_ = (b); i < i##_end_; ++i)
#define debug(...) fprintf(stderr, __VA_ARGS__)
#define mp make_pair
#define x first
```

```
#define y second
#define pb push_back
#define SZ(x) (int((x).size()))
#define ALL(x) (x).begin(), (x).end()

template<typename T> inline bool chkmin(T &a, const T &b) { return a > b ? a = b, 1 : 0; }
template<typename T> inline bool chkmax(T &a, const T &b) { return a < b ? a = b, 1 : 0; }
template<typename T> inline bool smin(T &a, const T &b)   { return a > b ? a = b : a;    }
template<typename T> inline bool smax(T &a, const T &b)   { return a < b ? a = b : a;    }

typedef long long LL;

const int N = (int) 2e5 + 5, mod = (int) 0;
int sum[N][26];
int main() {
 ios_base::sync_with_stdio(0);
 int tc;
 cin >> tc;
 for (int tt = 1; tt <= tc; ++tt) {
  cout << "Case #" << tt << ": ";
  int n, q;
  cin >> n >> q;
  string s;
  cin >> s;
  for (int j = 0; j < n; ++j) {
   for (int i = 0; i < 26; ++i) {
    sum[j + 1][i] = sum[j][i] ^ (s[j] == i + 'A');
   }
  }
  int res = 0;
  for (int j = 0; j < q; ++j) {
   int xl, xr;
   cin >> xl >> xr;
   --xl;
   int cnt = 0;
   for (int i = 0; i < 26; ++i)
    cnt += sum[xr][i] ^ sum[xl][i];
   if (cnt <= 1) ++res;
  }
  cout << res << '\n';

 }
}
```