semiexp

```
using namespace std;
#define MOD @
#define ADD(X,Y) ((X) = ((X) + (Y)%MOD) % MOD)
typedef long long i64; typedef vector<int> ivec; typedef vector<string>
svec;

int N;
bool C[101010];
char in[101010];
set<int> tree[101010];
bool valid[101010];
int profit[101010];

void rm_edge(int x, int y)
{
	tree[x].erase(y);
	tree[y].erase(x);
}

void visit(int p)
{
	if (C[p]) return;
	if (tree[p].size() == 1) {
		valid[p] = false;
```

```
            int q = *(tree[p].begin());
            rm_edge(p, q);
            visit(q);
        }
}

vector<int> graph[101010];
int best = 0;

void update_best(pair<pair<int,int>, pair<int,int>>& p, pair<int, int> v)
{
        if (v > p.first) {
            p.second = p.first;
            p.first = v;
        } else if (v > p.second) {
            p.second = v;
        }
}

pair<int, int> solve(int p, int rt)
{
        // non leaf, any
        pair<pair<int, int>, pair<int, int> > nl{ {0, -1}, {0, -1} },
any{ {0, -1}, {0, -1} };
        for (int i = 0; i < graph[p].size(); ++i) {
            int q = graph[p][i];
```

```
            if (q == rt) continue;


            auto tmp = solve(q, p);
            update_best(nl, { tmp.first, i });
            update_best(any, { tmp.second, i });
        }
        best = max(best, profit[p] + nl.first.first + nl.second.first);
        if (nl.first.second != any.first.second) best = max(best, profit[p]
+ nl.first.first + any.first.first);
        else {
            best = max(best, profit[p] + nl.first.first +
any.second.first);
            best = max(best, profit[p] + nl.second.first +
any.first.first);
        }


        if (graph[p].size() == 1) {
            return { -10, profit[p] };
        }
        return { nl.first.first + profit[p], any.first.first + profit[p] };
}


int main()
{
        scanf("%d", &N);
        for (int i = 0; i < N - 1; ++i) {
```

```
        int x, y;
        scanf("%d%d", &x, &y);
        --x; --y;
        tree[x].insert(y);
        tree[y].insert(x);
    }
    scanf("%s", in);
    for (int i = 0; i < N; ++i) C[i] = in[i] == 'W';

    bool hasW = false;
    for (int i = 0; i < N; ++i) if (C[i]) hasW = true;

    if (!hasW) {
        puts("0");
        return 0;
    }

    fill(valid, valid + N, true);
    for (int i = 0; i < N; ++i) visit(i);

    int base = -2;
    int ep = -1;
    int cnt = 0;
    for (int i = 0; i < N; ++i) if (valid[i]) {
        ++cnt;
```

```
            base += 2;
            ep = i;
            if ((tree[i].size() % 2 != 0) != C[i]) ++base;
            for (int j : tree[i]) graph[i].push_back(j);
        }

        if (cnt == 1) {
            puts("1");
            return 0;
        }

        for (int i = 0; i < N; ++i) if (valid[i]) {
            profit[i] = ((tree[i].size() % 2 != 0) != C[i]) ? 2 : 0;
        //    printf("%d %d\n", i, profit[i]);
        }

        //printf("%d\n", base);

        solve(ep, -1);

        printf("%d\n", base + min(-best, 0));
        return 0;
}
```

eddy1021

```
using namespace std;
typedef long long LL;
const LL mod7=1000000007LL;
inline LL getint(){
  LL _x=0,_tmp=1; char _tc=getchar();
  while( (_tc<'0'||_tc>'9')&&_tc!='-' ) _tc=getchar();
  if( _tc == '-' ) _tc=getchar() , _tmp = -1;
  while(_tc>='0'&&_tc<='9') _x*=10,_x+=(_tc-'0'),_tc=getchar();
  return _x*_tmp;
}
inline LL add(LL _x, LL _y, LL _mod=mod7){
  _x+=_y;
  return _x>=_mod ? _x-_mod : _x;
}
inline LL sub(LL _x, LL _y, LL _mod=mod7){
  _x-=_y;
  return _x<0 ? _x+_mod : _x;
}
inline LL mul(LL _x, LL _y ,LL _mod=mod7){
  _x*=_y;
  return _x>=_mod ? _x%_mod : _x;
}
LL mypow(LL _a, LL _x, LL _mod){
  if(_x == 0) return 1LL;
```

```
  LL _ret = mypow(mul(_a, _a, _mod), _x>>1, _mod);
  if(_x & 1) _ret=mul(_ret, _a, _mod);
  return _ret;
}
LL mymul(LL _a, LL _x, LL _mod){
  if(_x == 0) return 0LL;
  LL _ret = mymul(add(_a, _a, _mod), _x>>1, _mod);
  if(_x & 1) _ret=add(_ret, _a, _mod);
  return _ret;
}
void sleep(double sec = 1021){
  clock_t s = clock();
  while(clock() - s < CLOCKS_PER_SEC * sec);
}
#define Bye exit(0)
int __ = 1 , _cs;
/*********default*********/
const int N=101010;
void build(){

}
int n, vl[N];
vector<int> v[N];
char c[N];
void init(){
```

```
  n=getint();
  for(int i=1; i<n; i++){
    int ui=getint();
    int vi=getint();
    v[ui].push_back(vi);
    v[vi].push_back(ui);
  }
  scanf("%s", c+1);
  for(int i=1; i<=n; i++)
    v1[i]=(c[i] == 'W');
}
int root, sz[N];
void go(int now, int prt){
  sz[now]=(c[now] == 'W');
  for(int son: v[now]){
    if(son == prt) continue;
    go(son, now);
    sz[now]+=sz[son];
  }
}
int cnt[N], ans, bst, dp[N][2];
void DP(int now, int prt){
  for(int son: v[now]){
    if(son == prt or sz[son] == 0)
      continue;
```

```
    cnt[now] ++;
    DP(son, now);
    ans+=2;
  }
  for(int son: v[now]){
    if(son == prt or sz[son] == 0)
      continue;
    int tdp[2];
    tdp[0]=(1+(((vl[son]+cnt[son]+1)&1)?1:-1))+dp[son][0];
    tdp[1]=(1+(((vl[now]+cnt[now]+(now != prt))&1)?1:-1))+dp[son][1];
    bst=max(bst, tdp[0]+dp[now][1]);
    bst=max(bst, tdp[1]+dp[now][0]);
    dp[now][0]=max(dp[now][0], tdp[0]);
    dp[now][1]=max(dp[now][1], tdp[1]);
  }
  //printf("%d: %d %d\n", now, dp[now][0], dp[now][1]);
  ans+=(vl[now]+cnt[now]+(now!=prt))&1;
}
void solve(){
  for(int i=1; i<=n; i++)
    if(c[i] == 'W'){
      root=i;
      break;
    }
  if(!root){
```

```
    cout<<0<<endl;
    exit(0);
  }
  go(root, root);
  DP(root, root);
  cout<<ans-bst<<endl;
}
int main(){
  build();
  //__ = getint();
  while(__ --){
    init();
    solve();
  }
}
```

sugim48

```
using namespace std;

#define rep(i, N) for (int i = 0; i < N; i++)
#define pb push_back

typedef long long ll;
```

```
void dfs_ignore(int u, int p, vector<vector<int>>& G, vector<int>& a,
vector<bool>& ignore) {
	ignore[u] = a[u];
	for (int v: G[u]) if (v != p) {
		dfs_ignore(v, u, G, a, ignore);
		if (!ignore[v]) ignore[u] = false;
	}
}

int maji = INT_MIN / 10;

void dfs_ma(int u, int p, vector<vector<int>>& G, vector<int>& a,
vector<bool>& ignore, vector<int>& dp) {
	vector<int> unko = {0, 0};
	for (int v: G[u]) if (v != p && !ignore[v]) {
		dfs_ma(v, u, G, a, ignore, dp);
		unko.pb(dp[v]);
	}
	sort(unko.rbegin(), unko.rend());
	int x = a[u] ? 0 : 2;
	maji = max(maji, x + unko[0] + unko[1] + 1);
	dp[u] = x + unko[0];
}

int main() {
```

```
int N; cin >> N;
vector<vector<int>> G(N);
rep(i, N - 1) {
	int u, v; scanf("%d%d", &u, &v), u--, v--;
	G[u].pb(v), G[v].pb(u);
}
string s; cin >> s;
vector<int> a(N);
rep(u, N) a[u] = (s[u] == 'B');
int r = -1;
rep(u, N) if (!a[u]) r = u;
if (r == -1) {
	cout << 0 << endl;
	return 0;
}
vector<bool> ignore(N);
dfs_ignore(r, -1, G, a, ignore);
int ans = -1;
rep(u, N) if (!ignore[u]) {
	ans += 2;
	int deg = 0;
	for (int v: G[u]) if (!ignore[v]) deg++;
	a[u] ^= deg % 2;
	ans += !a[u];
}
```

```
	if (ans == 2) {
		cout << 1 << endl;
		return 0;
	}
	int ma = 1;
	vector<int> dp(N);
	dfs_ma(r, -1, G, a, ignore, dp);
	ma = max(ma, maji);
	cout << ans - ma << endl;
}
```

dreamoon

```
#define SZ(X) ((int)(X).size())
#define ALL(X) (X).begin(), (X).end()
#define REP(I, N) for (int I = 0; I < (N); ++I)
#define REPP(I, A, B) for (int I = (A); I < (B); ++I)
#define FOR(I, A, B) for (int I = (A); I <= (B); ++I)
#define FORS(I, S) for (int I = 0; S[I]; ++I)
#define RS(X) scanf("%s", (X))
#define SORT_UNIQUE(c) (sort(c.begin(),c.end()),
c.resize(distance(c.begin(),unique(c.begin(),c.end()))))
#define GET_POS(c,x) (lower_bound(c.begin(),c.end(),x)-c.begin())
#define CASET int ___T; scanf("%d", &___T); for(int cs=1;cs<=___T;cs++)
#define MP make_pair
#define PB push_back
```

```
#define MS0(X) memset((X), 0, sizeof((X)))
#define MS1(X) memset((X), -1, sizeof((X)))
#define LEN(X) strlen(X)
#define F first
#define S second
using namespace std;
typedef long long LL;
typedef unsigned long long ULL;
typedef long double LD;
typedef pair<int,int> PII;
typedef vector<int> VI;
typedef vector<LL> VL;
typedef vector<PII> VPII;
typedef pair<LL,LL> PLL;
typedef vector<PLL> VPLL;
template<class T> void _R(T &x) { cin >> x; }
void _R(int &x) { scanf("%d", &x); }
void _R(int64_t &x) { scanf("%lld", &x); }
void _R(double &x) { scanf("%lf", &x); }
void _R(char &x) { scanf(" %c", &x); }
void _R(char *x) { scanf("%s", x); }
void R() {}
template<class T, class... U> void R(T &head, U &... tail) { _R(head);
R(tail...); }
template<class T> void _W(const T &x) { cout << x; }
void _W(const int &x) { printf("%d", x); }
```

```
void _W(const int64_t &x) { printf("%lld", x); }

void _W(const double &x) { printf("%.16f", x); }

void _W(const char &x) { putchar(x); }

void _W(const char *x) { printf("%s", x); }

template<class T,class U> void _W(const pair<T,U> &x) {_W(x.F); putchar(' '); _W(x.S);}

template<class T> void _W(const vector<T> &x) { for (auto i = x.begin(); i != x.end(); _W(*i++)) if (i != x.cbegin()) putchar(' '); }

void W() {}

template<class T, class... U> void W(const T &head, const U &... tail) { _W(head); putchar(sizeof...(tail) ? ' ' : '\n'); W(tail...); }

#ifdef HOME

 #define DEBUG(...) {printf("# ");printf(__VA_ARGS__);puts("");}

#else

 #define DEBUG(...)

#endif

int MOD = 1e9+7;

void ADD(LL& x,LL v){x=(x+v)%MOD;if(x<0)x+=MOD;}

/*}}}*/

const int SIZE = 1e6+10;

VI e[SIZE];

char s[SIZE];

int have_w[SIZE];

int bw[SIZE];

int base;

void dfs0(int x,int lt){
```

```
    if(s[x]=='W')have_w[x]=1;
    for(int y: e[x]){
        if(y==lt)continue;
        dfs0(y,x);
        if(have_w[y]){
            have_w[x]=1;
            base+=2;
            bw[x]^=1;
        }
    }
    if(x!=lt&&have_w[x])bw[x]^=1;
    if(!bw[x])base++;
}
int dp[SIZE][2];
int ans;
const int INF = 1e8;
void dfs(int x,int lt){
    dp[x][1]=-INF;
    dp[x][0]=-INF;
    for(int y:e[x]){
        if(y==lt)continue;
        if(!have_w[y])continue;
        dfs(y,x);
        ans=min(ans,base-dp[y][1]);
        ans=min(ans,base-dp[y][0]+bw[x]);
```

```
            ans=min(ans,base-dp[x][0]-dp[y][1]+(2*bw[x]-1));
            ans=min(ans,base-dp[x][1]-dp[y][0]+(2*bw[x]-1));
            REP(j,2)dp[x][j]=max(dp[x][j],dp[y][j]);
        }
        REP(i,2)dp[x][i]+=1-(2*bw[x]-1);
        dp[x][1]=max(dp[x][1],1-(2*bw[x]-1));
        dp[x][0]=max(dp[x][0],1);
    }
    int main(){
        int N; R(N);
        REPP(i,1,N){
            int x,y;
            R(x,y);
            e[x].PB(y);
            e[y].PB(x);
        }
        RS(s+1);
        int w_cnt=0;
        FOR(i,1,N){
            if(s[i]=='W')w_cnt++;
        }
        if(w_cnt <= 1){
            W(w_cnt);
            return 0;
        }
```

```
    int st;
    FOR(i,1,N){
        if(s[i]=='W'){
            st=i;
        }
        else bw[i]=1;
    }
    dfs0(st,st);
    ans=INF;
    dfs(st,st);
    W(ans);
    return 0;
}
```

WA_TLE

```
using namespace std;
typedef long long int llint;
typedef long double lldo;
#define mp make_pair
#define mt make_tuple
#define pub push_back
#define puf push_front
#define pob pop_back
```

```
#define pof pop_front

#define fir first

#define sec second

#define res resize

#define ins insert

#define era erase

//cout<<setprecision(20)

//cin.tie(0);

//ios::sync_with_stdio(false);

const llint mod=1000000007;

const llint big=4.19e18+1;

const long double pai=3.14159265358979323846264338327950288419 7;

const long double eps=1e-15;

template <class T,class U>void mineq(T& a,U b){if(a>b){a=b;}}

template <class T,class U>void maxeq(T& a,U b){if(a<b){a=b;}}

llint gcd(llint a,llint b){if(a%b==0){return b;}else return gcd(b,a%b);}

llint lcm(llint a,llint b){return a/gcd(a,b)*b;}

template<class T> void SO(T& ve){sort(ve.begin(),ve.end());}

template<class T> void REV(T& ve){reverse(ve.begin(),ve.end());}

int LBI(vector<int>&ar,int in){return lower_bound(ar.begin(),ar.end(),in)-
ar.begin();}

int UBI(vector<int>&ar,int in){return upper_bound(ar.begin(),ar.end(),in)-
ar.begin();}

vector<vector<int>>go;

vector<bool>need;

string str;
```

```
int hru=0;
int solve(int ter,int per){
	//あるパスを選んで、その頂点のパリティによって減らす
	//白なら-2ということ
	if(!need[ter]){return 0;}
	int ha=0,hb=0;
	for(auto it:go[ter]){
		if(it==per){continue;}
		maxeq(hb,solve(it,ter));
		if(ha<hb){swap(ha,hb);}
	}
	if(str[ter]=='W'){ha+=2;}
	maxeq(hru,ha+hb);
	return ha;
}
int main(void){
	int n,i;cin>>n;
	go.res(n);need.res(n);
	for(i=1;i<n;i++){
		int x,y;cin>>x>>y;x--;y--;
		go[x].pub(y);
		go[y].pub(x);
	}
	for(i=0;i<n;i++){need[i]=true;}
	vector<int>ed(n);
```

```
queue<int>que;
for(i=0;i<n;i++){
	ed[i]=go[i].size();
	if(ed[i]<=1){que.push(i);}
}
//cerr<<"de"<<__LINE__<<endl;
cin>>str;
while(que.size()>0){
	int ter=que.front();que.pop();
	if(str[ter]=='W'){continue;}
	else{
		need[ter]=false;
		for(auto it:go[ter]){
			if(need[it]){
				ed[it]--;
				if(ed[it]==1){que.push(it);}
			}
		}
	}
}
int ans=0,ninu=0;
for(i=0;i<n;i++){
	if(need[i]){
		ninu++;
		ans+=2;
```

```
                int ki=0;
                for(auto it:go[i]){
                        if(need[it]){ki++;}
                }
                if(str[i]=='B'){ki++;}

                if(ki%2==0){ans++;str[i]='W';}
                else{str[i]='B';}
            }
        }
        if(ninu==1){ans+=2;}
        //cerr<<str<<endl;
        for(i=0;i<n;i++){if(need[i]){solve(i,-1);cout<<ans-
hru-2<<endl;return 0;}}
        cout<<0<<endl;
        return 0;
}
```

uwi

```
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.PrintWriter;
import java.util.Arrays;
import java.util.InputMismatchException;
```

```
public class Main {
	static InputStream is;
	static PrintWriter out;
	static String INPUT = "";

	static void solve()
	{
		int n = ni();
		int[] from = new int[n - 1];
		int[] to = new int[n - 1];
		for (int i = 0; i < n - 1; i++) {
			from[i] = ni() - 1;
			to[i] = ni() - 1;
		}
		int[][] g = packU(n, from, to);
		int[][] pars = parents3(g, 0);
		int[] par = pars[0], ord = pars[1], dep = pars[2];

		char[] s = ns().toCharArray();

		int[][] dpr = new int[2][n]; // return flip
		int[][] dpg = new int[2][n]; // go
		for(int i = n-1;i >= 0;i--){
			int cur = ord[i];
```

```
            int rsum = 0;
            int tog = 0;
            int ming = 0;
            for(int e : g[cur]){
                if(par[cur] == e)continue;
                if(dpr[0][e] > 0){
                    rsum += dpr[1][e] + 2;
                    tog++;

                    ming = Math.min(ming,
                                (dpg[1][e] + 1) - (dpr[1][e]
+ 2)
                                );
                }
            }

            int[] res = process(rsum, s[cur], tog, ming);

            dpr[0][cur] = res[0];
            dpr[1][cur] = res[1];
            dpg[0][cur] = res[2];
            dpg[1][cur] = res[3];
        }

        int ans = Integer.MAX_VALUE;
```

```
        int[][] ddpr = new int[2][n];
        int[][] ddpg = new int[2][n];

        for(int i = 0;i < n;i++){
            int cur = ord[i];
//          tr(cur);
//          tr(ddpr[0][cur], ddpr[1][cur]);
//          tr(ddpg[0][cur], ddpg[1][cur]);

            int rsum = 0;
            int tog = 0;
            int ming1 = 0;
            int ming2 = 0;
            for(int e : g[cur]){
                if(par[cur] == e)continue;
                if(dpr[0][e] > 0){
                    rsum += dpr[1][e] + 2;
                    tog++;

                    int v = (dpg[1][e] + 1) - (dpr[1][e] + 2);
                    if(v < ming1){
                        ming2 = ming1;
                        ming1 = v;
                    }else if(v < ming2){
```

```
                        ming2 = v;
                    }
                }
            }
            if(ddpr[0][cur] > 0){
                rsum += ddpr[1][cur] + 2;
                tog++;

                int v = (ddpg[1][cur] + 1) - (ddpr[1][cur] + 2);
                if(v < ming1){
                    ming2 = ming1;
                    ming1 = v;
                }else if(v < ming2){
                    ming2 = v;
                }
            }

            {
                int[] res = process(rsum, s[cur], tog, ming1);
//                  tr(cur, res);
                ans = Math.min(ans, res[2]);
            }

            for(int e : g[cur]){
                if(par[cur] == e)continue;
```

```
                        int lrsum = rsum, ltog = tog;
                        int lming = ming1;

                        if(dpr[0][e] > 0){
                            lrsum -= (dpr[1][e] + 2);
                            ltog -= 1;
                            int v = (dpg[1][e] + 1) - (dpr[1][e] + 2);
                            lming = ming1 == v ? ming2 : ming1;
                        }

//                      tr(cur, e, lrsum, ltog, lming, s[cur]);
                        int[] res = process(lrsum, s[cur], ltog, lming);

                        ddpr[0][e] = res[0];
                        ddpr[1][e] = res[1];
                        ddpg[0][e] = res[2];
                        ddpg[1][e] = res[3];
                    }
                }

//              tr(dpr);
//              tr(dpg);

                out.println(ans);
```

```
}

static int[] process(int rsum, char c, int tog, int ming)
{
	if(rsum == 0){
		if(c == 'B'){
			return new int[]{0, 1, 0, 1};
		}else{
			return new int[]{1, 0, 1, 0};
		}
	}else{
		int[] ret= new int[4];
		{
			char t = c;
			if(tog % 2 == 1){
				t = t == 'B' ? 'W' : 'B';
			}
			ret[0] = rsum + (t == 'W' ? 1 : 0);
			ret[1] = rsum + (t == 'W' ? 0 : 1);
		}
		{
			char t = c;
			if(tog % 2 == 0){
				t = t == 'B' ? 'W' : 'B';
			}
```

```
                ret[2] = rsum + ming + (t == 'W' ? 1 : 0);
                ret[3] = rsum + ming + (t == 'W' ? 0 : 1);
            }
            return ret;
        }
    }

    public static int[][] parents3(int[][] g, int root) {
        int n = g.length;
        int[] par = new int[n];
        Arrays.fill(par, -1);

        int[] depth = new int[n];
        depth[0] = 0;

        int[] q = new int[n];
        q[0] = root;
        for (int p = 0, r = 1; p < r; p++) {
            int cur = q[p];
            for (int nex : g[cur]) {
                if (par[cur] != nex) {
                    q[r++] = nex;
                    par[nex] = cur;
                    depth[nex] = depth[cur] + 1;
                }
```

```
            }
        }
        return new int[][] { par, q, depth };
    }

    static int[][] packU(int n, int[] from, int[] to) {
        int[][] g = new int[n][];
        int[] p = new int[n];
        for (int f : from)
            p[f]++;
        for (int t : to)
            p[t]++;
        for (int i = 0; i < n; i++)
            g[i] = new int[p[i]];
        for (int i = 0; i < from.length; i++) {
            g[from[i]][--p[from[i]]] = to[i];
            g[to[i]][--p[to[i]]] = from[i];
        }
        return g;
    }


    public static void main(String[] args) throws Exception
    {
        long S = System.currentTimeMillis();
        is = INPUT.isEmpty() ? System.in : new
```

```
ByteArrayInputStream(INPUT.getBytes());
		out = new PrintWriter(System.out);

		solve();
		out.flush();
		long G = System.currentTimeMillis();
		tr(G-S+"ms");
	}

	private static boolean eof()
	{
		if(lenbuf == -1)return true;
		int lptr = ptrbuf;
		while(lptr < lenbuf)if(!isSpaceChar(inbuf[lptr++]))return
false;

		try {
			is.mark(1000);
			while(true){
				int b = is.read();
				if(b == -1){
					is.reset();
					return true;
				}else if(!isSpaceChar(b)){
					is.reset();
					return false;
```

```
                        }
                    }
                } catch (IOException e) {
                    return true;
                }
            }

            private static byte[] inbuf = new byte[1024];
            static int lenbuf = 0, ptrbuf = 0;

            private static int readByte()
            {
                if(lenbuf == -1)throw new InputMismatchException();
                if(ptrbuf >= lenbuf){
                    ptrbuf = 0;
                    try { lenbuf = is.read(inbuf); } catch (IOException e)
{ throw new InputMismatchException(); }
                    if(lenbuf <= 0)return -1;
                }
                return inbuf[ptrbuf++];
            }

            private static boolean isSpaceChar(int c) { return !(c >= 33 && c <=
126); }
//          private static boolean isSpaceChar(int c) { return !(c >= 32 && c <=
126); }
            private static int skip() { int b; while((b = readByte()) != -1 &&
```

```
isSpaceChar(b)); return b; }

	private static double nd() { return Double.parseDouble(ns()); }
	private static char nc() { return (char)skip(); }

	private static String ns()
	{
		int b = skip();
		StringBuilder sb = new StringBuilder();
		while(!(isSpaceChar(b))){
			sb.appendCodePoint(b);
			b = readByte();
		}
		return sb.toString();
	}

	private static char[] ns(int n)
	{
		char[] buf = new char[n];
		int b = skip(), p = 0;
		while(p < n && !(isSpaceChar(b))){
			buf[p++] = (char)b;
			b = readByte();
		}
		return n == p ? buf : Arrays.copyOf(buf, p);
```

```
	}

	private static char[][] nm(int n, int m)
	{
		char[][] map = new char[n][];
		for(int i = 0;i < n;i++)map[i] = ns(m);
		return map;
	}

	private static int[] na(int n)
	{
		int[] a = new int[n];
		for(int i = 0;i < n;i++)a[i] = ni();
		return a;
	}

	private static int ni()
	{
		int num = 0, b;
		boolean minus = false;
		while((b = readByte()) != -1 && !((b >= '0' && b <= '9') || b
== '-'));
		if(b == '-'){
			minus = true;
			b = readByte();
		}
```

```
		while(true){
			if(b >= '0' && b <= '9'){
				num = num * 10 + (b - '0');
			}else{
				return minus ? -num : num;
			}
			b = readByte();
		}
	}

	private static long nl()
	{
		long num = 0;
		int b;
		boolean minus = false;
		while((b = readByte()) != -1 && !((b >= '0' && b <= '9') || b
== '-'));
		if(b == '-'){
			minus = true;
			b = readByte();
		}

		while(true){
			if(b >= '0' && b <= '9'){
				num = num * 10 + (b - '0');
```

```
                }else{
                        return minus ? -num : num;
                }
                b = readByte();
            }
        }

        private static void tr(Object... o) { if(INPUT.length() != 0)
System.out.println(Arrays.deepToString(o)); }
}
```

DEGwer

```
using namespace std;
vector<int>pat[101010];
bool flag[101010];
bool isv[101010];
string str;
bool dfs(int node)
{
        flag[node] = true;
        bool r = (str[node] == 'W');
        for (int i = 0; i < pat[node].size(); i++)
        {
                int v = pat[node][i];
```

```
            if (flag[v])continue;
            r |= dfs(v);
        }
        return isv[node] = r;
}
int sc[101010];
int maxi = 0;
int calc(int node)
{
        flag[node] = true;
        int m1 = 0, m2 = 0;
        for (int i = 0; i < pat[node].size(); i++)
        {
            int v = pat[node][i];
            if (!isv[v])continue;
            if (flag[v])continue;
            int t = calc(v);
            if (m1 < t)m2 = m1, m1 = t;
            else if (m2 < t)m2 = t;
        }
        //printf("%d %d %d\n", node + 1, m1, m2);
        maxi = max(maxi, m1 + m2 + ((str[node] == 'W') ? 2 : 0));
        return m1 + ((str[node] == 'W') ? 2 : 0);
}
int main()
```

```
{
	int num;
	scanf("%d", &num);
	for (int i = 0; i < num - 1; i++)
	{
		int za, zb;
		scanf("%d%d", &za, &zb);
		za--, zb--;
		pat[za].push_back(zb);
		pat[zb].push_back(za);
	}
	cin >> str;
	int rr = -1;
	for (int i = 0; i < num; i++)if (str[i] == 'W')rr = i;
	if (rr == -1)
	{
		printf("0\n");
		return 0;
	}
	dfs(rr);
	int cnt = 0;
	for (int i = 0; i < num; i++)cnt += isv[i];
	if (cnt == 1)
	{
		printf("1\n");
```

```
            return 0;
        }
        for (int i = 0; i < num; i++)
        {
            //if (isv[i])printf("%d\n", i + 1);
            for (int j = 0; j < pat[i].size(); j++)
            {
                if (isv[i] && isv[pat[i][j]])str[i] = 'W' + 'B' -
str[i];
            }
        }
        //cout << str << endl;
        fill(flag, flag + num, false);
        calc(rr);
        int ans = cnt * 2 - 2;
        for (int i = 0; i < num; i++)if (isv[i] && str[i] == 'W')ans++;
        printf("%d\n", ans - maxi);
}
```

logicmachine

```
namespace loquat {
using vertex_t = size_t;
}
namespace loquat {
```

```
namespace edge_param {
struct to_ {
	vertex_t to;
	explicit to_(vertex_t t = 0)
		: to(t)
	{ }
};
}
namespace detail {
template <typename T, typename... Params>
struct edge_param_wrapper : public T, edge_param_wrapper<Params...> {
};
template <typename T>
struct edge_param_wrapper<T> : public T {
	template <typename U>
	explicit edge_param_wrapper(U&& x)
		: T(std::forward<U>(x))
	{ }
};
}
template <typename... Params>
struct edge : public detail::edge_param_wrapper<edge_param::to_, Params...>
{
	template <typename... Args>
	explicit edge(Args&&... args)
		: detail::edge_param_wrapper<edge_param::to_, Params...>(
```

```
                    std::forward<Args>(args)...)
        { }
};
}
namespace loquat {
template <typename EdgeType>
class adjacency_list {
public:
        using edge_type = EdgeType;
        using edge_list = std::vector<edge_type>;
private:
        std::vector<std::vector<EdgeType>> m_edges;
public:
        explicit adjacency_list(size_t n)
                : m_edges(n)
        { }
        size_t size() const {
                return m_edges.size();
        }
        const edge_list& operator[](vertex_t u) const {
                return m_edges[u];
        }
        edge_list& operator[](vertex_t u){
                return m_edges[u];
        }
```

```
        template <typename... Args>
        void add_edge(vertex_t from, Args&&... args){
                m_edges[from].emplace_back(std::forward<Args>(args)...);
        }
};
}
namespace loquat {
template <typename EdgeType, typename Behavior, typename... Args>
std::vector<typename Behavior::state_type>
undirected_tree_dynamic_programming(
        const adjacency_list<EdgeType>& graph,
        Behavior behavior,
        Args&&... args)
{
        using state_type = typename Behavior::state_type;
        using frame_type = std::pair<vertex_t, size_t>;
        const size_t n = graph.size();
        std::vector<state_type> temporary(n), result(n);
        for(vertex_t v = 0; v < n; ++v){
                temporary[v] = behavior.initial(v, args...);
        }
        std::stack<frame_type> frames;
        frames.emplace(0, 0);
        while(!frames.empty()){
                const auto frame = frames.top();
```

```
            frames.pop();
            const auto u = frame.first;
            const auto i = frame.second;
            const vertex_t p = (frames.empty() ? u : frames.top().first);
            if(i == graph[u].size()){
                  for(const auto& e : graph[u]){
                        if(e.to == p){ continue; }
                        const auto& x = temporary[e.to];
                        const auto& y = temporary[u];
                        temporary[u] = behavior.merge(y, x, u, e,
args...);
                  }
            }else{
                  const auto& e = graph[u][i];
                  frames.emplace(u, i + 1);
                  if(e.to != p){ frames.emplace(e.to, 0); }
            }
      }
      frames.emplace(0, 0);
      result[0] = temporary[0];
      while(!frames.empty()){
            const auto frame = frames.top();
            frames.pop();
            const auto u = frame.first;
            const auto i = frame.second;
            const vertex_t p = (frames.empty() ? u : frames.top().first);
```

```
            if(i == 0){
                for(const auto& e : graph[u]){
                    if(e.to == p){
                        const auto& x = temporary[e.to];
                        const auto& y = temporary[u];
                        result[u] = behavior.merge(y, x, u, e,
args...);
                        break;
                    }
                }
            }
            if(i != graph[u].size()){
                const auto& e = graph[u][i];
                frames.emplace(u, i + 1);
                if(e.to != p){
                    const auto& x = temporary[e.to];
                    const auto& y = result[u];
                    temporary[u] = behavior.purge(y, x, u, e,
args...);
                    frames.emplace(e.to, 0);
                }
            }
        }
        return result;
}
}
```

```
namespace loquat {
template <typename T, size_t K, typename Comparator = std::less<T>>
class top_k {
public:
	using value_type = T;
private:
	Comparator m_comparator;
	size_t m_size;
	std::array<T, K> m_data;
public:
	top_k()
		: m_comparator()
		, m_size(0)
		, m_data()
	{ }
	bool empty() const {
		return m_size == 0;
	}
	size_t size() const {
		return m_size;
	}
	const value_type& operator[](size_t i) const {
		return m_data[i];
	}
	void push(const value_type& x){
		if(m_size == K && !m_comparator(x, m_data[m_size - 1]))
```

```
{ return; }
            if(m_size == K){
                  m_data[m_size - 1] = x;
            }else{
                  m_data[m_size++] = x;
            }
            for(size_t i = m_size - 1; i > 0; --i){
                  if(m_comparator(m_data[i], m_data[i - 1])){
                        std::swap(m_data[i], m_data[i - 1]);
                  }
            }
      }
};
}
using namespace std;
using edge = loquat::edge<>;
struct behavior {
      using state_type = loquat::top_k<int, 2, std::greater<int>>;
      using edge_type = edge;
      state_type initial(loquat::vertex_t v, const vector<int>& c) const {
            state_type s;
            s.push(c[v]);
            return s;
      }
      state_type merge(
```

```
        state_type y,
        const state_type& x,
        loquat::vertex_t u,
        const edge_type& e,
        const vector<int>& c) const
    {
        if(!x.empty()){ y.push(x[0] + c[u]); }
        return y;
    }
    state_type purge(
        const state_type& y,
        const state_type& x,
        loquat::vertex_t u,
        const edge_type& e,
        const vector<int>& c) const
    {
        if(!x.empty() && !y.empty() && y[0] == x[0] + c[u]){
            state_type z;
            if(y.size() > 1){ z.push(y[1]); }
            return z;
        }else{
            return y;
        }
    }
};
```

```
int main(){
	ios_base::sync_with_stdio(false);
	int n;
	cin >> n;
	loquat::adjacency_list<edge> g(n);
	for(int i = 1; i < n; ++i){
		int a, b;
		cin >> a >> b;
		--a; --b;
		g.add_edge(a, b);
		g.add_edge(b, a);
	}
	string s;
	cin >> s;
	vector<int> deg(n);
	for(int i = 0; i < n; ++i){ deg[i] = g[i].size(); }
	for(int i = 0; i < n; ++i){
		int u = i;
		while(deg[u] == 1 && s[u] == 'B'){
			int v = -1;
			for(const auto& e : g[u]){
				v = e.to;
				if(deg[v] > 0){ break; }
			}
			--deg[u];
```

```
                --deg[v];
                u = v;
            }
        }
        vector<int> c(n);
        for(int i = 0; i < n; ++i){
            const int x = (s[i] == 'W' ? 0 : 1);
            c[i] = 1 - (x ^ (deg[i] & 1));
        }
        const auto dp = loquat::undirected_tree_dynamic_programming(g,
behavior(), c);
        int shortcut = 0;
        for(const auto& p : dp){
            if(!p.empty()){ shortcut = max(shortcut, p[0]); }
        }
        const int d = shortcut * 2 - accumulate(c.begin(), c.end(), 0);
        int k = 0;
        for(int i = 0; i < n; ++i){
            if(deg[i] > 0){ ++k; }
        }
        if(k == 0){
            cout << (find(s.begin(), s.end(), 'W') != s.end() ? 1 : 0)
<< endl;
        }else{
            const int answer = (k - 1) * 2 - d;
            cout << answer << endl;
```

```
	}
	return 0;
}
```

kmjp

```
using namespace std;
typedef signed long long ll;

#undef _P
#define _P(...) (void)printf(__VA_ARGS__)
#define FOR(x,to) for(x=0;x<(to);x++)
#define FORR(x,arr) for(auto& x:arr)
#define ITR(x,c) for(__typeof(c.begin()) x=c.begin();x!=c.end();x++)
#define ALL(a) (a.begin()),(a.end())
#define ZERO(a) memset(a,0,sizeof(a))
#define MINUS(a) memset(a,0xff,sizeof(a))
//------------------------------------------------------

int N;
set<int> E[101010];
string C;
int dp[101010];
int ma;
```

```
int dfs(int cur,int pre) {
	vector<int> V(2,0);
	FORR(e,E[cur]) if(e!=pre) V.push_back(dfs(e,cur));
	sort(ALL(V));
	reverse(ALL(V));

	ma=max(ma,C[cur]+V[0]+V[1]);
	return C[cur]+V[0];
}

void solve() {
	int i,j,k,l,r,x,y; string s;

	cin>>N;
	FOR(i,N-1) {
		cin>>x>>y;
		E[x-1].insert(y-1);
		E[y-1].insert(x-1);
	}
	int ret=0;
	cin>>C;
	FOR(i,N) {
		C[i]=C[i]=='W';
		ret+=C[i];
```

```
}
if(ret==0) return _P("0\n");
if(ret==1) return _P("1\n");

queue<int> Q;
FOR(i,N) if(E[i].size()==1) Q.push(i);

while(Q.size()) {
	x=Q.front();
	Q.pop();
	if(C[x]==0) {
		C[x]=2;
		y=*E[x].begin();
		E[y].erase(x);
		if(E[y].size()==1) Q.push(y);
	}
}
ret=-2;
int root;
FOR(i,N) if(C[i]!=2) {
	C[i]^=(E[i].size()%2);
	//cout<<i<<" "<<(int)C[i]<<endl;
	ret+=2+C[i];
	root=i;
}
```

```
	dfs(root,-1);
	//cout<<ret<<" "<<ma<<endl;
	cout<<ret-2*ma<<endl;

}


int main(int argc,char** argv){
	string s;int i;
	if(argc==1) ios::sync_with_stdio(false), cin.tie(0);
	FOR(i,argc-1) s+=argv[i+1],s+='\n'; FOR(i,s.size())
ungetc(s[s.size()-1-i],stdin);
	cout.tie(0); solve(); return 0;
}
```

pekempey

```
using namespace std;

const int N = 1e5;
vector<int> g[N];
bool del[N];
int cost[N];
int dist[N];
```

```
void dfs(int u, int p) {
  for (int v : g[u]) if (v != p && !del[v]) {
    dist[v] = dist[u] + 1 + cost[u];
    dfs(v, u);
  }
}

int main() {
  int n;
  cin >> n;
  vector<int> d(n);
  for (int i = 0; i < n - 1; i++) {
    int u, v;
    cin >> u >> v;
    u--;
    v--;
    g[u].push_back(v);
    g[v].push_back(u);
    d[u]++;
    d[v]++;
  }
  string s;
  cin >> s;

  if (n == 1) {
```

```
  cout << (s[0] == 'W') << endl;
  return 0;
}

queue<int> q;
for (int i = 0; i < n; i++) {
  if (d[i] == 1 && s[i] == 'B') {
    q.push(i);
  }
}
int ans = (n - 1) * 2;
while (!q.empty()) {
  int u = q.front(); q.pop();
  del[u] = true;
  dist[u] = -1e9;
  ans -= 2;
  for (int v : g[u]) {
    d[v]--;
    if (d[v] == 1 && s[v] == 'B') {
      q.push(v);
    }
  }
}
for (int i = 0; i < n; i++) {
  cost[i] = ((s[i] == 'W') + d[i]) % 2 == 0 ? -1 : 1;
```

```
}
int k = -1;
for (int i = 0; i < n; i++) {
  if (!del[i]) {
    k = i;
    break;
  }
}
if (ans == 0) {
  cout << 1 << endl;
  return 0;
}
if (k == -1) {
  cout << 0 << endl;
  return 0;
}
for (int i = 0; i < n; i++) {
  if (!del[i]) {
    ans += ((s[i] == 'W') + d[i]) % 2;
  }
}
dfs(k, -1);
int u = -1;
int mx = -1;
for (int i = 0; i < n; i++) {
```

```
    if (!del[i] && d[i] == 1 && mx < dist[i]) {
      mx = dist[i];
      u = i;
    }
  }
  dist[u] = 0;
  dfs(u, -1);
  mx = -1;
  for (int i = 0; i < n; i++) {
    if (!del[i] && d[i] == 1 && mx < dist[i]) {
      mx = dist[i];
    }
  }
  ans -= mx;
  cout << ans << endl;
}
```

kmcode

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

int k;
int n;
```

```
#define MAX 100002

vector<int> v[MAX];

char buf[MAX];
string s;

bool ok[MAX];

bool flag[MAX];

inline bool dfs(int b,int pr=-1){
    if(pr!=-1)flag[b]^=true;
    if(s[b]=='W'){
        ok[b]=true;
        //cout<<"ok"<<endl;
    }
    for(int go:v[b]){
        if(go==pr)continue;
        ok[b]|=dfs(go,b);
    }
    for(int go:v[b]){
        if(go==pr)continue;
        if(ok[go]){
```

```
            flag[b]^=true;
        }
    }
    return ok[b];
}
int dist[MAX];
int ds[MAX];

pair<int,int> bfs(int b){
    memset(dist,-1,sizeof(dist));
    dist[b]=0;
    ds[b]=0;
    queue<int> q;
    q.push(b);
    pair<pair<int,int> ,int> p=make_pair(make_pair(-1,-1),-1);
    while(!q.empty()){
        b=q.front();
        q.pop();
        p=max(p,make_pair(make_pair(dist[b],ds[b]),b));
       // cout<<"comp "<<dist[b]<<" "<<b<<endl;
        for(int go:v[b]){
            if(ok[go]==false)continue;
            if(dist[go]>=0)continue;
            dist[go]=dist[b]+(int)(flag[go]==false);
            ds[go]=ds[b]+1;
```

```
            // cout<<"go "<<go<<" "<<dist[go]<<endl;
             q.push(go);
         }
     }
    // cout<<"res "<<p.first<<" "<<p.second<<endl;
     return make_pair(p.first.first,p.second);
}
bool cmp(int a){
     return a==1;
}
bool cmpp(int a){
     if(a==0&&ok[a]==true){
     //    cerr<<"!! "<<a<<endl;
     }
     return flag[a]==0&&ok[a]==true;
}
int main() {
       cin >> n;
       for (int i = 1; i < n; i++) {
             int a, b;
             scanf("%d%d",&a,&b);
         a--;
         b--;
         v[a].push_back(b);
         v[b].push_back(a);
```

```
    }
scanf("%s",buf);
s=buf;
int star=-1;
int cn=0;
for(int i=0;i<s.size();i++){
    if(s[i]=='W'){
        star=i;
        cn++;
    }
    if(s[i]=='B'){
        flag[i]=true;
    }
    else{
        flag[i]=false;
    }
}
if(star==-1){
    puts("0");
    return 0;
}
if(cn==1){
    puts("1");
    return 0;
}
```

```
    dfs(star);
    int ans=bfs(bfs(star).second).first;
    int sum=count_if(ok,ok+n,cmp)-1;
    sum*=2;
    int S=0;
    for(int i=0;i<n;i++){
        if(cmpp(i)){
            S++;
        }
    }
    sum+=S;
    sum-=ans*2;
    cerr<<(count_if(ok,ok+n,cmp)-1)*2<<" "<<S<<" "<<ans<<endl;
    printf("%d\n",sum);
      return 0;
}
```