```cpp
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<int, int> P;
const int MAX_N = (1 << 21) - 1;
const int inf = numeric_limits<int>::max();


ll seg_add[MAX_N];
ll seg_max[MAX_N];

// call init(0,0,n) to init
// node k => [l,r)
void init(int k,int l,int r){
  if(r - l == 1){
    seg_add[k] = 0; // init leaf
    seg_max[k] = 0;
  }else{
    int chl = k*2+1;
    int chr = k*2+2;
    seg_add[k] = 0; // init internal-node
    seg_max[k] = 0;
    init(chl, l, (l+r)/2);
    init(chr, (l+r)/2, r);
  }
}

// call add(a,b,x,0,0,n) to add
// node k => [l,r)
void add(int a,int b,int x,int k,int l, int r){
  // not cross
  if(r <= a || b <= l)return;
  else if(a <= l && r <= b){
    // [a,b) contain [l,r)
    seg_add[k] += x;
  } else{
    // otherwise
    int chl = k*2+1;
    int chr = k*2+2;
    add(a,b,x,chl,l,(l+r)/2);
    add(a,b,x,chr,(l+r)/2,r);
    seg_max[k] = max(seg_max[chl] + seg_add[chl],
                     seg_max[chr] + seg_add[chr]);
  }
}

// call find(a,b,0,0,n) to find max-value in  [a,b)
// node k => [l,r)
ll find(int a,int b,int k,int l,int r){
  if(r <= a || b <= l) return 0;
  else if(a <= l && r <= b){
    // [a,b) contain [l,r)
    return seg_max[k] + seg_add[k];
  } else{
    // otherwise
    int chl = k*2+1;
    int chr = k*2+2;
    int vl = find(a,b,chl,l,(l+r)/2);
    int vr = find(a,b,chr,(l+r)/2,r);
    return max(vl,vr) + seg_add[k];
  }
}

int main(void){
  int n;
  vector< P > d;
  int upbound = 100010;
  cin >> n;
  for(int i = 0;i < n;++i){
    int s,t;
```

```
    cin >> s >> t;
    d.push_back(P(s,t));
  }
  for(int i = 1;i < n;++i){
    add(d[i].first,d[i].second,1,0,0,upbound);
  }
  ll res = find(0,upbound,0,0,upbound);
  for(int i = 1;i < n;++i){
    add(d[i-1].first,d[i-1].second,1,0,0,upbound);
    add(d[i].first,d[i].second,-1,0,0,upbound);
    res = min(res,find(0,upbound,0,0,upbound));
  }
  cout << res << endl;
  return 0;
}
```