Disjoint Segment

- Given a set of segments $X = \{(a1, b1), \ldots, (an, bn)\}$ in which ***ai < bi*** are coordinates of the segment *i* on a line, $i = 1, \ldots, n$.
- Find a subset of *X* having the **largest cardinality** in which **no two segments** of the subset **intersect**
- **Input**
  - Line 1: Contains a positive integer n (***1 <= n <= 100000***)
  - Line i+1 (i=1,..,n): contains a_i and b_i (***1 <= a_i <= b_i <= 1000000***)
- **Output**
  - Number of segments in the solution found

Disjoint Segment

- Example

stdin	stdout
6 0 10 3 7 6 14 9 11 12 15 17 19	4

Disjoint Segment- Hint

- Áp dụng tham lam để chọn được số lượng đoạn lớn nhất
- Nếu có 2 đoạn trùng nhau, ta sẽ ưu tiên chọn khoảng có điểm cuối nhỏ hơn
- Sắp xếp các đoạn tăng dần theo điểm kết thúc
- Biến phụ trợ:

  ***last**:* lưu điểm kết thúc của đoạn trước đó ta đã chọn
- Duyệt qua tất cả các đoạn, kiểm tra xem điểm bắt đầu của điểm đó có lớn hơn điểm kết thúc của điểm cuối cùng ta đã chọn hay không
- Nếu đoạn đang xét thỏa mãn, tăng số lượng đoạn được chọn lên, và cập nhật last

Implementation

```
#include <bits/stdc++.h>
#define maxn 100005
using namespace std;
int n;
pair<int,int> a[maxn];

// sort the segments by second element of pairs
bool cmp (pair<int,int> a, pair<int,int> b) {
    return a.second < b.second;
}

void input() {
    cin >> n;
    for (int i=1; i<=n; i++) {
        cin >> a[i].first >> a[i].second;
    }
}
```

Implementation

```
void solve() {
    int res = 0;         // result
    int last = -1;       // the end point of last chosen segment
    sort(a+1, a+n+1, cmp);
    for (int i=1; i<=n; i++)
    if (a[i].first > last) {   // not overlap
        res ++;
        last = a[i].second;
    }
    cout << res << endl;
}

int main() {
    input();
    solve();
    return 0;
}
```