t2.n^2logn做法，枚举5和1两个点，4这个点可以在枚举1的时候存下来，然后2也可以枚举的时候算出来，然后logn算3.  
t3.我们把一个星星y最大的那个点编号为1, 然后顺时针往下走  
所以一个星星的2, 5点y相等, 5的y最大  
这个思路是枚举5点  
对于每个5点, 我们向前找到可用的所有1, 2点, 并且计算3, 4点的组合，对于每个1点, 它可用的4点组合都是x介于1和5两个点之间的点，然后, 2点可以是5点左边任何一个y与5点相同的点，然后, 可以发现, 对于一个给定的1点, 只要做一遍前缀和就可以算出它和前面每个2点, 以及夹在它们之间的3点的所有组合的数量，然后这样可以发现, 这些可能的1点所对应的可能的4点的组合能加, 对应的可能的2, 3点的组合也能加，所以就扫一遍, 前缀和  
然后, 我们是从左往右枚举5点的, 所以5点可以看作定点，容易发现的是, 1点和2点在这个过程中是特殊的，然后3, 4点相对来说只是个数字  
把每个点转换为一种信号, 每个信号有两个属性: 这个点到下一个点之间有多少个3, 4点; 这个点到底是1点还是2点。然后最后做前缀和的时候要把前面的2点算上

1. #include <cstdio>
2. #include <algorithm>
3. #ifdef DEBUG
4. #define echo(...) fprintf(stderr, \_\_VA\_ARGS\_\_)
5. #else
6. void Unused\_Func(...) {}
7. #define echo(...) Unused\_Func(\_\_VA\_ARGS\_\_)
8. #endif
9. void solve();
10. int main()
11. {
12. int T;
13. scanf("%d", &T);
14. while (T--)
15. solve();
16. return 0;
17. }
18. typedef long long i64;
19. const int MaxN = 5e3 + 7;
20. const int Mod = 1e9 + 7;
21. int n;
22. struct Point
23. {
24. int x, y;
25. friend bool operator<(const Point &a, const Point &b)
26. { return a.x < b.x; }
27. };
28. Point pts[MaxN];
29. enum class SigTy { Eq, Gt, };
30. SigTy sig[MaxN];
31. int post[MaxN];
32. // Sum to last section
33. i64 sumcis[MaxN];
34. // All sum to section
35. i64 sumbeg[MaxN];
36. i64 add(i64 a, i64 b)
37. {
38. return (a + b) % Mod;
39. }
40. void solve()
41. {
42. scanf("%d", &n);
43. for (int i = 1; i <= n; ++i)
44. scanf("%d %d", &pts[i].x, &pts[i].y);
45. std::sort(pts + 1, pts + n + 1);
46. i64 res = 0;
47. for (int now = 1; now <= n; ++now)
48. {
49. int lim = 1;
50. post[1] = 0;
51. // Count for every section
52. for (Point \*it = pts + now - 1; it != pts; --it)
53. if (it->y < pts[now].y)
54. ++post[lim];
55. else
56. {
57. ++lim;
58. post[lim] = 0;
59. sig[lim] = it->y == pts[now].y ? SigTy::Eq : SigTy::Gt;
60. }
61. // Add sections
62. for (int i = 2; i <= lim; ++i)
63. sumcis[i] = add(post[i - 1], sumcis[i - 1]);
64. int fac = 0;
65. sumbeg[lim + 1] = 0;
66. for (int i = lim; i > 0; --i)
67. {
68. sumbeg[i] = add(sumbeg[i + 1], fac \* post[i] % Mod);
69. if (sig[i] == SigTy::Eq)
70. ++fac;
71. }
72. // Add final result
73. for (int i = 1; i <= lim; ++i)
74. if (sig[i] == SigTy::Gt)
75. res = add(res, sumcis[i] \* sumbeg[i] % Mod);
76. }
77. printf("%lld\n", res);
78. }