1869B - 2D Путешествие

Author: **programpiggy**

Hint

What will happen if there are no major cities?

Solution

1869B - 2D Traveling

First of all, it's easy to see that if there are no major cities, the minimum value of the total cost should be $|x_a - x_b| + |y_a - y_b|$ — the optimal choice is to fly directly from city a to city b.

Claim. Piggy will pass through a maximum of 2 major cities.

Proof. If he passes through 3 or more major cities in a row, then he can fly directly from the first one to the last one. If he passes through 2 major cities and passes an ordinary city between them, the cost must be higher than flying directly between these two major cities. So the optimal choice always consists of no more than 2 major cities, and they are in a row.

Thus, you can express the optimal choice as $a(\to s)(\to t) \to b$, where s and t are both major cities. If you naively enumerate s and t, the total complexity of the solution will be $O(k^2)$. But after seeing that s and t work independently, we can enumerate them separately. The total complexity decreases to $O(n + k)$.

Implementation

```
#include <bits/stdc++.h>
#define all(s) s.begin(), s.end()
using namespace std;
using ll = long long;
using ull = unsigned long long;

const int _N = 1e5 + 5;
```

```cpp
int T;

void solve() {
	int n, k, s, t; cin >> n >> k >> s >> t;
	vector<int> x(n + 1), y(n + 1);
	for (int i = 1; i <= n; i++) cin >> x[i] >> y[i];
	ll ans = llabs(x[s] - x[t]) + llabs(y[s] - y[t]);
	ll mins = LLONG_MAX / 2, mint = LLONG_MAX / 2;
	for (int i = 1; i <= k; i++) {
		mins = min(mins, llabs(x[s] - x[i]) + llabs(y[s] - y[i]));
		mint = min(mint, llabs(x[t] - x[i]) + llabs(y[t] - y[i]));
	}
	ans = min(ans, mins + mint);
	cout << ans << '\n';
	return;
}

int main() {
	ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
	cin >> T;
	while (T--) {
		solve();
	}
}
```