Data Structures:

Planar: 包含 `MPS`, `chords`, `vertices` 和 `memo`，

Vertices: `int`，儲存vertices的數量

Chords: `map<int, int>` 儲存所有的chord，因為是以當前的j來尋找chord，所以以 chord 的 `end` 作為 key，`start` 作為 value，在查詢上會比較方便

Memo: `vector< vector< map<int, int> > >` 用 N * N 的 `vector`，每個element中都是 `map<int, int>`，也就是 `memo[i][j]` 中儲存其中的 MPS chords，如上一樣是以 chord 的 end 作為 key，start 作為 value。用 map 也可以用 `map::size` 來比較 element 間的大小。相較之下用 array 在取值和比較大小的部分可能就沒有這麼方便

MPS: `map<int, int> Planar::MPS(int i, int j)` 做法和 HW2 一樣是 Bottom-up，用2個 Loop 從小的 Subproblem 跑到大的 Subproblem，共跑C N取2個 Subproblem