PA2 Report

電機三 梁正 B08901169

Maximum Planar Subset (Top-down Algorithm)

Top-down method will reduce the time of 100000 input case from 10+ min to about 3 min

```
//top-down
int findSubset(int** MPS, int i, int j){
    if (i >= j){
        MPS[i][j] = 0;
        return MPS[i][j];
    }
    if (MPS[i][j] > 0){
        return MPS[i][j];
    }

    int k = Chords[j];
    if (k < i || k > j){
        MPS[i][j] = findSubset(MPS, i, j-1);
    }
    else if (k > i && k < j){
        MPS[i][j] = findSubset(MPS, i, j-1);
        int temp = 1 + findSubset(MPS, i, k-1) + findSubset(MPS, k+1, j-1);
        if (MPS[i][j] < temp){
            MPS[i][j] = temp;
        }
    }
    else if (k == i){
        if (j == i + 1){
            MPS[i][j] = 1;
        }else{
            MPS[i][j] = 1 + findSubset(MPS, i+1, j-1);
        }
    }
    return MPS[i][j];
}
```

I use a vector to store the input of chords, and a 2-D array to store maximum planar subset.

```
vector<int> Chords(180010);
```

```
//initialize M table
int** MPS = new int*[_2N];
for (int i=0; i<_2N; i++){
    MPS[i] = new int[_2N]();
    // for (int j=0; j<_2N; j++){
    //  MPS[i][j] = 0;
    // }
}
```

Using **`new int[size]()`** will initialize the array without looping it (save a lot of time!)