Leetcode May Challenge DAY: 25

1. Python

```
class Solution:
    def maxUncrossedLines(self, A: List[int], B: List[int]) -> int:
        dp = [[0] * (len(B) + 1) for i in range(len(A) + 1)]
        for i in range(1, len(A) + 1):
            for j in range(1, len(B) + 1):
                if(A[i - 1] == B[j - 1]):
                    dp[i][j] = 1 + dp[i - 1][j - 1]
                else:
                    dp[i][j] = max(dp[i - 1][j], dp[i][j - 1])
        return dp[len(A)][len(B)]
```

2. C++

```cpp
class Solution {
public:
    int maxUncrossedLines(vector<int>& A, vector<int>& B) {
        vector<vector<int>> dp(A.size() + 1, vector<int>(B.size() + 1));
        for(int i = 1; i <= A.size(); i++){
            for(int j = 1; j <= B.size(); j++){
                if(A[i-1] == B[j - 1])
                    dp[i][j] = 1 + dp[i - 1][j - 1];
                else
                    dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]);
            }
        }
        return dp[A.size()][B.size()];
    }
};
```

3. JAVA

```
class Solution {
   public int maxUncrossedLines(int[] A, int[] B) {
      int[][] dp = new int[A.length + 1][B.length + 1];
      for(int i = 1; i <= A.length; i++){
         for(int j = 1; j <= B.length; j++){
            if(A[i-1] == B[j - 1])
               dp[i][j] = 1 + dp[i - 1][j - 1];
            else
               dp[i][j] = Math.max(dp[i - 1][j], dp[i][j - 1]);
         }
      }
      return dp[A.length][B.length];
   }
}
```