**DP Questions**

1. **MCP**

|  |
| --- |
| **import** **java.util.\***;    **class** **MinimumCostPath** {    **private** **static** **int** **min**(**int** x, **int** y, **int** z)  {  **if** (x < y)  **return** (x < z) ? x : z;  **else**  **return** (y < z) ? y : z;  }    **private** **static** **int** **minCost**(**int** cost[][], **int** m, **int** n)  {  **int** i, j;  **int** tc[][] = **new** **int**[m + **1**][n + **1**];    tc[**0**][**0**] = cost[**0**][**0**];    **for** (i = **1**; i <= m; i++)  tc[i][**0**] = tc[i - **1**][**0**] + cost[i][**0**];    **for** (j = **1**; j <= n; j++)  tc[**0**][j] = tc[**0**][j - **1**] + cost[**0**][j];  **for** (i = **1**; i <= m; i++)  **for** (j = **1**; j <= n; j++)  tc[i][j] = min(tc[i - **1**][j - **1**],  tc[i - **1**][j], tc[i][j - **1**])  + cost[i][j];    **return** tc[m][n];  }    **public** **static** **void** **main**(String args[])  {  **int** cost[][]  = { { **1**, **2**, **3** }, { **4**, **8**, **2** }, { **1**, **5**, **3** } };  System.out.println(minCost(cost, **2**, **2**));  }  } |

1. **Edit Distance**

|  |
| --- |
| **class** **Solution**  {  **public** **int** **editDistance**(String s, String t) {    **int** m = s.length();  **int** n = t.length();    **int** dp [][] = **new** **int**[m+**1**][n+**1**];    **for**(**int** i=**0**;i<=m;i++){  dp[i][**0**] = i;  }    **for**(**int** j=**0**;j<=n;j++){  dp[**0**][j] = j;  }    **for**(**int** i=**1**;i<=m;i++){  **for**(**int** j=**1**;j<=n;j++){  **if**(s.charAt(i-**1**) == t.charAt(j-**1**)){  dp[i][j] = dp[i-**1**][j-**1**];  }    **else**{  dp[i][j] = **1**+Math.min(dp[i-**1**][j],Math.min(dp[i][j-**1**],dp[i-**1**][j-**1**]));  }  }  }    **return** dp[m][n];  }  } |