11312 UVA (3)

11392 UVA (4)

http://codeforces.com/contest/653/problem/E (6)

http://codeforces.com/contest/769/problem/C 5 //FL:ODD/**** | bfs+greed NICE

10968 UVA (3) //EASY + NICE (bfs withot <=2 nodes)

http://codeforces.com/contest/796/problem/D (3) //NICE+EASY ... print visited in bfs (not par)

10888 UVA (4) //VERY NICE — but not main technique ... ++ DP /or/ MCMF

http://codeforces.com/contest/821/problem/D (5) //VERY NICE — Consider only points not GRID

http://www.spoj.com/problems/DIGOKEYS/ (4) //Easy [Nice problem — weird statement]

http://www.spoj.com/problems/SPIKES/ (3) //Easy bfs (# of 's' * 2)

http://www.spoj.com/problems/MULTII/ (4) //VERY NICE: BFS over numbers (K*10+d)%N

http://www.spoj.com/problems/ADV04F1/ (5) //VERY NICE: [imple] ~ N^4*BigConstant

http://www.spoj.com/problems/INVESORT/ (5) //Big limit (really usefull :P)

BFS-Grid

10977 UVA (3)

928 UVA (3)

13116 UVA (4)

314 UVA (3)

11487 UVA (4)

5622 LA (7)

11931 UVA (5)

http://www.spoj.com/problems/KNMOVE/ 3 //simple knights

http://www.spoj.com/problems/SERGRID/ 3 //almost classical

http://www.spoj.com/problems/NAKANJ/ 3 //Classical chess — KNIGHT

http://www.spoj.com/problems/PUCMM223/ (4) //NICE (but not many languages) — 2 moving [x][y]

http://www.spoj.com/problems/SPIRALGR/ (4) //NICE (not typical) [SIEVE]

http://www.spoj.com/problems/DCEPC706/ (4) //NICE — travelling outside

http://codeforces.com/contest/35/problem/C (3) //No obstacles [multiple starts]

Bellman-Ford

http://www.spoj.com/problems/ARBITRAG/ (4) //Or Floyd-Warshall

Blossom

11439 (UVA)

Bridges

http://codeforces.com/contest/732/problem/F 7

http://codeforces.com/contest/700/problem/C 7

http://www.spoj.com/problems/EC_P/ (3) //bridges ONLY

http://www.spoj.com/problems/SUBMERGE/ (3) //Direct articulation

http://www.spoj.com/problems/GRAFFDEF/ (5) //Bridge tree

Centroid/Tree center

http://codeforces.com/contest/715/problem/C 9

http://codeforces.com/contest/741/problem/D 8

13164 UVA (7)

http://codeforces.com/contest/752/problem/F 5

http://codeforces.com/contest/766/problem/E 6

http://codeforces.com/contest/833/problem/D 7 //Very nice — hard (thinking + imple) + FW

http://www.spoj.com/problems/HOLI/ (4) //VERY NICE: 2*Distances from centroids

DFS

12186 UVA (3)

http://codeforces.com/contest/734/problem/E (5)

http://codeforces.com/contest/727/problem/A (3)

http://codeforces.com/contest/723/problem/E (6)

http://codeforces.com/contest/709/problem/E (6)

http://codeforces.com/contest/710/problem/E (4)

http://codeforces.com/contest/758/problem/E (8)

11323 UVA (5)

http://codeforces.com/contest/760/problem/B (3)

http://codeforces.com/contest/761/problem/E (6)

http://codeforces.com/contest/638/problem/B (3) //connect cons. letters

http://codeforces.com/contest/638/problem/C (4) //greedy idea — easy

http://codeforces.com/contest/638/problem/D (5) //spec-DAG articulatin

http://codeforces.com/contest/767/problem/C (4)

http://codeforces.com/contest/781/problem/C (5)

http://codeforces.com/contest/794/problem/D (5) //NICE! Right done dfs

http://codeforces.com/contest/802/problem/K (5) //Slightly DP-like (NICE) TREE

http://codeforces.com/contest/813/problem/C (3) //Simply 2 DFS: NICE + EASY

http://codeforces.com/contest/841/problem/D (4) //DFS while tracking "next"

http://codeforces.com/contest/845/problem/G (5) //Keep track of cycles

http://codeforces.com/contest/844/problem/E (5) //Post-Order → line, Connect i → N-2: star
http://www.spoj.com/problems/CAC/ (5) //VERY NICE! — Find all cycles in cactus

http://codeforces.com/contest/849/problem/C (3) //State search by gauss

http://codeforces.com/contest/846/problem/E (5) //NICE: DFS + some overflow logic
http://www.spoj.com/problems/KOZE/ (3) //NICE: Floods
http://www.spoj.com/problems/RIOI_2_3/ (4) //DFS /OR/ BFS /OR/ DSU [NICE][EASY][BF]
http://www.spoj.com/problems/MAKEMAZE/ (3) //EASY — Simple dfs on grid

http://codeforces.com/contest/861/problem/F (5) //VERY NICE: Modify dfs tree so it remains connected
http://www.spoj.com/problems/GHOSTS/ (3) //NICE — must remain dag after each QR
http://www.spoj.com/problems/AMR10J/ (5) //VERY NICE! — DFS+DP [DAG with cycles]

http://codeforces.com/contest/24/problem/A (2)//NICE [DFS-ON-CYCLE]

http://codeforces.com/contest/29/problem/C (3) //Find begining/end of line (graph)

http://codeforces.com/contest/29/problem/D (4) //Tree [implementation][simulation]

Dijkstra

http://codeforces.com/contest/716/problem/D 7

12047 UVA 4

11514 UVA 4

http://codeforces.com/contest/757/problem/F 7

11338 UVA (4)

11374 UVA (4)

11097 UVA (4) //Divide to N*1000 nodes and go!

13172 UVA (5) //6*DJ per query + permutations

10816 UVA (4) //Easy Linear-Search by answer + DJ with path

http://codeforces.com/contest/827/problem/F 7 //Very nice — Even&Odd

http://www.spoj.com/problems/DELIVER/ (5) //Normalize coordinates + Optimalize

http://www.spoj.com/problems/CCHESS/ (4) //Dijkstra as knight

DSU

http://codeforces.com/contest/723/problem/F 7

13153 UVA (5)

13169 UVA (3)

11987 UVA (3)

11474 UVA (4)

http://codeforces.com/contest/687/problem/D 6

http://codeforces.com/contest/680/problem/E 7 //+precalculation/brute force

http://codeforces.com/contest/766/problem/D 5

http://www.spoj.com/problems/LEXSTR/ (3) //Nice na stringu

http://codeforces.com/contest/805/problem/C 3 //NICE (dijkstra like :P)

http://www.spoj.com/problems/IITKWPCI/ (3) //VERY NICE

http://www.spoj.com/problems/FRNDCIRC (3) //Classical DSU (NICE for practice)

http://www.spoj.com/problems/FOXLINGS/ (3) Easy — just renumbering

http://www.spoj.com/problems/NITTROAD/ (4) //Process from back

http://www.spoj.com/problems/SHAHBG/ (2) //DSU not needes (simulated by array)

http://codeforces.com/contest/598/problem/D (3) //Can be solved with DFS too

http://codeforces.com/contest/9/problem/E (4) //Making one big cycle

http://codeforces.com/contest/25/problem/D (4) //Could be done linear too

http://codeforces.com/contest/28/problem/B (4) //NICE [imho bad statement]

Euler Tour
Flows
Floyd Warshall
Graph

http://codeforces.com/contest/27/problem/D (5)

11387 (UVA) 4

http://www.spoj.com/problems/VFRIEND2/ (5) //Graph possible check

http://codeforces.com/contest/859/problem/E (4) //VERY NICE (2 cases: CYCLE [x2] / TREE [x(Size+1)]

http://codeforces.com/contest/847/problem/C (2) //Forest making Easy&Nice

http://codeforces.com/contest/863/problem/C (3) //Cycle in states