Crossword Puzzle

locked

by PRASHANTB1984

Editorial by rock19

Recursively, try every possible valid position to place a word.

Suppose we have k words, then start from first word and try to place it at valid position in the grid, either horizontally or vertically, and then move ahead with the updated grid for the next word. If for any word there is no valid position found, then backtrack and restore the previous grid(i.e, grid before when last word is not placed) and so on.

If we place all the words in the grid for any configuration, then print the grid as an answer and terminate the resursion.

Statistics

Difficulty: **Medium**
Required Knowledge: **recursion brute-force**
Publish Date: **Mar 09 2017**

Set by rock19

Problem Setter's code :

```
#include <bits/stdc++.h>
using namespace std;

vector<string> grid(10);
vector<string> words;
bool f;

void call(int ind)
{
    if(!f) {
        return;
    }
    if(ind == words.size()) {
        if(f) {
            for(auto word: grid) {
                cout<<word<<endl;
            }
            f=false;
        }
        return;
    }
    int i,j,p,q,k;
    for(i=0;i<10;++i) {
        for(j=0;j<10;++j) {
            p=i,q=j;
            for(k=0;k<words[ind].size() && p+k<10;++k) {
                if(grid[p+k][q] != '-' && grid[p+k][q] != words[ind][k]) {
                    break;
                }
            }

            if(k==words[ind].size()) {
                vector<string> temp = grid;
                for(k=0;k<words[ind].size();++k) {
                    grid[p+k][q] = words[ind][k];
                }
                call(ind+1);
                grid = temp;
            }

            for(k=0;k<words[ind].size() && q+k<10;++k) {
                if(grid[p][q+k] != '-' && grid[p][q+k] != words[ind][k]) {
                    break;
                }
            }

            if(k==words[ind].size()) {
                vector<string> temp = grid;
                for(k=0;k<words[ind].size();++k) {
                    grid[p][q+k] = words[ind][k];
                }
                call(ind+1);
                grid = temp;
            }
        }
    }
}

int main()
{
    f=true;

    int i,j;
    for(i=0;i<10;++i) {
        cin>>grid[i];
    }

    string s,w;
    cin>>w;

    for(auto x: w) {
        if(x==';') {
            words.push_back(s);
            s="";
        } else
            s+=x;
    }
    words.push_back(s);
    call(0);

    return 0;
}
```