Computer Science 320SC – (2018)

Programming Assignment 6

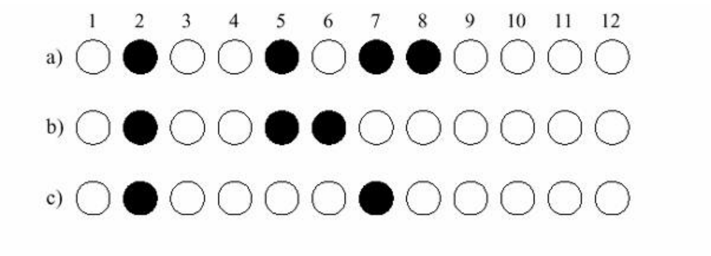
Due: Saturday, October 20th (11:58pm)

Requirements

This sixth assignment lets you get familiar with further algorithm design techniques covered in class (e.g. network flow). There are two algorithms/programs required but three different submissions (two data sets for one of them). It is worth 5% of your total course marks. Please try and test with your own generated input cases before submitting to the automated marker.

Task 1: Peg Removal

We have a few pegs placed in a line of twelve holes. We want to remove pegs by jumping over other pegs. Jumped over pegs disappear as a result of a move. A move is possible if there is a straight line of three adjacent holes, where two adjecent holes contain a peg and the third is empty. The outside peg may jump over the middle peg and be placed in the vacant hole on the other outside position. To illustrate these moves consider the following figure (black nodes are pegged holes). Peg at position 8 jumps to the hole at position 6 then the peg at position 5 jumps past the peg at position 6 and ends up at position 7.



Your goal is to find a sequence of moves such that as few pegs as possible are left.

The input consists of several cases. The first line is an integer denoting how many. The subsequent lines are text strings of 12 characters in length consisting of ‘-’ (minus) and ‘o’ (little oh). An ’o’ represents an peg and an ‘-’ represents an empty hole. Output an integer denoting the smallest number of remaining pegs obtained by some sequence of moves.

Sample Input:

```
5
--ooo--oo---
ooooo-----oo
oooooooooooo
-o-----oo---
-oo-oo-oo---
```

Sample Output: for program named `peg.ext`

```
2
4
12
2
2
```

Task 2: UN Laptops

The United Nations (UN) is holding a computer convention and various delagates from around the world come with laptops and want to plug them into outlets at the venue. Unfortunately, not all countries use the same type of plugs. The UN venue supports a few popular types directly. Also they have an unlimited supply of various types of adaptors. A delagate may need one or more adaptors to connect and power his/her computer. Your task is to decide the minimum number of delagates that are left without power, given a set of outlets (of the building), types of adaptors available, and the types of plugs (of the laptops).

The input will consist of several cases as specified by an integer on the first line.

Each case is then given in three parts. The first line is an integer $1 \leq n \leq 500$ indicating the number of outlets in the building. The next n lines consist of a string indicating the type of each outlet. The next line of the case consists of an integer $1 \leq m \leq 500$ indicating the number of laptops. The next m lines consist of a string indicating the type of the plug for each laptop. The next line of the case consists of an integer $1 \leq k \leq 1000$ indicating the number of adaptors types available. This is followed by k lines of two strings separated by one space. These lines indicate the plug conversion for an adaptor (first type to second type). All strings will be alphanumeric and of length at most 25.

For example in the following input the number of cases is 2. For the first case we have three outlets of two types (one NZ and two British), then three laptops (two have NZ plugs and one has a British plug) Finally we have two types of adapters: the first one converts NZ to American and the other one converts American to British. All three laptops can be used, with one of the NZ plugs using both adaptors.

Sample Input:

```
2
3
NZ
British
British
3
NZ
British
NZ
2
NZ American
American British
4
A
B
C
D
5
B
C
B
B
X
3
B X
A X
X D
```

Sample Output:

```
0
2
```

Submit programs `laptopsE.ext` and `laptopsH.ext` to test easy and harder data sets.

Submission Procedure

For each of these submissions you may use the folling extensions for `ext` to indicate which language/compiler to use: { `java`, `cpp`, `py`, `cs` }, that indicates Java/C++/Python/C#(mono). The automarker runs on a Debian linux box and uses the most recent compilers that are packaged by them. Please use just one source file per problem.

For this assignment there is two marks awarded for Task 1 and and three marks awarded for Task 2.