```
# Shortest Cycle
# Copyright (C) 2007 Alasdair Turner
# This program is free software; you can redistribute it and/or
# modify it under the terms of the GNU General Public License
# as published by the Free Software Foundation; either version 2
# of the License, or (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
push_list = []
pop_list = []
live_paths = []
setmark([-1,0])
depth = 1
path_index = 0
for i in connections():
     pop_list.append([path_index,i])
     live_paths.append(1)
     i.setmark([path_index,depth])
     path_index = path_index + 1
if path_index < 2:
     return -1 # no cycle possible
live_path_count = path_index
 while live_path_count > 1:
     curs = pop_list.pop()
     path_index = curs[0]
     this_node = curs[1]
     live_paths[path_index] = live_paths[path_index] – 1
     for i in this_node.connections():
         if i.mark() is none:
             i.setmark([path_index,depth+1])
             push_list.append([path_index,i])
             live_paths[path_index] = live_paths[path_index] + 1
         elif i.mark()[0] != path_index and i.mark()[0] != -1:
             # found a cycle!
             return i.mark()[1] + this_node.mark()[1] + 1
     if live_paths[path_index] == 0:
         live_path_count = live_path_count – 1
     if len(pop_list) == 0:
         depth = depth + 1
         pop_list = push_list
         push_list = []
return -1 # no cycle found
```