**COSC 352 Project 1 Solution - DrRacket**

#lang racket

(define Graph

'((A (B E))

(B (E F))

(C (D))

(D ())

(E (C F))

(F (D G))

(G ())))

(define (neighbors a-node a-graph)

(cond

((empty? a-graph) (error 'neighbors "can't happen"))

(else (cond

((eq? (first (first a-graph)) a-node)

(second (first a-graph)))

(else (neighbors a-node (rest a-graph)))))))

(define (find-route origination destination graph)

(cond

((eq? origination destination) (list destination))

(else (local ((define possible-route

(find-route/list (neighbors origination graph) destination graph)))

(cond

((boolean? possible-route) #f)

(else (cons origination possible-route)))))))

(define (find-route/list lo-Os D graph)

(cond

((empty? lo-Os) #f)

(else (local ((define possible-route (find-route (first lo-Os) D graph)))

(cond

((boolean? possible-route) (find-route/list (rest lo-Os) D graph))

(else possible-route))))))

> Graph

'((A (B E)) (B (E F)) (C (D)) (D ()) (E (C F)) (F (D G)) (G ()))

> (neighbors 'E Graph)

'(C F)

> (find-route 'A 'B Graph)

'(A B)

> (find-route 'A 'F Graph)

'(A B E F)

>

\*\*

File

Open …

project1

Open

Etc.