THE TRAVELLING SALESMAN PROBLEM: 2-OPT HEURISTIC

RAESETJE BONJO SEFALA- 844165

APRIL 19, 2017

WRITTEN IN PYTHON

```
################ IT STARTS HERE #################################################
# -*- coding: utf-8 -*-
"""
Created on Thu Apr 20 01:07:03 2017

@author: amyse
"""

#!/usr/local/bin/python

#Traveling Salesman Solution using 2-opt Algorithm
#Raesetje Bonjo Sefala 844165

import math, numpy

def getDistance(city1, city2):
          return   math.sqrt((int(city2[0]) - int(city1[0]))**2 + (int(city2[1])-int(city1[1]))**2)

#A function to get the total weight of a path

def getWeight(perm):
          #Set the initial distance to 0
   dist = 0
   perm=cities
          #Calculate and add the distance between each city
   for i in range(len(perm)-1):
      dist += getDistance(perm[i,:], perm[i+1,:])

          #to add the final city back to the initial city to the total dist
   dist += getDistance(perm[-1,:], perm[0,:])
          #We now have the total distance so return it
   return dist

   results = 0

   next =numpy.copy(cities)

   weight=getWeight(next)
   for i in range(0,(len(cities)-3)):
      for j in range(i+2, len(cities)-2):
         ii=i+1
         jj=j-1
         tmp=numpy.copy(next[i,:])
         next[i,:]=numpy.copy(next[j,:])
         next[j,:]=numpy.copy(tmp)

         for nw in range(0,(jj)):
            next[ii+nw,:]=numpy.copy(cities[(jj-nw),:])
```

```
        for pw in range(j+1,len(cities)-1):
                next[pw,:]=numpy.copy(cities[pw,:])

        new_weight = getWeight(next)
          #If the new tour is better than the old tour, set new tour as current best
        if new_weight <= weight:
           best = numpy.copy(next)
        else:
          best= numpy.copy(next)

   results = new_weight
          #Return an arbitrary path and the weight
   return [best, results]

##################################################################
#initializations
cities = numpy.random.random_integers(0, 100, (10, 2))
opt_tour = two_opt(cities)
print ('The optimum tour is: %s (%f)' % (opt_tour[0], opt_tour[1]))
print ('There are %d cities in this tour.' % (len(opt_tour[0])))
```