Practical No: 10

Aim: Path finding using Ant Colony Optimization with an application.

Step1: Install the package.

```
julia> using Pkg

julia> Pkg.add("AntColony")
  Installing known registries into `C:\Users\RDNC\.julia`
    Updating registry at `C:\Users\RDNC\.julia\registries\General.toml`
   Resolving package versions...
   Installed AntColony — v0.1.1
```

Step2: Create a distance matrix which defines the cost of travelling between nodes. Note that this matrix represents a directed graph and therefore the matrix need not be symmetric.

```
julia> using AntColony

julia> distance_matrix = rand(10, 10)
10×10 Matrix{Float64}:
 0.626576   0.215774  0.910835   0.233799   0.158596  0.462252   0.88989   0.352988   0.261076  0.294935
 0.686963   0.442676  0.812288   0.386587   0.761905  0.670398   0.346865  0.449512   0.390563  0.422448
 0.0833134  0.898298  0.353293   0.135502   0.75569   0.829764   0.738392  0.0197433  0.632107  0.838567
 0.736643   0.352804  0.682236   0.870784   0.499835  0.0085831  0.135644  0.114847   0.969153  0.0776233
 0.244256   0.292848  0.0505338  0.202153   0.134019  0.591827   0.793424  0.413879   0.975119  0.430897
 0.506635   0.340756  0.97189    0.0406632  0.907445  0.763469   0.351253  0.362993   0.258515  0.666049
 0.455432   0.327666  0.835823   0.770899   0.208032  0.792952   0.460538  0.707352   0.769139  0.232132
 0.337398   0.882639  0.61941    0.317005   0.39938   0.445222   0.89962   0.805825   0.412226  0.687299
 0.693259   0.987839  0.289205   0.844867   0.926623  0.556758   0.252484  0.0569513  0.416715  0.0671433
 0.378485   0.79057   0.325417   0.331642   0.989294  0.749057   0.155219  0.115432   0.724047  0.493602
```

Step3: Find a path which visits all nodes and returns to the start node. Note the function doesn't return the start node twice.

```
julia> aco(distance_matrix, is_tour = true)
10-element Vector{Int64}:
 10
  9
  6
  4
  3
  5
  7
  2
  1
  8
```

Step 4: Find a path with a specific start and end node

```
julia> aco(distance_matrix, start_node = 1, end_node = 5)
10-element Vector{Int64}:
  1
  3
  9
  6
  4
  8
 10
  7
  2
  5
```