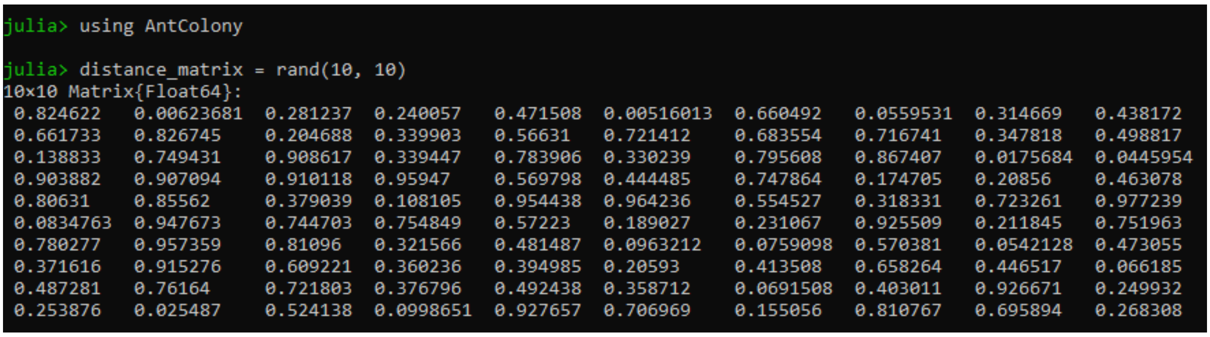
Algorithm for Optimization

Practical No.7

AIM: Path finding using Ant Colony Optimization with an Application

using AntColony

distance_matrix = rand(10, 10)

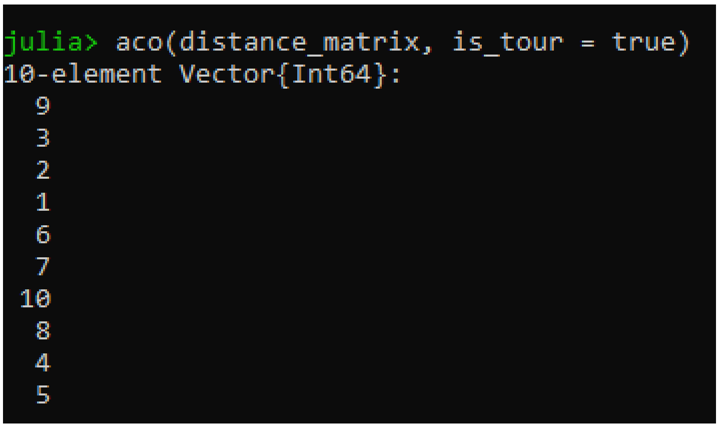


```
julia> using AntColony

julia> distance_matrix = rand(10, 10)
10×10 Matrix{Float64}:
 0.824622   0.00623681  0.281237  0.240057   0.471508  0.00516013  0.660492   0.0559531  0.314669   0.438172
 0.661733   0.826745    0.204688  0.339903   0.56631   0.721412    0.683554   0.716741   0.347818   0.498817
 0.138833   0.749431    0.908617  0.339447   0.783906  0.330239    0.795608   0.867407   0.0175684  0.0445954
 0.903882   0.907094    0.910118  0.95947    0.569798  0.444485    0.747864   0.174705   0.20856    0.463078
 0.80631    0.85562     0.379039  0.108105   0.954438  0.964236    0.554527   0.318331   0.723261   0.977239
 0.0834763  0.947673    0.744703  0.754849   0.57223   0.189027    0.231067   0.925509   0.211845   0.751963
 0.780277   0.957359    0.81096   0.321566   0.481487  0.0963212   0.0759098  0.570381   0.0542128  0.473055
 0.371616   0.915276    0.609221  0.360236   0.394985  0.20593     0.413508   0.658264   0.446517   0.066185
 0.487281   0.76164     0.721803  0.376796   0.492438  0.358712    0.0691508  0.403011   0.926671   0.249932
 0.253876   0.025487    0.524138  0.0998651  0.927657  0.706969    0.155056   0.810767   0.695894   0.268308
```

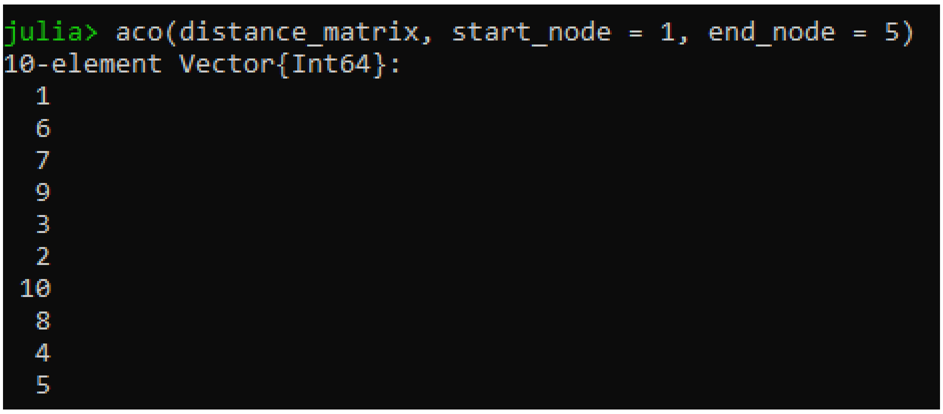
# note that distance_matrix[3, 5] = travel distance from node 5 to node 3

aco(distance_matrix, is_tour = true)



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

aco(distance_matrix, start_node = 1, end_node = 5)



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