SIMULATED ANNEALING

```
current_solution <- greedy(matrix)[0]
best_solution <- copy current_solution

while  temperature >1 do
        new_solution <- copy current_solution
        tour_pos1 <- 0
        tour_pos2 <- 0
end while
        while ((tour_pos1= tour_pos2) or tour_pos1 =0 and tour_pos2 = length of matrix -1 or tour_pos2 = 0
and tour_pos1 = length of matrix -1))
                tour_pos1 <- random(0, length of matrix)
                tour_pos2 <-random (0, length of matrix)
        end while
        if tour_pos1= 0 or tour_pos1 =length of new solution -1:
                city1 <- new solution [tour_pos1]
                city2 <- new solution[tour_pos2]
                new solution[0] <- city2
                new solution[ length of new solution -1] <- city2
                new solution [tour_pos2] <-city1
        else if tour_pos2 = 0 or tour_pos2 = length of new solution -1:
                city1 <- new solution[tour_pos2]
                city2 <- new solution[tour_pos1]
                new solution[0] <-city2
                new solution[ length of new solution -1] <- city2
                new solution[tour_pos1] <- city1
        else:
                city1 <- new solution[tour_pos1]
                city2  <- new solution [tour_pos2]
                new solution [tour_pos1] <- city2
                new solution [tour_pos2] <- city1
        end if
        current energy <- function path_cost( matrix, current solution)
        new energy <- fucntion path_cost( matrix, new solution)

        if  function accept probability( current energy, new energy, temperature) > random number:
                current solution <- copy of new solution
        end if
        if function path_cost( matrix, current solution) < function path_cost(matrix, best solution):
                best solution <- copy of current solution
                OUTPUT “Better solution: “, function path_cost( matrix, best solution))
                temperature <- temperature * (1- cooling rate)
        end if
return best solution, function path_cost(matrix, best solution)
```