Algoritmo

CreateRandomCoordinates

Entrada

Valor N entero

Algoritmo Ramdom

Salida

Secuencia coordinates de coordenadas

Pseudocodigo

coordinates<-[n][2]

para i=0 hasta que i<coordinates.lenght {

coordinates[i][0]<- Math.abs(random.nextInt()%MAP\_SIZE);

coordinates[i][1]<- Math.abs(random.nextInt()%MAP\_SIZE);

i<-i++

}

Res<-coordinates

Algoritmo

initCostsByCoordinates

Entrada

Valor i,j

Salida

Pseudocodigo

Para i=0 hasta que i<coordinates.lenght{

Para j=0 hasta que j<coordinates.lenght{

Costs[i][j]<- calculateTravelCostsBetweenCities(i,j);

j<-j++

}

i<-i++

}

Algoritmo

calculateTravelCostsBetweenCities

Entrada

Valor i,j

Salida

Valor n de costo de viaje entre ciudades

Pseudocodigo

Int dx <- coordinates[i][0]-coordinates[j][0];

Int dy<- coordinates[i][1]-coordinates[j][1];

Res<- Math.sqrt(dx\*dx+dy\*dy);

Algoritmo

calculateCosts

Entrada

Secuencia route

Algoritmo Ramdom

¿?

Pseudocodigo

Res<- calculateCosts(route, false);

Algoritmo

calculateCosts

Entrada

Secuencia route

Valor isVerbose boolean

Salida

Valor travelCost costo de viaje

Pseudocodigo

travelCosts<-0

para i=1 hasta que i=>route.length {

travelCost<-travelcost+ costs[route[i-1]][route[i]];

i++;

Si isVerbose==verdadero{

System.out.println("costs from "+route[n-1]+" to "+route[0]+": "+costs[route[n-1]][route[0]]);

}

}

travelCosts <-travelCost+ costs[route[n-1]][route[0]];

si isVerbose ==true{

System.out.println("costs from "+route[n-1]+" to "+route[0]+": "+costs[route[n-1]][route[0]]);

}

Res<-travelCost

Algoritmo

printRoute

Entrada

Secuencia route

Salida

Pseudocodigo

Para i=0 hasta que i=>route.lenght {

System.out.print(route[i]+" ");

I++

}

Algoritmo

printCosts

Entrada

Salida

Pseudocodigo

System.out.println("costs matrix for the traveling salesman problem:");

Para i=0 hasta que i>=costs.lenght{

Para j=0 hastq que j=>costs[i].lenght{

System.out.print(costs[i][j]+" ");

J++

}

I++

System.out.print("\n");