INFO 6205 Spring 2023 Project

Travelling Salesman

Aakash Rajawat (NUID: 002764127)
Ruchita Iyer (NUID: 002768966)

Introduction

Aim: is to develop an efficient algorithm to solve this classic optimization problem. The Travelling Salesman Problem involves finding the shortest possible route that a salesman can take to visit a set of cities exactly once and then return to the starting city. The project aims to identify the most efficient algorithm or combination of algorithms to solve TSP for different scenarios such as varying numbers of cities and different distributions of city locations. The goal would be to develop a solution that can solve TSP instances with many cities in a reasonable amount of time, making it applicable to real-world scenarios.

Approach: We started the travelling salesman problem with Christofides algorithm. We then chose two algorithms each from Tactical method and Strategic method to see which one optimises the problem further and gives the shortest route.

Program

Data Structures & classes

Data Structure:

- Lists
- ArrayLists

Classes:

class TSPRandomSwapping-

- private double distanceTo(City city): Haversin's distance function
- private static List<City> readCities(String filename): Read CSV file and create city objects
- public static double calculateDistance(List<City> tour): Calculate the total distance of the tour
- public static List<City> createTour(List<City> cities): Create a random tour
- public static List<City> swapCities(List<City> tour): Swap two cities in the tour randomly

class TSPSimulatedAnnealing-

- private static void readCSV(String filePath): Helper method to read input CSV file
- public static int getRandomIndex(int range): Helper method to get a random index within a given range
- private void calculateDistance(): Helper method to calculate the total distance of the tour
- public static double getDistanceBetweenCities(City city1, City city2): Helper method to calculate the distance between two cities
- public void swapCities(int index1, int index2): Helper method to swap two cities in the tour

class TspChristofides-

- public double distanceTo(City other)
- public static List<City> readCitiesFromCSV(String filename): Method to read input CSV file
- public static List<City> buildMinimumSpanningTree(List<City> cities): Method to build the spanning tree
- public static List<City> findOddDegreeVertices(List<City> cities): Method to find odd degree vertices
- public static List<City> buildMinimumWeightPerfectMatching(List<City> cities): Method to match vertices with perfect weight matching
- public static List<City> findEulerianTour(List<City> cities): Method to calculate the tour distance
- public static double calculateTourLength(List<City> tour): Method to find the total distance

class TSP2Opt-

- public static double distance (Point a, Point b): Calculate the distance between two points on the earth's surface
- public static double tourDistance(List<Integer> tour, double[][] distances): Calculate the distance of a tour
- public static List<Integer> twoOptSwap(List<Integer> tour, int i, int j): Swap two edges in a tour using the 2-opt algorithm

class TSPGeneticAlgorithm-

- public List<City> solve(): Method to initialize a population, evolve it, and return the optimal route
- public List<Route> initPopulation(): Method to create a population of random routes
- public List<City> shuffle (List<City> cities): Method to shuffle a list of cities

- private List<Route> evolvePopulation(List<Route> population): method to select parents, perform crossover and mutation, and create a new population
- private Route selectParent(List<Route> population): Method to randomly select a parent route from the population of routes
- private Route crossover(Route parent1, Route parent2): Method to create a child route
- private void mutate(Route route): Method to randomly swaps two cities in a given route
- private Route getBestRoute(List<Route> population): Returns the best route from the population
- public static List<City> readCitiesFromFile(String fileName): Method to read input CSV file
- public static double calculateDistance(City city1, City city2): Method to calculate the distance between two cities
- public static double calculateTotalDistance(List<City> cities): Method to calculate the total distance of the tour

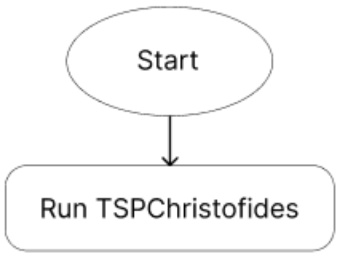
Algorithm

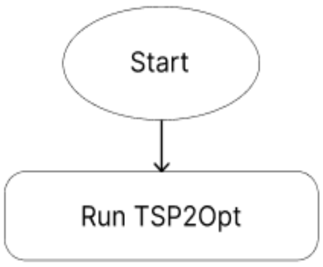
- **Christofides**- The Christofides algorithm is an algorithm for finding approximate solutions to the travelling salesman problem, on instances where the distances form a metric space (they are symmetric and obey the triangle inequality). It is an approximation algorithm that guarantees that its solutions will be within a factor of 3/2 of the optimal solution length.
- **2 opt**- The algorithm essentially iterates along a path where the cost is known and cuts and combines the path until a desired or optimal solution is found. One feature of the algorithm is that it can be terminated when a desired length or cost is achieved. It takes a route that crosses over itself and reorder it so that it does not. A complete 2-opt local search will compare every possible valid combination of the swapping mechanism.
- **Random Swapping**- Random swap algorithm aims at solving clustering by a sequence of prototype swaps, and by fine-tuning their exact location by k-means. This randomized search strategy is simple to implement and efficient.
- **Simulated Annealing**- Simulated Annealing is a stochastic global search algorithm which means it uses randomness as part of its search for the best solution. SA uses this concept of temperature in determining the probability of transitioning (stochasticity of the search) to a worse solution in order to more widely explore the search space and have a better chance of finding the global optimum.
- **Genetic Algorithm**- Genetic algorithms are heuristic search algorithms inspired by the process that supports the evolution of life. The algorithm is designed to replicate the natural selection process to carry generation, i.e. survival of the fittest of beings. The genetic algorithm depends on selection criteria, crossover, and mutation operators.

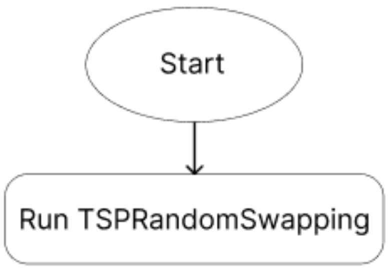
Invariants

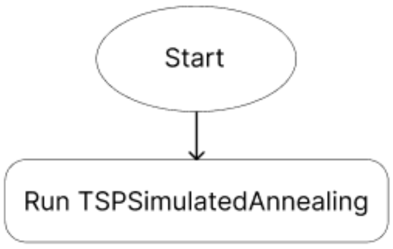
- The input is a complete undirected graph: There must be a weighted edge connecting every pair of cities in the set, and the weight of each edge represents the distance between the two cities.
- The objective is to find a Hamiltonian cycle with minimum weight: A Hamiltonian cycle is a cycle that passes through every vertex exactly once and returns to the starting vertex.
- The solution is a valid TSP tour: The solution must visit each city exactly once and return to the starting city, forming a Hamiltonian cycle.
- The solution is optimal: The solution found by the algorithm is the shortest possible Hamiltonian cycle, i.e., it has the minimum total weight among all Hamiltonian cycles in the graph.

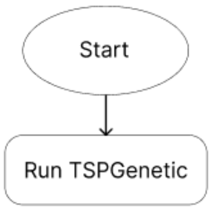
Flow Charts (inc. UI Flow)





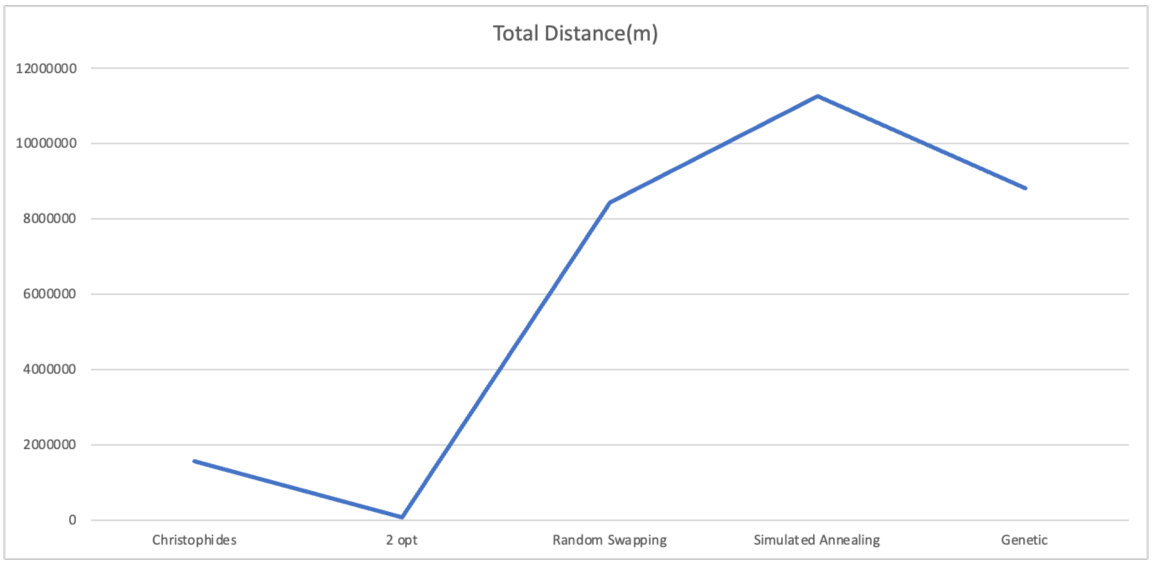


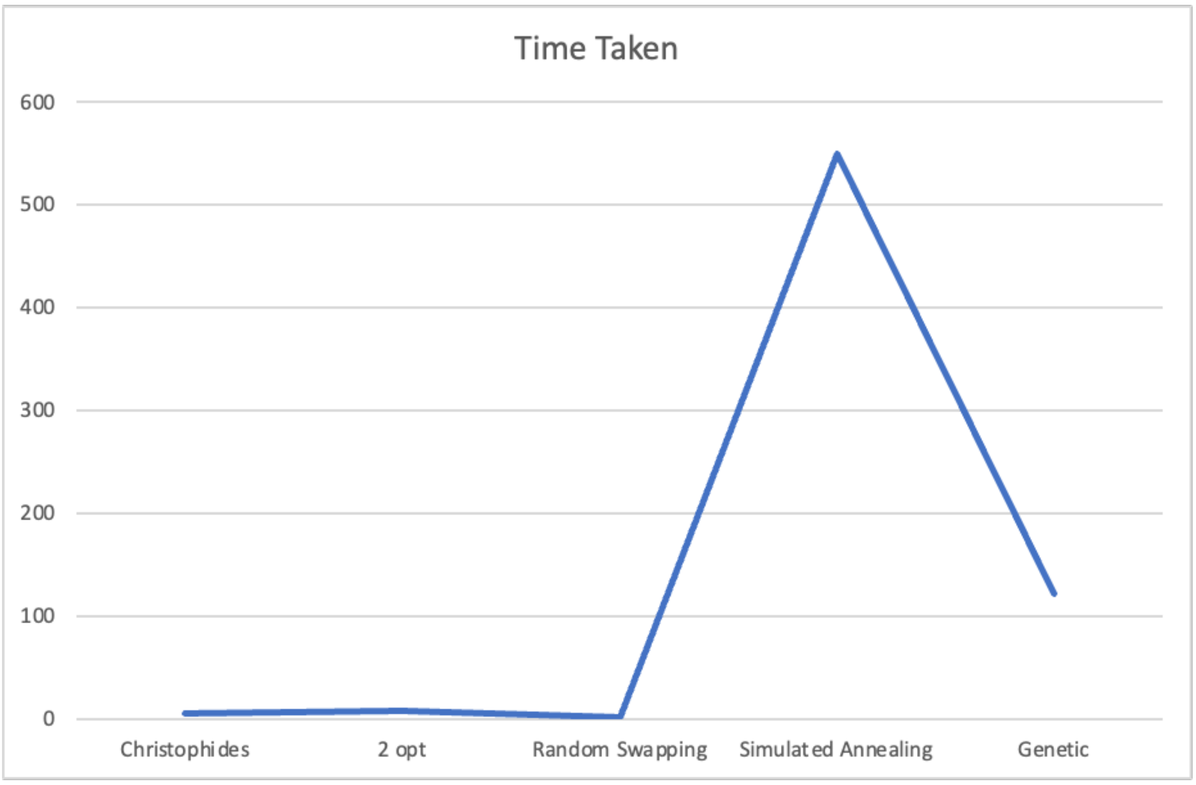




Observations & Graphical Analysis

Algorithm	Total distance (m)	Time taken (sec)
Christofides	1560778.12	5.39
2 opt	648786.5	7.89
Random Swapping	8438998.49	1.24
Simulated Annealing	11266889.71	550
Genetic	8809403.54	121





Results & Mathematical Analysis

Mathematical Formula used:

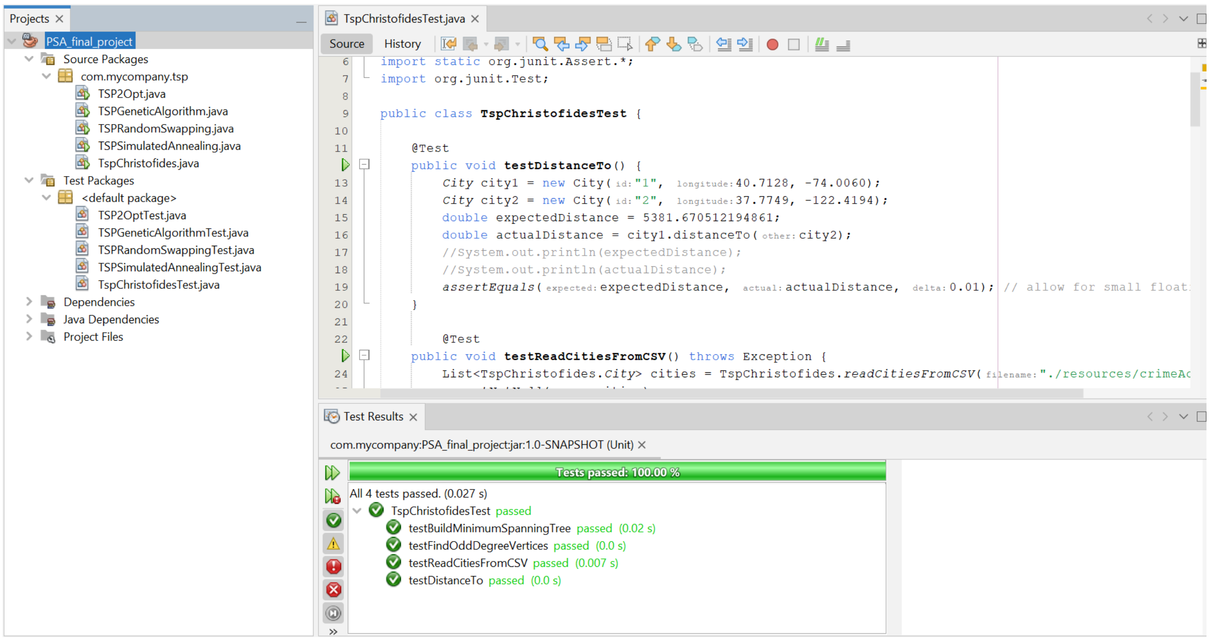
```
double dLat = Math.toRadians(lat2 - lat1);
double dLon = Math.toRadians(lon2 - lon1);
double a = Math.sin(dLat / 2) * Math.sin(dLat / 2)
            + Math.cos(Math.toRadians(lat1)) * Math.cos(Math.toRadians(lat2))
            * Math.sin(dLon / 2) * Math.sin(dLon / 2);
double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
return R * c * 1000;
```

Where R= radius of the earth

Here the distance between two points is calculated with respect to the radius of Earth.

Unit tests:

Christofides algorithm



```
Projects
PSA_final_project
  Source Packages
    com.mycompany.tsp
      TSP2Opt.java
      TSPGeneticAlgorithm.java
      TSPRandomSwapping.java
      TSPSimulatedAnnealing.java
      TspChristofides.java
  Test Packages
    <default package>
      TSP2OptTest.java
      TSPGeneticAlgorithmTest.java
      TSPRandomSwappingTest.java
      TSPSimulatedAnnealingTest.java
      TspChristofidesTest.java
  Dependencies
  Java Dependencies
  Project Files
```

TspChristofidesTest.java

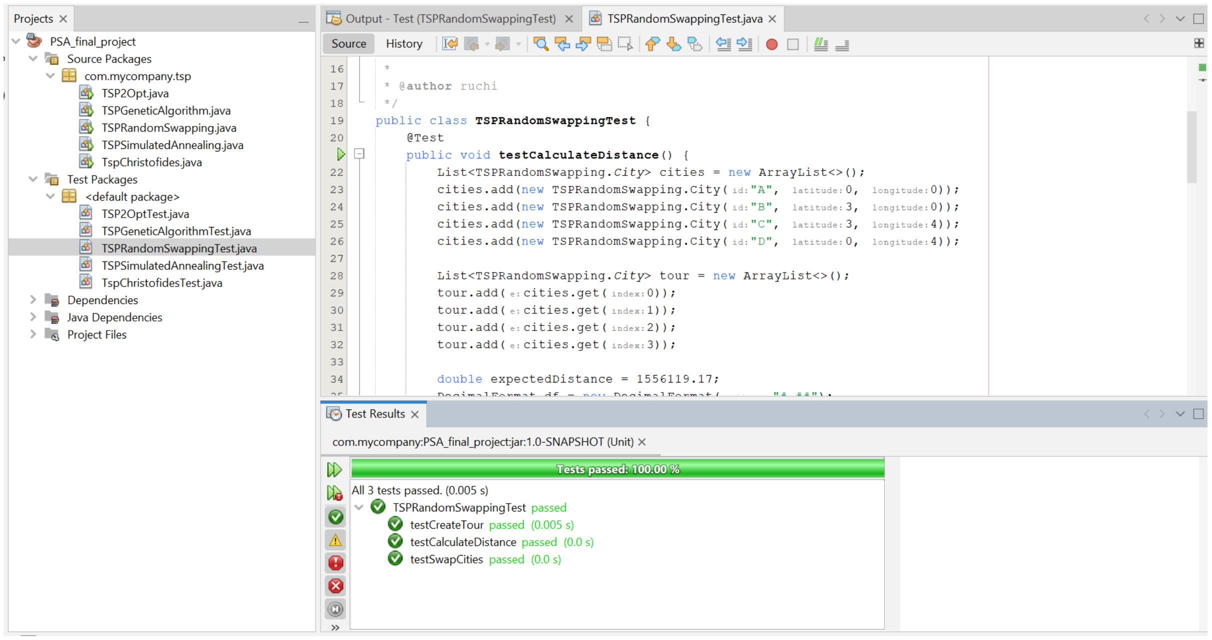
```java
import static org.junit.Assert.*;
import org.junit.Test;

public class TspChristofidesTest {

    @Test
    public void testDistanceTo() {
        City city1 = new City( id:"1",  longitude:40.7128, -74.0060);
        City city2 = new City( id:"2",  longitude:37.7749, -122.4194);
        double expectedDistance = 5381.670512194861;
        double actualDistance = city1.distanceTo( other:city2);
        //System.out.println(expectedDistance);
        //System.out.println(actualDistance);
        assertEquals( expected:expectedDistance,  actual:actualDistance,  delta:0.01); // allow for small float
    }

        @Test
    public void testReadCitiesFromCSV() throws Exception {
        List<TspChristofides.City> cities = TspChristofides.readCitiesFromCSV( filename:"./resources/crimeAc
```

Test Results

com.mycompany:PSA_final_project:jar:1.0-SNAPSHOT (Unit)

```
Tests passed: 100.00 %
All 4 tests passed. (0.027 s)
  TspChristofidesTest passed
    testBuildMinimumSpanningTree passed (0.02 s)
    testFindOddDegreeVertices passed (0.0 s)
    testReadCitiesFromCSV passed (0.007 s)
    testDistanceTo passed (0.0 s)
```

Random Swapping algorithm



```
Projects
PSA_final_project
  Source Packages
    com.mycompany.tsp
      TSP2Opt.java
      TSPGeneticAlgorithm.java
      TSPRandomSwapping.java
      TSPSimulatedAnnealing.java
      TspChristofides.java
  Test Packages
    <default package>
      TSP2OptTest.java
      TSPGeneticAlgorithmTest.java
      TSPRandomSwappingTest.java
      TSPSimulatedAnnealingTest.java
      TspChristofidesTest.java
  Dependencies
  Java Dependencies
  Project Files
```

Output - Test (TSPRandomSwappingTest) | TSPRandomSwappingTest.java

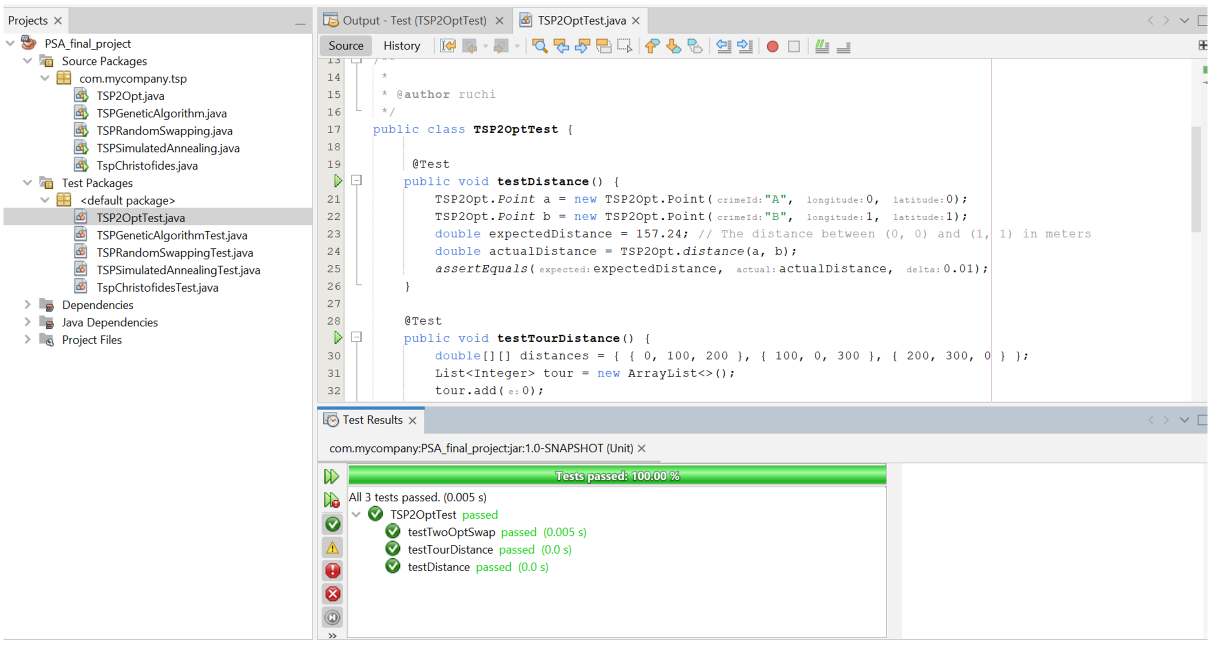
```java
 *
 * @author ruchi
 */
public class TSPRandomSwappingTest {
    @Test
    public void testCalculateDistance() {
        List<TSPRandomSwapping.City> cities = new ArrayList<>();
        cities.add(new TSPRandomSwapping.City( id:"A",  latitude:0,  longitude:0));
        cities.add(new TSPRandomSwapping.City( id:"B",  latitude:3,  longitude:0));
        cities.add(new TSPRandomSwapping.City( id:"C",  latitude:3,  longitude:4));
        cities.add(new TSPRandomSwapping.City( id:"D",  latitude:0,  longitude:4));

        List<TSPRandomSwapping.City> tour = new ArrayList<>();
        tour.add( e:cities.get( index:0));
        tour.add( e:cities.get( index:1));
        tour.add( e:cities.get( index:2));
        tour.add( e:cities.get( index:3));

        double expectedDistance = 1556119.17;
```

Test Results

com.mycompany:PSA_final_project:jar:1.0-SNAPSHOT (Unit)

```
Tests passed: 100.00 %
All 3 tests passed. (0.005 s)
  TSPRandomSwappingTest passed
    testCreateTour passed (0.005 s)
    testCalculateDistance passed (0.0 s)
    testSwapCities passed (0.0 s)
```

2 opt algorithm



```
Projects
PSA_final_project
  Source Packages
    com.mycompany.tsp
      TSP2Opt.java
      TSPGeneticAlgorithm.java
      TSPRandomSwapping.java
      TSPSimulatedAnnealing.java
      TspChristofides.java
  Test Packages
    <default package>
      TSP2OptTest.java
      TSPGeneticAlgorithmTest.java
      TSPRandomSwappingTest.java
      TSPSimulatedAnnealingTest.java
      TspChristofidesTest.java
  Dependencies
  Java Dependencies
  Project Files
```

Output - Test (TSP2OptTest) | TSP2OptTest.java

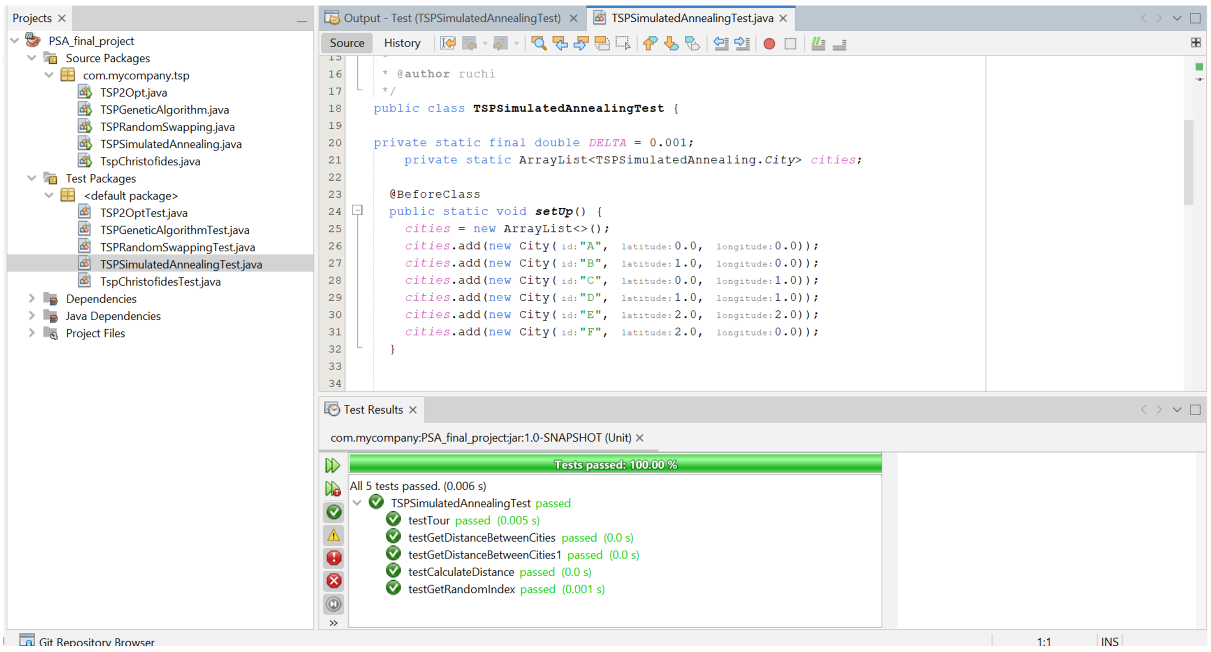
```java
 *
 * @author ruchi
 */
public class TSP2OptTest {

     @Test
    public void testDistance() {
        TSP2Opt.Point a = new TSP2Opt.Point( crimeId:"A",  longitude:0,  latitude:0);
        TSP2Opt.Point b = new TSP2Opt.Point( crimeId:"B",  longitude:1,  latitude:1);
        double expectedDistance = 157.24; // The distance between (0, 0) and (1, 1) in meters
        double actualDistance = TSP2Opt.distance(a, b);
        assertEquals( expected:expectedDistance,  actual:actualDistance,  delta:0.01);
    }

    @Test
    public void testTourDistance() {
        double[][] distances = { { 0, 100, 200 }, { 100, 0, 300 }, { 200, 300, 0 } };
        List<Integer> tour = new ArrayList<>();
        tour.add( e:0);
```

Test Results

com.mycompany:PSA_final_project:jar:1.0-SNAPSHOT (Unit)

```
Tests passed: 100.00 %
All 3 tests passed. (0.005 s)
  TSP2OptTest passed
    testTwoOptSwap passed (0.005 s)
    testTourDistance passed (0.0 s)
    testDistance passed (0.0 s)
```

Simulated Annealing algorithm



```
Projects
PSA_final_project
  Source Packages
    com.mycompany.tsp
      TSP2Opt.java
      TSPGeneticAlgorithm.java
      TSPRandomSwapping.java
      TSPSimulatedAnnealing.java
      TspChristofides.java
  Test Packages
    <default package>
      TSP2OptTest.java
      TSPGeneticAlgorithmTest.java
      TSPRandomSwappingTest.java
      TSPSimulatedAnnealingTest.java
      TspChristofidesTest.java
  Dependencies
  Java Dependencies
  Project Files
```

Output - Test (TSPSimulatedAnnealingTest) | TSPSimulatedAnnealingTest.java

```java
 * @author ruchi
 */
public class TSPSimulatedAnnealingTest {

public static final double DELTA = 0.001;
    private static ArrayList<TSPSimulatedAnnealing.City> cities;

  @BeforeClass
  public static void setUp() {
    cities = new ArrayList<>();
    cities.add(new City( id:"A",  latitude:0.0,  longitude:0.0));
    cities.add(new City( id:"B",  latitude:1.0,  longitude:0.0));
    cities.add(new City( id:"C",  latitude:0.0,  longitude:1.0));
    cities.add(new City( id:"D",  latitude:1.0,  longitude:1.0));
    cities.add(new City( id:"E",  latitude:2.0,  longitude:2.0));
    cities.add(new City( id:"F",  latitude:2.0,  longitude:0.0));
  }
```

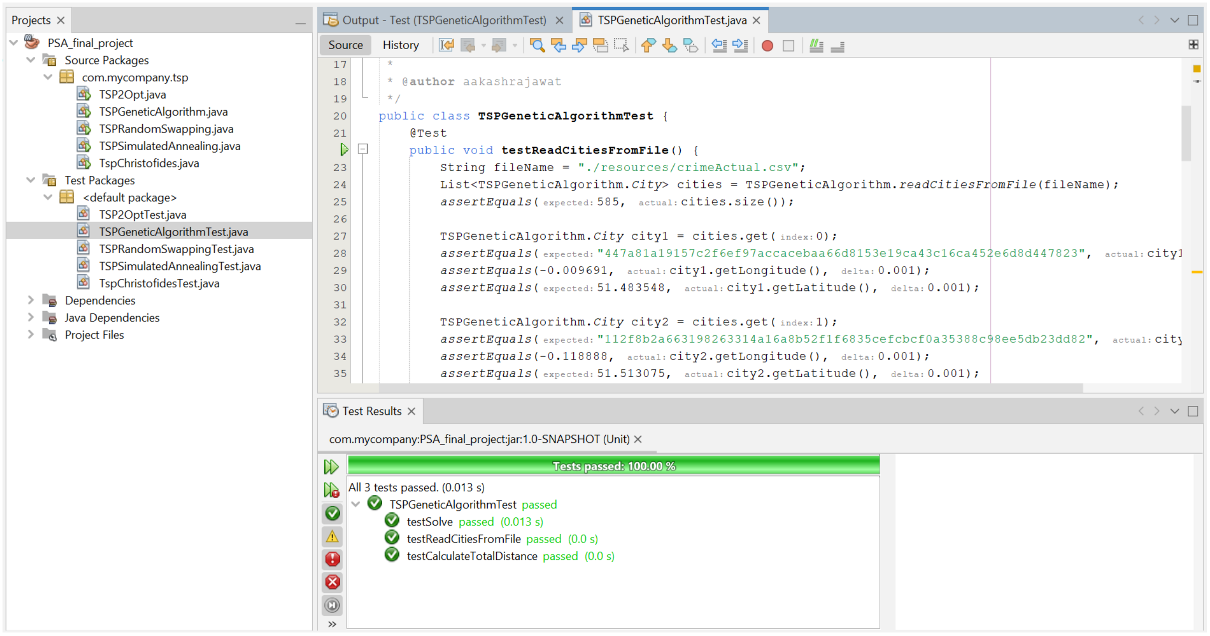
Test Results

com.mycompany:PSA_final_project:jar:1.0-SNAPSHOT (Unit)

```
Tests passed: 100.00 %
All 5 tests passed. (0.006 s)
  TSPSimulatedAnnealingTest passed
    testTour passed (0.005 s)
    testGetDistanceBetweenCities passed (0.0 s)
    testGetDistanceBetweenCities1 passed (0.0 s)
    testCalculateDistance passed (0.0 s)
    testGetRandomIndex passed (0.001 s)
```

Git Repository Browser   1:1   INS

Genetic algorithm

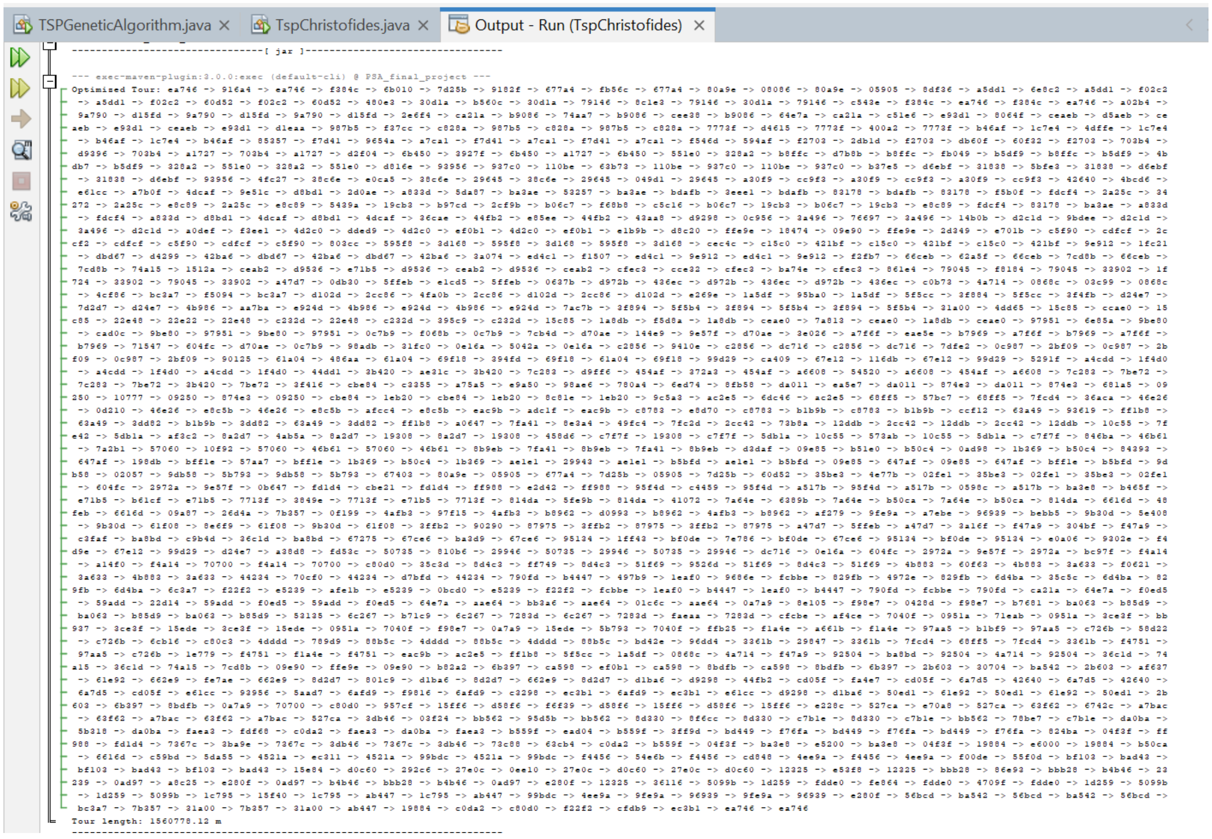


```java
 *
 * @author aakashrajawat
 */
public class TSPGeneticAlgorithmTest {
    @Test
    public void testReadCitiesFromFile() {
        String fileName = "./resources/crimeActual.csv";
        List<TSPGeneticAlgorithm.City> cities = TSPGeneticAlgorithm.readCitiesFromFile(fileName);
        assertEquals( expected: 585,  actual: cities.size());

        TSPGeneticAlgorithm.City city1 = cities.get( index: 0);
        assertEquals( expected: "447a81a19157c2f6ef97accacebaa66d8153e19ca43c16ca452e6d8d447823",  actual: city1
        assertEquals(-0.009691,  actual: city1.getLongitude(),  delta: 0.001);
        assertEquals( expected: 51.483548,  actual: city1.getLatitude(),  delta: 0.001);

        TSPGeneticAlgorithm.City city2 = cities.get( index: 1);
        assertEquals( expected: "112f8b2a663198263314a16a8b52f1f6835cefcbcf0a35388c98ee5db23dd82",  actual: city
        assertEquals(-0.118888,  actual: city2.getLongitude(),  delta: 0.001);
        assertEquals( expected: 51.513075,  actual: city2.getLatitude(),  delta: 0.001);
```

```
Tests passed: 100.00 %
All 3 tests passed. (0.013 s)
  TSPGeneticAlgorithmTest passed
    testSolve passed (0.013 s)
    testReadCitiesFromFile passed (0.0 s)
    testCalculateTotalDistance passed (0.0 s)
```

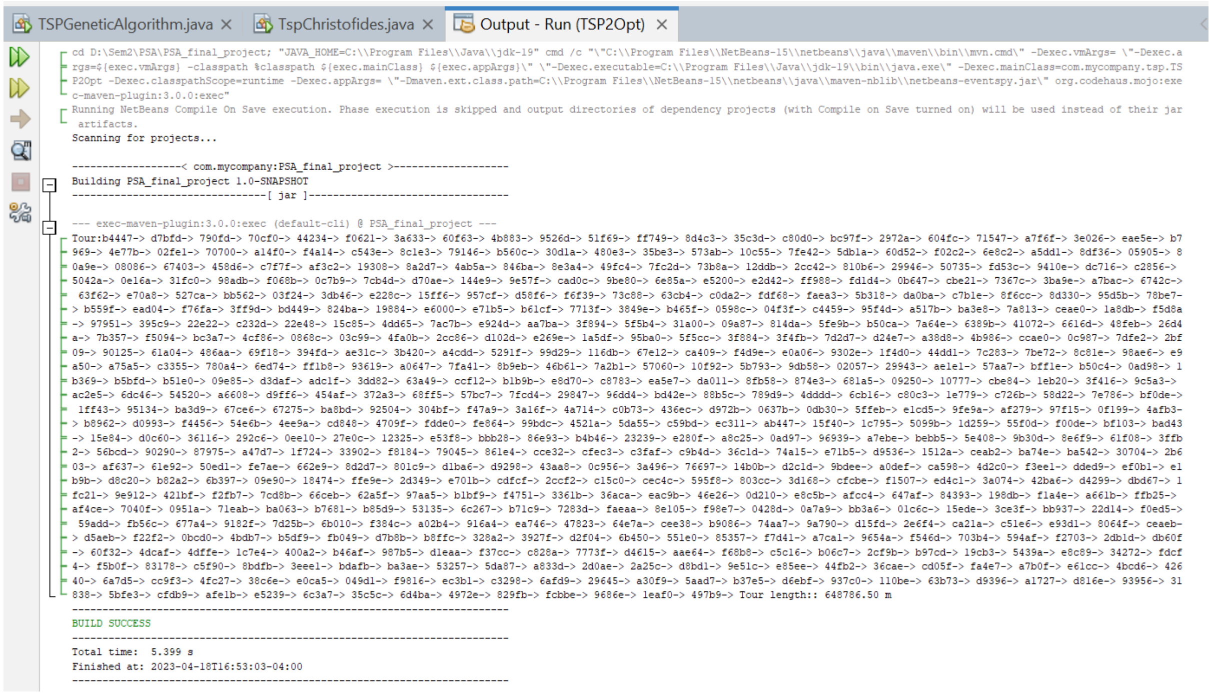
Output

Christofides



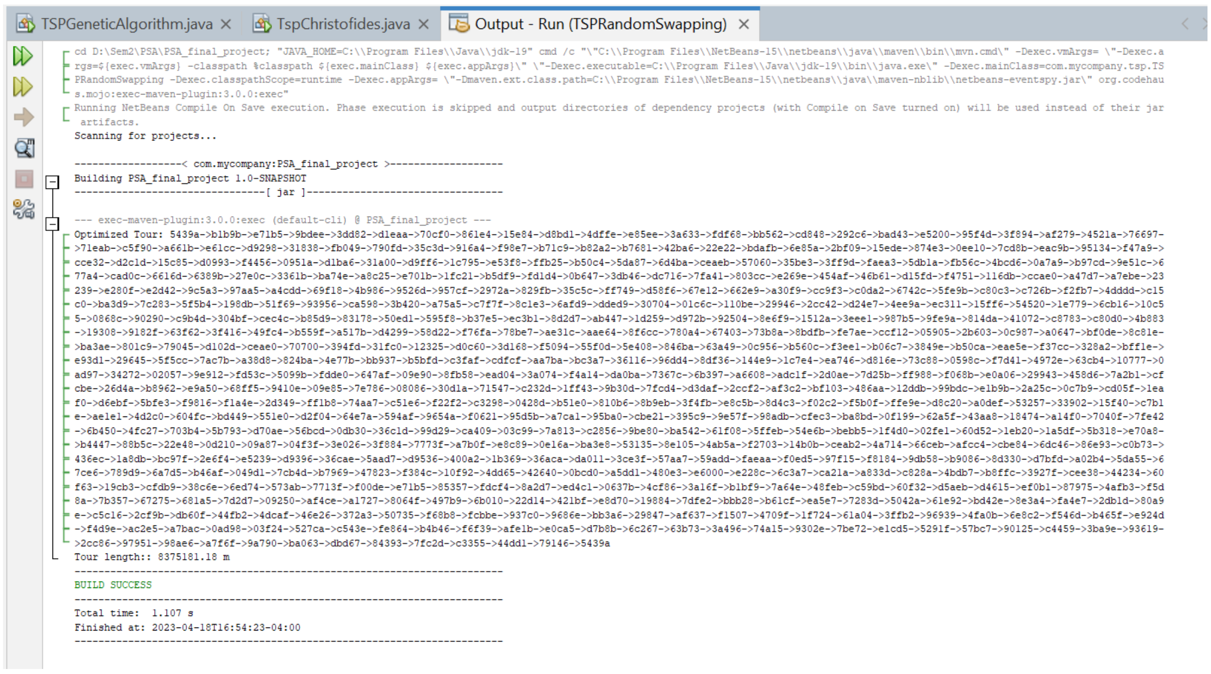
```
-------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSA_final_project ---
Optimized Tour: ea746 -> 916a4 -> ea746 -> f384c -> 6b010 -> 7d25b -> 9182f -> 677a4 -> fb56c -> 677a4 -> 80a9e -> 08086 -> 80a9e -> 05905 -> 8df36 -> a5dd1 -> 6e8c2 -> a5dd1 -> f02c2
 -> a5dd1 -> f02c2 -> 60d52 -> f02c2 -> 60d52 -> 480e3 -> 30d1a -> b560c -> 30d1a -> 79146 -> 8c1e3 -> 79146 -> 30d1a -> 79146 -> c543e -> f384c -> ea746 -> f384c -> ea746 -> a02b4 ->
 9a790 -> d15fd -> 9a790 -> d15fd -> 9a790 -> d15fd -> 2e6f4 -> ca21a -> b9086 -> 74aa7 -> b9086 -> cee38 -> b9086 -> 64e7a -> ca21a -> c51e6 -> e93d1 -> 8064f -> ceaeb -> d5aeb -> ce
aeb -> e93d1 -> ceaeb -> e93d1 -> d1eaa -> 987b5 -> f37cc -> c828a -> 987b5 -> c828a -> 987b5 -> c828a -> 7773f -> d4615 -> 7773f -> 400a2 -> 7773f -> b46af -> 1c7e4 -> 4dffe -> 1c7e4
 -> b46af -> 1c7e4 -> b46af -> 85357 -> f7d41 -> 9654a -> a7ca1 -> f7d41 -> a7ca1 -> f7d41 -> a7ca1 -> f546d -> 594af -> f2703 -> 2db1d -> f2703 -> db60f -> 60f32 -> f2703 -> 703b4 ->
 d9396 -> 703b4 -> a1727 -> 703b4 -> a1727 -> d2f04 -> 6b450 -> 3927f -> 6b450 -> a1727 -> 6b450 -> 551e0 -> 328a2 -> b8ffc -> d7b8b -> b8ffc -> fb049 -> b5df9 -> b8ffc -> b5df9 -> 4b
db7 -> b5df9 -> 328a2 -> 551e0 -> 328a2 -> 551e0 -> d816e -> 93956 -> 937c0 -> 110be -> 63b73 -> 110be -> 937c0 -> 110be -> 937c0 -> b37e5 -> d6ebf -> 31838 -> 5bfe3 -> 31838 -> d6ebf
 -> 31838 -> d6ebf -> 93956 -> 4fc27 -> 38c6e -> e0ca5 -> 38c6e -> 29645 -> 38c6e -> 29645 -> 049d1 -> 29645 -> a30f9 -> cc9f3 -> a30f9 -> cc9f3 -> a30f9 -> cc9f3 -> 42640 -> 4bcd6 ->
 e61cc -> a7b0f -> 4dcaf -> 9e51c -> d8bd1 -> 2d0ae -> a833d -> 5da87 -> ba3ae -> 53257 -> ba3ae -> bdafb -> 3eee1 -> bdafb -> 83178 -> bdafb -> 83178 -> f5b0f -> fdcf4 -> 2a25c -> 34
272 -> 2a25c -> e8c89 -> 2a25c -> e8c89 -> 5439a -> 19cb3 -> b97cd -> 2cf9b -> b06c7 -> f68b8 -> c5c16 -> b06c7 -> 19cb3 -> b06c7 -> 19cb3 -> e8c89 -> fdcf4 -> 83178 -> ba3ae -> a833d
 -> fdcf4 -> a833d -> d8bd1 -> 4dcaf -> d8bd1 -> 4dcaf -> 36cae -> 44fb2 -> e85ee -> 44fb2 -> 43aa8 -> d9298 -> 0c956 -> 3a496 -> 76697 -> 3a496 -> 14b0b -> d2c1d -> 9bdee -> d2c1d ->
 3a496 -> d2c1d -> a0def -> f3ee1 -> 4d2c0 -> dded9 -> 4d2c0 -> ef0b1 -> 4d2c0 -> ef0b1 -> e1b9b -> d8c20 -> ffe9e -> 18474 -> 09e90 -> ffe9e -> 2d349 -> e701b -> c5f90 -> cdfcf -> 2c
cf2 -> cdfcf -> c5f90 -> cdfcf -> c5f90 -> 803cc -> 595f8 -> 3d168 -> 595f8 -> 3d168 -> 595f8 -> 3d168 -> cec4c -> c15c0 -> 421bf -> c15c0 -> 421bf -> c15c0 -> 421bf -> 9e912 -> 1fc21
 -> dbd67 -> d4299 -> 42ba6 -> dbd67 -> 42ba6 -> dbd67 -> 42ba6 -> 3a074 -> ed4c1 -> f1507 -> ed4c1 -> 9e912 -> ed4c1 -> 9e912 -> f2fb7 -> 66ceb -> 62a5f -> 66ceb -> 7cd8b -> 66ceb ->
 7cd8b -> 74a15 -> 1512a -> ceab2 -> d9536 -> e71b5 -> d9536 -> ceab2 -> d9536 -> ceab2 -> cfec3 -> cce32 -> cfec3 -> ba74e -> cfec3 -> 861e4 -> 79045 -> f8184 -> 79045 -> 33902 -> 1f
724 -> 33902 -> 79045 -> 33902 -> a47d7 -> 0db30 -> 5ffeb -> e1cd5 -> 5ffeb -> 0637b -> d972b -> 436ec -> d972b -> 436ec -> d972b -> 436ec -> c0b73 -> 4a714 -> 0868c -> 03c99 -> 0868c
 -> 4cf86 -> bc3a7 -> f5094 -> bc3a7 -> d102d -> 2cc86 -> 4fa0b -> 2cc86 -> d102d -> 2cc86 -> d102d -> e269e -> 1a5df -> 95ba0 -> 1a5df -> 5f5cc -> 3f884 -> 5f5cc -> 3f4fb -> d24e7 ->
 7d2d7 -> d24e7 -> 4b986 -> aa7ba -> e924d -> 4b986 -> e924d -> 4b986 -> e924d -> 7ac7b -> 3f894 -> 5f5b4 -> 3f894 -> 5f5b4 -> 3f894 -> 5f5b4 -> 31a00 -> 4dd65 -> 15c85 -> ccae0 -> 15
c85 -> 22e48 -> 22e22 -> 22e48 -> c232d -> 22e48 -> c232d -> 395c9 -> c232d -> 15c85 -> 1a8db -> f5d8a -> 1a8db -> ceae0 -> 7a813 -> ceae0 -> 1a8db -> ceae0 -> 97951 -> 6e85a -> 9be80
 -> cad0c -> 9be80 -> 97951 -> 9be80 -> 97951 -> 0c7b9 -> f068b -> 0c7b9 -> 7cb4d -> d70ae -> 144e9 -> 9e57f -> d70ae -> 3e026 -> a7f6f -> eae5e -> b7969 -> a7f6f -> b7969 -> a7f6f ->
 b7969 -> 71547 -> 604fc -> d70ae -> 0c7b9 -> 98adb -> 31fc0 -> 0e16a -> 5042a -> 0e16a -> c2856 -> 9410e -> c2856 -> dc716 -> c2856 -> dc716 -> 7dfe2 -> 0c987 -> 2bf09 -> 0c987 -> 2b
f09 -> 0c987 -> 2bf09 -> 90125 -> 61a04 -> 486aa -> 61a04 -> 69f18 -> 394fd -> 69f18 -> 61a04 -> 69f18 -> 99d29 -> ca409 -> 67e12 -> 116db -> 67e12 -> 99d29 -> 5291f -> a4cdd -> 1f4d0
 -> a4cdd -> 1f4d0 -> a4cdd -> 1f4d0 -> 44dd1 -> 3b420 -> ae31c -> 3b420 -> 7c283 -> d9ff6 -> 454af -> 372a3 -> 454af -> a6608 -> 54520 -> a6608 -> 454af -> a6608 -> 7c283 -> 7be72 ->
 7c283 -> 7be72 -> 3b420 -> 7be72 -> 3f416 -> cbe84 -> c3355 -> a75a5 -> e9a50 -> 98ae6 -> 780a4 -> 6ed74 -> 8fb58 -> da011 -> ea5e7 -> da011 -> 874e3 -> da011 -> 874e3 -> 681a5 -> 09
250 -> 10777 -> 09250 -> 874e3 -> 09250 -> cbe84 -> 1eb20 -> cbe84 -> 1eb20 -> 8c81e -> 1eb20 -> 9c5a3 -> ac2e5 -> 6dc46 -> ac2e5 -> 68ff5 -> 57bc7 -> 68ff5 -> 7fcd4 -> 36aca -> 46e26
 -> 0d210 -> 46e26 -> e8c5b -> 46e26 -> e8c5b -> afcc4 -> e8c5b -> eac9b -> adc1f -> eac9b -> c8783 -> e8d70 -> c8783 -> b1b9b -> c8783 -> b1b9b -> ccf12 -> 63a49 -> 93619 -> ff1b8 ->
 63a49 -> 3dd82 -> b1b9b -> 3dd82 -> 63a49 -> 3dd82 -> ff1b8 -> a0647 -> 7fa41 -> 8e3a4 -> 49fc4 -> 7fc2d -> 2cc42 -> 73b8a -> 12ddb -> 2cc42 -> 12ddb -> 2cc42 -> 12ddb -> 10c55 -> 7f
e42 -> 5db1a -> af3c2 -> 8a2d7 -> 4ab5a -> 8a2d7 -> 19308 -> 8a2d7 -> 19308 -> 458d6 -> c7f7f -> 19308 -> c7f7f -> 5db1a -> 10c55 -> 573ab -> 10c55 -> 5db1a -> c7f7f -> 846ba -> 46b61
 -> 7a2b1 -> 57060 -> 10f92 -> 57060 -> 46b61 -> 57060 -> 46b61 -> 8b9eb -> 7fa41 -> 8b9eb -> 7fa41 -> 8b9eb -> d3daf -> 09e85 -> b51e0 -> b50c4 -> 0ad98 -> 1b369 -> b50c4 -> 84393 ->
 647af -> 198db -> bff1e -> 57aa7 -> bff1e -> 1b369 -> b50c4 -> 1b369 -> ae1e1 -> 29943 -> ae1e1 -> b5bfd -> ae1e1 -> b5bfd -> 09e85 -> 647af -> 09e85 -> 647af -> bff1e -> b5bfd -> 9d
b58 -> 02057 -> 9db58 -> 5b793 -> 9db58 -> 5b793 -> 67403 -> 80a9e -> 05905 -> 677a4 -> 7d25b -> 05905 -> 7d25b -> 60d52 -> 35be3 -> 4e77b -> 02fe1 -> 35be3 -> 02fe1 -> 35be3 -> 02fe1
 -> 604fc -> 2972a -> 9e57f -> 0b647 -> fd1d4 -> cbe21 -> fd1d4 -> ff988 -> e2d42 -> ff988 -> 95f4d -> c4459 -> 95f4d -> a517b -> 95f4d -> a517b -> 0598c -> a517b -> ba3e8 -> b465f ->
 e71b5 -> b61cf -> e71b5 -> 7713f -> 3849e -> 7713f -> e71b5 -> 7713f -> 814da -> 5fe9b -> 814da -> 41072 -> 7a64e -> 6389b -> 7a64e -> b50ca -> 7a64e -> b50ca -> 814da -> 6616d -> 48
feb -> 6616d -> 09a87 -> 26d4a -> 7b357 -> 0f199 -> 4afb3 -> 97f15 -> 4afb3 -> b8962 -> d0993 -> b8962 -> 4afb3 -> b8962 -> af279 -> 9fe9a -> a7ebe -> 96939 -> bebb5 -> 9b30d -> 5e408
 -> 9b30d -> 61f08 -> 8e6f9 -> 61f08 -> 9b30d -> 61f08 -> 3ffb2 -> 90290 -> 87975 -> 3ffb2 -> 87975 -> 3ffb2 -> 87975 -> a47d7 -> 5ffeb -> a47d7 -> 3a16f -> f47a9 -> 304bf -> f47a9 ->
 c3faf -> ba8bd -> c9b4d -> 36c1d -> ba8bd -> 67275 -> 67ce6 -> ba3d9 -> 67ce6 -> 95134 -> 1ff43 -> bf0de -> 7e786 -> bf0de -> 67ce6 -> 95134 -> bf0de -> 95134 -> e0a06 -> 9302e -> f4
d9e -> 67e12 -> 99d29 -> d24e7 -> a38d8 -> fd53c -> 50735 -> 810b6 -> 29946 -> 50735 -> 29946 -> 50735 -> 29946 -> dc716 -> 0e16a -> 604fc -> 2972a -> 9e57f -> 2972a -> bc97f -> f4a14
 -> a14f0 -> f4a14 -> 70700 -> f4a14 -> 70700 -> c80d0 -> 35c3d -> 8d4c3 -> ff749 -> 8d4c3 -> 51f69 -> 9526d -> 51f69 -> 8d4c3 -> 51f69 -> 4b883 -> 60f63 -> 4b883 -> 3a633 -> f0621 ->
 3a633 -> 4b883 -> 3a633 -> 44234 -> 70cf0 -> 44234 -> d7bfd -> 44234 -> 790fd -> b4447 -> 497b9 -> 1eaf0 -> 9686e -> fcbbe -> 829fb -> 4972e -> 829fb -> 6d4ba -> 35c5c -> 6d4ba -> 82
9fb -> 6d4ba -> 6c3a7 -> f22f2 -> e5239 -> afe1b -> e5239 -> 0bcd0 -> e5239 -> f22f2 -> fcbbe -> 1eaf0 -> b4447 -> 1eaf0 -> b4447 -> 790fd -> fcbbe -> 790fd -> ca21a -> 64e7a -> f0ed5
 -> 59add -> 22d14 -> 59add -> f0ed5 -> 59add -> f0ed5 -> 64e7a -> aae64 -> bb3a6 -> aae64 -> 01c6c -> aae64 -> 0a7a9 -> 8e105 -> f98e7 -> 0428d -> f98e7 -> b7681 -> ba063 -> b85d9 ->
 ba063 -> b85d9 -> ba063 -> b85d9 -> 53135 -> 6c267 -> b71c9 -> 6c267 -> 7283d -> 6c267 -> 7283d -> faeaa -> 7283d -> cfcbe -> af4ce -> 7040f -> 0951a -> 71eab -> 0951a -> 3ce3f -> bb
937 -> 3ce3f -> 15ede -> 3ce3f -> 15ede -> 0951a -> 7040f -> f98e7 -> 0a7a9 -> 15ede -> 5b793 -> 7040f -> ffb25 -> f1a4e -> a661b -> f1a4e -> 97aa5 -> b1bf9 -> 97aa5 -> c726b -> 58d22
 -> c726b -> 6cb16 -> c80c3 -> 4dddd -> 789d9 -> 88b5c -> 4dddd -> 88b5c -> 4dddd -> 88b5c -> bd42e -> 96dd4 -> 3361b -> 29847 -> 3361b -> 7fcd4 -> 68ff5 -> 7fcd4 -> 3361b -> f4751 ->
 97aa5 -> c726b -> 1e779 -> f4751 -> f1a4e -> f4751 -> eac9b -> ac2e5 -> ff1b8 -> 5f5cc -> 1a5df -> 0868c -> 4a714 -> f47a9 -> 92504 -> ba8bd -> 92504 -> 4a714 -> 92504 -> 36c1d -> 74
a15 -> 36c1d -> 74a15 -> 7cd8b -> 09e90 -> ffe9e -> 09e90 -> b82a2 -> 6b397 -> ca598 -> ef0b1 -> ca598 -> 8bdfb -> ca598 -> 8bdfb -> 6b397 -> 2b603 -> 30704 -> ba542 -> 2b603 -> af637
 -> 61e92 -> 662e9 -> fe7ae -> 662e9 -> 8d2d7 -> 801c9 -> d1ba6 -> 8d2d7 -> 662e9 -> 8d2d7 -> d1ba6 -> d9298 -> 44fb2 -> cd05f -> fa4e7 -> cd05f -> 6a7d5 -> 42640 -> 6a7d5 -> 42640 ->
 6a7d5 -> cd05f -> e61cc -> 93956 -> 5aad7 -> 6afd9 -> f9816 -> 6afd9 -> c3298 -> ec3b1 -> 6afd9 -> ec3b1 -> e61cc -> d9298 -> d1ba6 -> 50ed1 -> 61e92 -> 50ed1 -> 61e92 -> 50ed1 -> 2b
603 -> 6b397 -> 8bdfb -> 0a7a9 -> 70700 -> c80d0 -> 957cf -> 15ff6 -> d58f6 -> f6f39 -> d58f6 -> 15ff6 -> d58f6 -> 15ff6 -> e228c -> 527ca -> e70a8 -> 527ca -> 63f62 -> 6742c -> a7bac
 -> 63f62 -> a7bac -> 63f62 -> a7bac -> 527ca -> 3db46 -> 03f24 -> bb562 -> 95d5b -> bb562 -> 8d330 -> 8f6cc -> 8d330 -> c7b1e -> 8d330 -> c7b1e -> bb562 -> 78be7 -> c7b1e -> da0ba ->
 5b318 -> da0ba -> faea3 -> fdf68 -> c0da2 -> faea3 -> da0ba -> faea3 -> b559f -> ead04 -> b559f -> 3ff9d -> bd449 -> f76fa -> bd449 -> f76fa -> bd449 -> f76fa -> 824ba -> 04f3f -> ff
988 -> fd1d4 -> 7367c -> 3ba9e -> 7367c -> 3db46 -> 7367c -> 3db46 -> 73c88 -> 63cb4 -> c0da2 -> b559f -> 04f3f -> ba3e8 -> e5200 -> ba3e8 -> 04f3f -> 19884 -> e6000 -> 19884 -> b50ca
 -> 6616d -> c59bd -> 5da55 -> 4521a -> ec311 -> 4521a -> 99bdc -> 4521a -> 99bdc -> f4456 -> 54e6b -> f4456 -> cd848 -> 4ee9a -> f4456 -> 4ee9a -> f00de -> 55f0d -> bf103 -> bad43 ->
 bf103 -> bad43 -> bf103 -> bad43 -> 15e84 -> d0c60 -> 292c6 -> 27e0c -> 0ee10 -> 27e0c -> d0c60 -> 27e0c -> d0c60 -> 12325 -> e53f8 -> 12325 -> bbb28 -> 86e93 -> bbb28 -> b4b46 -> 23
239 -> 0ad97 -> a8c25 -> e280f -> 0ad97 -> b4b46 -> bbb28 -> b4b46 -> 0ad97 -> e280f -> 12325 -> 36116 -> 5099b -> 1d259 -> fdde0 -> fe864 -> fdde0 -> 4709f -> fdde0 -> 1d259 -> 5099b
 -> 1d259 -> 5099b -> 1c795 -> 15f40 -> 1c795 -> ab447 -> 1c795 -> ab447 -> 99bdc -> 4ee9a -> 9fe9a -> 96939 -> 9fe9a -> 96939 -> e280f -> 56bcd -> ba542 -> 56bcd -> ba542 -> 56bcd ->
 bc3a7 -> 7b357 -> 31a00 -> 7b357 -> 31a00 -> ab447 -> 19884 -> c0da2 -> c80d0 -> f22f2 -> cfdb9 -> ec3b1 -> ea746 -> ea746
Tour length: 1560778.12 m
------------------------------------------------------------------------
```

2 opt


TSPGeneticAlgorithm.java × TspChristofides.java × Output - Run (TSP2Opt) ×

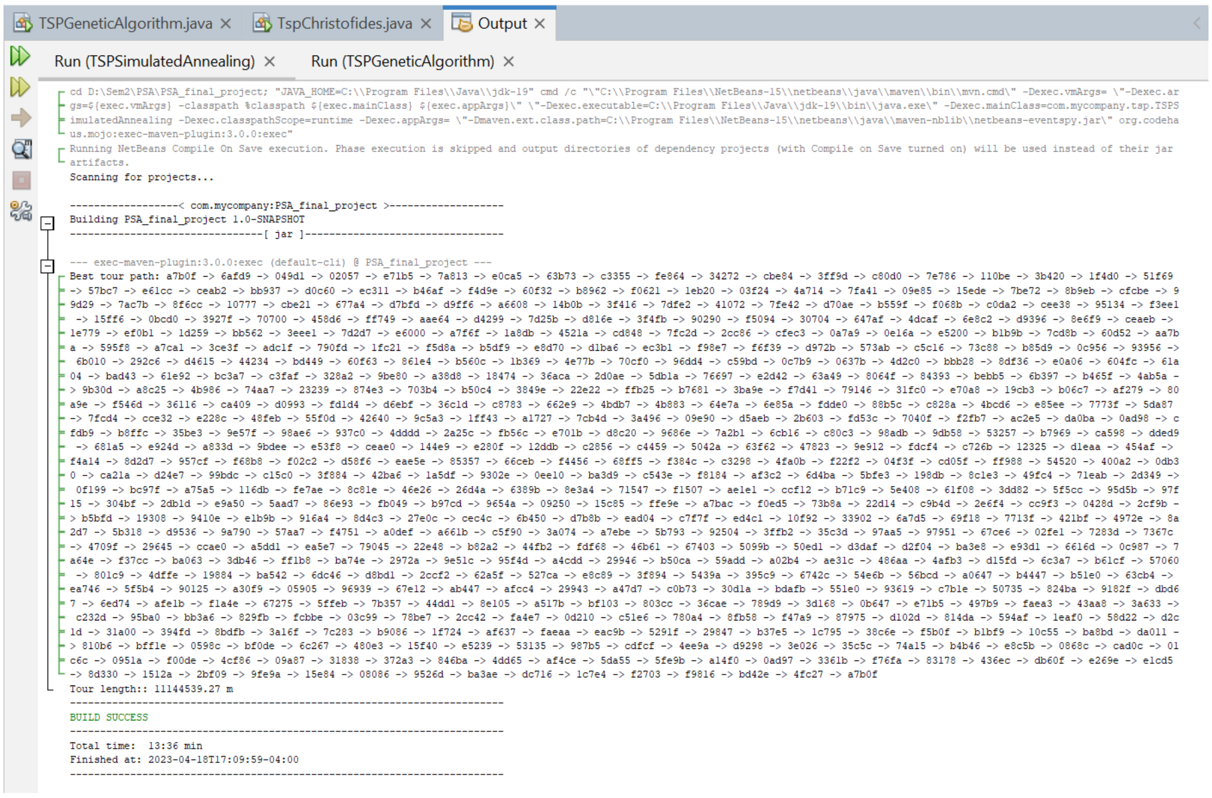
```
cd D:\Sem2\PSA\PSA_final_project; "JAVA_HOME=C:\\Program Files\\Java\\jdk-19" cmd /c "\"C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven\\bin\\mvn.cmd\" -Dexec.vmArgs= \"-Dexec.a
rgs=${exec.vmArgs} -classpath %classpath ${exec.mainClass} ${exec.appArgs}\" \"-Dexec.executable=C:\\Program Files\\Java\\jdk-19\\bin\\java.exe\" -Dexec.mainClass=com.mycompany.tsp.TS
P2Opt -Dexec.classpathScope=runtime -Dexec.appArgs= \"-Dmaven.ext.class.path=C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven-nblib\\netbeans-eventspy.jar\" org.codehaus.mojo:exe
c-maven-plugin:3.0.0:exec"
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects (with Compile on Save turned on) will be used instead of their jar
 artifacts.
Scanning for projects...

------------------< com.mycompany:PSA_final_project >-------------------
Building PSA_final_project 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSA_final_project ---
Tour:b4447-> d7bfd-> 790fd-> 70cf0-> 44234-> f0621-> 3a633-> 60f63-> 4b883-> 9526d-> 51f69-> ff749-> 8d4c3-> 35c3d-> c80d0-> bc97f-> 2972a-> 604fc-> 71547-> a7f6f-> 3e026-> eae5e-> b7
969-> 4e77b-> 02fe1-> 70700-> a14f0-> f4a14-> c543e-> 8c1e3-> 79146-> b560c-> 30d1a-> 480e3-> 35be3-> 573ab-> 10c55-> 7fe42-> 5db1a-> 60d52-> f02c2-> 6e8c2-> a5dd1-> 8df36-> 05905-> 8
0a9e-> 08086-> 67403-> 458d6-> c7f7f-> af3c2-> 19308-> 8a2d7-> 4ab5a-> 846ba-> 8e3a4-> 49fc4-> 7fc2d-> 73b8a-> 12ddb-> 2cc42-> 810b6-> 29946-> 50735-> fd53c-> 9410e-> dc716-> c2856->
5042a-> 0e16a-> 31fc0-> 98adb-> f068b-> 0c7b9-> 7cb4d-> d70ae-> 144e9-> 9e57f-> cad0c-> 9be80-> 6e85a-> e5200-> e2d42-> ff988-> fd1d4-> 0b647-> cbe21-> 7367c-> 3ba9e-> a7bac-> 6742c->
 63f62-> e70a8-> 527ca-> bb562-> 03f24-> 3db46-> e228c-> 15ff6-> 957cf-> d58f6-> f6f39-> 73c88-> 63cb4-> c0da2-> fdf68-> faea3-> 5b318-> da0ba-> c7b1e-> 8f6cc-> 8d330-> 95d5b-> 78be7-
> b559f-> ead04-> f76fa-> 3ff9d-> bd449-> 824ba-> 19884-> e6000-> e71b5-> b61cf-> 7713f-> 3849e-> b465f-> 0598c-> 04f3f-> c4459-> 95f4d-> a517b-> ba3e8-> 7a813-> ceae0-> 1a8db-> f5d8a
-> 97951-> 395c9-> 22e22-> c232d-> 22e48-> 15c85-> 4dd65-> 7ac7b-> e924d-> aa7ba-> 3f894-> 5f5b4-> 31a00-> 09a87-> 814da-> 5fe9b-> b50ca-> 7a64e-> 6389b-> 41072-> 6616d-> 48feb-> 26d4
a-> 7b357-> f5094-> bc3a7-> 4cf86-> 0868c-> 03c99-> 4fa0b-> 2cc86-> d102d-> e269e-> 1a5df-> 95ba0-> 5f5cc-> 3f884-> 3f4fb-> 7d2d7-> d24e7-> a38d8-> 4b986-> ccae0-> 0c987-> 7dfe2-> 2bf
09-> 90125-> 61a04-> 486aa-> 69f18-> 394fd-> ae31c-> 3b420-> a4cdd-> 5291f-> 99d29-> 116db-> 67e12-> ca409-> f4d9e-> e0a06-> 9302e-> 1f4d0-> 44dd1-> 7c283-> 7be72-> 8c81e-> 98ae6-> e9
a50-> a75a5-> c3355-> 780a4-> 6ed74-> ff1b8-> 93619-> a0647-> 7fa41-> 8b9eb-> 46b61-> 7a2b1-> 57060-> 10f92-> 5b793-> 9db58-> 02057-> 29943-> ae1e1-> 57aa7-> bff1e-> b50c4-> 0ad98-> 1
b369-> b5bfd-> b51e0-> 09e85-> d3daf-> adc1f-> 3dd82-> 63a49-> ccf12-> b1b9b-> e8d70-> c8783-> ea5e7-> da011-> 8fb58-> 874e3-> 681a5-> 09250-> 10777-> cbe84-> 1eb20-> 3f416-> 9c5a3->
ac2e5-> 6dc46-> 54520-> a6608-> d9ff6-> 454af-> 372a3-> 68ff5-> 57bc7-> 7fcd4-> 29847-> 96dd4-> bd42e-> 88b5c-> 789d9-> 4dddd-> 6cb16-> c80c3-> 1e779-> c726b-> 58d22-> 7e786-> bf0de->
 1ff43-> 95134-> ba3d9-> 67ce6-> 67275-> ba8bd-> 92504-> 304bf-> f47a9-> 3a16f-> 4a714-> c0b73-> 436ec-> d972b-> 0637b-> 0db30-> 5ffeb-> e1cd5-> 9fe9a-> af279-> 97f15-> 0f199-> 4afb3-
> b8962-> d0993-> f4456-> 54e6b-> 4ee9a-> cd848-> 4709f-> fdde0-> fe864-> 99bdc-> 4521a-> 5da55-> c59bd-> ec311-> ab447-> 15f40-> 1c795-> 5099b-> 1d259-> 55f0d-> f00de-> bf103-> bad43
-> 15e84-> d0c60-> 36116-> 292c6-> 0ee10-> 27e0c-> 12325-> e53f8-> bbb28-> 86e93-> b4b46-> 23239-> e280f-> a8c25-> 0ad97-> 96939-> a7ebe-> bebb5-> 5e408-> 9b30d-> 8e6f9-> 61f08-> 3ffb
2-> 56bcd-> 90290-> 87975-> a47d7-> 1f724-> 33902-> f8184-> 79045-> 861e4-> cce32-> cfec3-> c3faf-> c9b4d-> 36c1d-> 74a15-> e71b5-> d9536-> 1512a-> ceab2-> ba74e-> ba542-> 30704-> 2b6
03-> af637-> 61e92-> 50ed1-> fe7ae-> 662e9-> 8d2d7-> 801c9-> d1ba6-> d9298-> 43aa8-> 0c956-> 3a496-> 76697-> 14b0b-> d2c1d-> 9bdee-> a0def-> ca598-> 4d2c0-> f3ee1-> dded9-> ef0b1-> e1
b9b-> d8c20-> b82a2-> 6b397-> 09e90-> 18474-> ffe9e-> 2d349-> e701b-> cdfcf-> 2ccf2-> c15c0-> cec4c-> 595f8-> 803cc-> 3d168-> cfcbe-> f1507-> ed4c1-> 3a074-> 42ba6-> d4299-> dbd67-> 1
fc21-> 9e912-> 421bf-> f2fb7-> 7cd8b-> 66ceb-> 62a5f-> 97aa5-> b1bf9-> f4751-> 3361b-> 36aca-> eac9b-> 46e26-> 0d210-> e8c5b-> afcc4-> 647af-> 84393-> 198db-> f1a4e-> a661b-> ffb25->
af4ce-> 7040f-> 0951a-> 71eab-> ba063-> b7681-> b85d9-> 53135-> 6c267-> b71c9-> 7283d-> faeaa-> 8e105-> f98e7-> 0428d-> 0a7a9-> bb3a6-> 01c6c-> 15ede-> 3ce3f-> bb937-> 22d14-> f0ed5->
 59add-> fb56c-> 677a4-> 9182f-> 7d25b-> 6b010-> f384c-> a02b4-> 916a4-> ea746-> 47823-> 64e7a-> cee38-> b9086-> 74aa7-> 9a790-> d15fd-> 2e6f4-> ca21a-> c51e6-> e93d1-> 8064f-> ceaeb-
> d5aeb-> f22f2-> 0bcd0-> 4bdb7-> b5df9-> fb049-> d7b8b-> b8ffc-> 328a2-> 3927f-> d2f04-> 6b450-> 551e0-> 85357-> f7d41-> a7ca1-> 9654a-> f546d-> 703b4-> 594af-> f2703-> 2db1d-> db60f
-> 60f32-> 4dcaf-> 4dffe-> 1c7e4-> 400a2-> b46af-> 987b5-> d1eaa-> f37cc-> c828a-> 7773f-> d4615-> aae64-> f68b8-> c5c16-> b06c7-> 2cf9b-> b97cd-> 19cb3-> 5439a-> e8c89-> 34272-> fdcf
4-> f5b0f-> 83178-> c5f90-> 8bdfb-> 3eee1-> bdafb-> ba3ae-> 53257-> 5da87-> a833d-> 2d0ae-> 2a25c-> d8bd1-> 9e51c-> e85ee-> 44fb2-> 36cae-> cd05f-> fa4e7-> a7b0f-> e61cc-> 4bcd6-> 426
40-> 6a7d5-> cc9f3-> 4fc27-> 38c6e-> e0ca5-> 049d1-> f9816-> ec3b1-> c3298-> 6afd9-> 29645-> a30f9-> 5aad7-> b37e5-> d6ebf-> 937c0-> 110be-> 63b73-> d9396-> a1727-> d816e-> 93956-> 31
838-> 5bfe3-> cfdb9-> afe1b-> e5239-> 6c3a7-> 35c5c-> 6d4ba-> 4972e-> 829fb-> fcbbe-> 9686e-> 1eaf0-> 497b9-> Tour length:: 648786.50 m
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  5.399 s
Finished at: 2023-04-18T16:53:03-04:00
------------------------------------------------------------------------
```

Random Swapping


TSPGeneticAlgorithm.java × TspChristofides.java × Output - Run (TSPRandomSwapping) ×

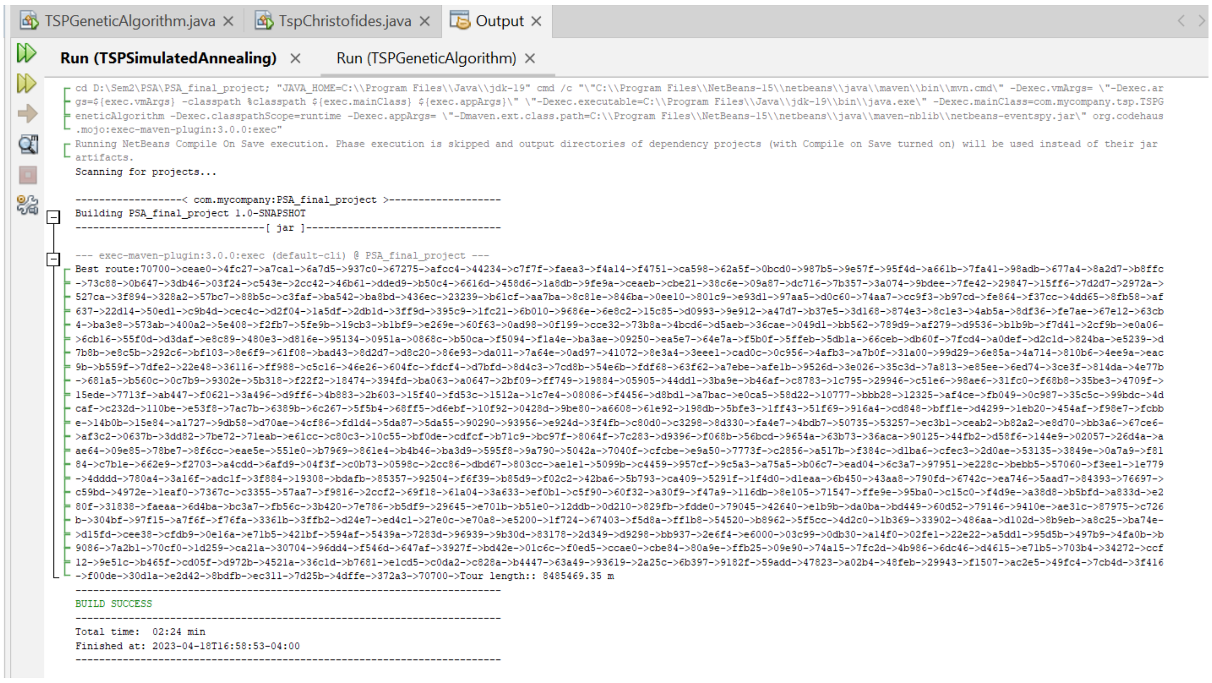
```
cd D:\Sem2\PSA\PSA_final_project; "JAVA_HOME=C:\\Program Files\\Java\\jdk-19" cmd /c "\"C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven\\bin\\mvn.cmd\" -Dexec.vmArgs= \"-Dexec.a
rgs=${exec.vmArgs} -classpath %classpath ${exec.mainClass} ${exec.appArgs}\" \"-Dexec.executable=C:\\Program Files\\Java\\jdk-19\\bin\\java.exe\" -Dexec.mainClass=com.mycompany.tsp.TS
PRandomSwapping -Dexec.classpathScope=runtime -Dexec.appArgs= \"-Dmaven.ext.class.path=C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven-nblib\\netbeans-eventspy.jar\" org.codehau
s.mojo:exec-maven-plugin:3.0.0:exec"
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects (with Compile on Save turned on) will be used instead of their jar
 artifacts.
Scanning for projects...

------------------< com.mycompany:PSA_final_project >-------------------
Building PSA_final_project 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSA_final_project ---
Optimized Tour: 5439a->b1b9b->e71b5->9bdee->3dd82->d1eaa->70cf0->861e4->15e84->d8bd1->4dffe->e85ee->3a633->fdf68->bb562->cd848->292c6->bad43->e5200->95f4d->3f894->af279->4521a->76697-
>71eab->c5f90->a661b->e61cc->d9298->31838->fb049->790fd->35c3d->916a4->f98e7->b71c9->b82a2->b7681->42ba6->22e22->bdafb->6e85a->2bf09->15ede->874e3->0ee10->7cd8b->eac9b->95134->f47a9->
cce32->d2c1d->15c85->d0993->f4456->0951a->d1ba6->31a00->d9ff6->1c795->e53f8->ffb25->b50c4->5da87->6d4ba->ceaeb->57060->35be3->3ff9d->faea3->5db1a->fb56c->4bcd6->0a7a9->b97cd->9e51c->6
77a4->cad0c->6616d->6389b->27e0c->3361b->ba74e->a8c25->e701b->1fc21->b5df9->fd1d4->0b647->3db46->dc716->7fa41->803cc->e269e->454af->46b61->d15fd->f4751->116db->ccae0->a47d7->a7ebe->23
239->e280f->e2d42->9c5a3->97aa5->a4cdd->69f18->4b986->9526d->957cf->2972a->829fb->35c5c->ff749->d58f6->67e12->662e9->a30f9->cc9f3->c0da2->6742c->5fe9b->c80c3->c726b->f2fb7->4dddd->c15
c0->ba3d9->7c283->5f5b4->198db->51f69->93956->ca598->3b420->a75a5->c7f7f->8c1e3->6afd9->dded9->30704->01c6c->110be->29946->2cc42->d24e7->4ee9a->ec311->15ff6->54520->1e779->6cb16->10c5
5->0868c->90290->c9b4d->304bf->cec4c->b85d9->83178->50ed1->595f8->b37e5->ec3b1->8d2d7->ab447->1d259->d972b->92504->8e6f9->1512a->3eee1->987b5->9fe9a->814da->41072->c8783->c80d0->4b883
->19308->9182f->63f62->3f416->49fc4->b559f->a517b->d4299->58d22->f76fa->78be7->ae31c->aae64->8f6cc->780a4->67403->73b8a->8bdfb->fe7ae->ccf12->05905->2b603->0c987->a0647->bf0de->8c81e-
>ba3ae->801c9->79045->d102d->ceae0->70700->394fd->31fc0->12325->d0c60->3d168->f5094->55f0d->5e408->846ba->63a49->0c956->b560c->f3ee1->b06c7->3849e->b50ca->eae5e->f37cc->328a2->bff1e->
e93d1->29645->5f5cc->7ac7b->a38d8->824ba->4e77b->bb937->b5bfd->c3faf->cdfcf->aa7ba->bc3a7->36116->96dd4->8df36->144e9->1c7e4->ea746->d816e->73c88->0598c->f7d41->4972e->63cb4->10777->0
ad97->34272->02057->9e912->fd53c->5099b->fdde0->647af->09e90->8fb58->ead04->3a074->f4a14->da0ba->7367c->6b397->a6608->adc1f->2d0ae->7d25b->ff988->f068b->e0a06->29943->458d6->7a2b1->cf
cbe->26d4a->b8962->e9a50->68ff5->9410e->09e85->7e786->08086->30d1a->71547->c232d->1ff43->9b30d->7fcd4->d3daf->2ccf2->af3c2->bf103->486aa->12ddb->99bdc->e1b9b->2a25c->0c7b9->cd05f->1ea
f0->d6ebf->5bfe3->f9816->f1a4e->2d349->ff1b8->74aa7->c51e6->f22f2->c3298->0428d->b51e0->810b6->8b9eb->3f4fb->e8c5b->8d4c3->f02c2->f5b0f->ffe9e->d8c20->a0def->53257->33902->15f40->c7b1
e->ae1e1->4d2c0->604fc->bd449->551e0->d2f04->64e7a->594af->9654a->f0621->95d5b->a7ca1->95ba0->cbe21->395c9->9e57f->98adb->cfec3->ba8bd->0f199->62a5f->43aa8->18474->a14f0->7040f->7fe42
->6b450->4fc27->703b4->5b793->d70ae->56bcd->0db30->36c1d->99d29->ca409->03c99->7a813->c2856->9be80->ba542->61f08->5ffeb->54e6b->bebb5->1f4d0->02fe1->60d52->1eb20->1a5df->5b318->e70a8-
>b4447->88b5c->22e48->0d210->09a87->04f3f->3e026->3f884->7773f->a7b0f->e8c89->0e16a->ba3e8->53135->8e105->4ab5a->f2703->14b0b->ceab2->4a714->66ceb->afcc4->cbe84->6dc46->86e93->c0b73->
436ec->1a8db->bc97f->2e6f4->e5239->d9396->36cae->5aad7->d9536->400a2->1b369->36aca->da011->3ce3f->57aa7->59add->faeaa->f0ed5->97f15->f8184->9db58->b9086->8d330->d7bfd->a02b4->5da55->6
7ce6->789d9->6a7d5->b46af->049d1->7cb4d->b7969->47823->f384c->10f92->4dd65->42640->0bcd0->a5dd1->480e3->e6000->e228c->6c3a7->ca21a->a833d->c828a->4bdb7->b8ffc->3927f->cee38->44234->60
f63->19cb3->cfdb9->38c6e->6ed74->573ab->7713f->f00de->e71b5->85357->fdcf4->8a2d7->ed4c1->0637b->4cf86->3a16f->b1bf9->7a64e->48feb->c59bd->60f32->d5aeb->d4615->ef0b1->87975->4afb3->f5d
8a->7b357->67275->681a5->7d2d7->09250->af4ce->a1727->8064f->497b9->6b010->22d14->421bf->e8d70->19884->7dfe2->bbb28->b61cf->ea5e7->7283d->5042a->61e92->bd42e->8e3a4->fa4e7->2db1d->80a9
e->c5c16->2cf9b->db60f->44fb2->4dcaf->46e26->372a3->50735->f68b8->fcbbe->937c0->9686e->bb3a6->29847->af637->f1507->4709f->1f724->61a04->3ffb2->96939->4fa0b->6e8c2->f546d->b465f->e924d
->f4d9e->ac2e5->a7bac->0ad98->03f24->527ca->c543e->fe864->b4b46->f6f39->afe1b->e0ca5->d7b8b->6c267->63b73->3a496->74a15->9302e->7be72->e1cd5->5291f->57bc7->90125->c4459->3ba9e->93619-
>2cc86->97951->98ae6->a7f6f->9a790->ba063->dbd67->84393->7fc2d->c3355->44dd1->79146->5439a
Tour length:: 8375181.18 m
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.107 s
Finished at: 2023-04-18T16:54:23-04:00
------------------------------------------------------------------------
```

Simulated Annealing



```
cd D:\Sem2\PSA\PSA_final_project; "JAVA_HOME=C:\\Program Files\\Java\\jdk-19" cmd /c "\"C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven\\bin\\mvn.cmd\" -Dexec.vmArgs= \"-Dexec.ar
gs=${exec.vmArgs} -classpath %classpath ${exec.mainClass} ${exec.appArgs}\" \"-Dexec.executable=C:\\Program Files\\Java\\jdk-19\\bin\\java.exe\" -Dexec.mainClass=com.mycompany.tsp.TSPS
imulatedAnnealing -Dexec.classpathScope=runtime -Dexec.appArgs= \"-Dmaven.ext.class.path=C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven-nblib\\netbeans-eventspy.jar\" org.codeha
us.mojo:exec-maven-plugin:3.0.0:exec"
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects (with Compile on Save turned on) will be used instead of their jar
artifacts.
Scanning for projects...

------------------< com.mycompany:PSA_final_project >-------------------
Building PSA_final_project 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSA_final_project ---
Best tour path: a7b0f -> 6afd9 -> 049d1 -> 02057 -> e71b5 -> 7a813 -> e0ca5 -> 63b73 -> c3355 -> fe864 -> 34272 -> cbe84 -> 3ff9d -> c80d0 -> 7e786 -> 110be -> 3b420 -> 1f4d0 -> 51f69
-> 57bc7 -> e61cc -> ceab2 -> bb937 -> d0c60 -> ec311 -> b46af -> f4d9e -> 60f32 -> b8962 -> f0621 -> 1eb20 -> 03f24 -> 4a714 -> 7fa41 -> 09e85 -> 15ede -> 7be72 -> 8b9eb -> cfcbe -> 9
9d29 -> 7ac7b -> 8f6cc -> 10777 -> cbe21 -> 677a4 -> d7bfd -> d9ff6 -> a6608 -> 14b0b -> 3f416 -> 7dfe2 -> 41072 -> 7fe42 -> d70ae -> b559f -> f068b -> c0da2 -> cee38 -> 95134 -> f3ee1
 -> 15ff6 -> 0bcd0 -> 3927f -> 70700 -> 458d6 -> ff749 -> aae64 -> d4299 -> 7d25b -> d816e -> 3f4fb -> 90290 -> f5094 -> 30704 -> 647af -> 4dcaf -> 6e8c2 -> d9396 -> 8e6f9 -> ceaeb ->
1e779 -> ef0b1 -> 1d259 -> bb562 -> 3eee1 -> 7d2d7 -> e6000 -> a7f6f -> 1a8db -> 4521a -> cd848 -> 7fc2d -> 2cc86 -> cfec3 -> 0a7a9 -> 0e16a -> e5200 -> b1b9b -> 7cd8b -> 60d52 -> aa7b
a -> 595f8 -> a7ca1 -> 3ce3f -> adc1f -> 790fd -> 1fc21 -> f5d8a -> b5df9 -> e8d70 -> d1ba6 -> ec3b1 -> f98e7 -> f6f39 -> d972b -> 573ab -> c5c16 -> 73c88 -> b85d9 -> 0c956 -> 93956 ->
 6b010 -> 292c6 -> d4615 -> 44234 -> bd449 -> 60f63 -> 861e4 -> b560c -> 1b369 -> 4e77b -> 70cf0 -> 96dd4 -> c59bd -> 0c7b9 -> 0637b -> 4d2c0 -> bbb28 -> 8df36 -> e0a06 -> 604fc -> 61a
04 -> bad43 -> 61e92 -> bc3a7 -> c3faf -> 328a2 -> 9be80 -> a38d8 -> 18474 -> 36aca -> 2d0ae -> 5db1a -> 76697 -> e2d42 -> 63a49 -> 8064f -> 84393 -> bebb5 -> 6b397 -> b465f -> 4ab5a -
> 9b30d -> a8c25 -> 4b986 -> 74aa7 -> 23239 -> 874e3 -> 703b4 -> b50c4 -> 3849e -> 22e22 -> ffb25 -> b7681 -> 3ba9e -> f7d41 -> 79146 -> 31fc0 -> e70a8 -> 19cb3 -> b06c7 -> af279 -> 80
a9e -> f546d -> 36116 -> ca409 -> d0993 -> fd1d4 -> d6ebf -> 36c1d -> c8783 -> 662e9 -> 4bdb7 -> 4b883 -> 64e7a -> 6e85a -> fdde0 -> 88b5c -> c828a -> 4bcd6 -> e85ee -> 7773f -> 5da87
-> 7fcd4 -> cce32 -> e228c -> 48feb -> 55f0d -> 42640 -> 9c5a3 -> 1ff43 -> a1727 -> 7cb4d -> 3a496 -> 09e90 -> d5aeb -> 2b603 -> fd53c -> 7040f -> f2fb7 -> ac2e5 -> da0ba -> 0ad98 -> c
fdb9 -> b8ffc -> 35be3 -> 9e57f -> 98ae6 -> 937c0 -> 4dddd -> 2a25c -> fb56c -> e701b -> d8c20 -> 9686e -> 7a2b1 -> 6cb16 -> c80c3 -> 98adb -> 9db58 -> 53257 -> b7969 -> ca598 -> dded9
 -> 681a5 -> e924d -> a833d -> 9bdee -> e53f8 -> ceae0 -> 144e9 -> e280f -> 12ddb -> c2856 -> c4459 -> 5042a -> 63f62 -> 47823 -> 9e912 -> fdcf4 -> c726b -> 12325 -> d1eaa -> 454af ->
f4a14 -> 8d2d7 -> 957cf -> f68b8 -> f02c2 -> d58f6 -> eae5e -> 85357 -> 66ceb -> f4456 -> 68ff5 -> f384c -> c3298 -> 4fa0b -> f22f2 -> 04f3f -> cd05f -> ff988 -> 54520 -> 400a2 -> 0db3
0 -> ca21a -> d24e7 -> 99bdc -> c15c0 -> 3f884 -> 42ba6 -> 1a5df -> 9302e -> 0ee10 -> ba3d9 -> c543e -> f8184 -> af3c2 -> 6d4ba -> 5bfe3 -> 198db -> 8c1e3 -> 49fc4 -> 71eab -> 2d349 ->
 0f199 -> bc97f -> a75a5 -> 116db -> fe7ae -> 8c81e -> 46e26 -> 26d4a -> 6389b -> 8e3a4 -> 71547 -> f1507 -> ae1e1 -> ccf12 -> b71c9 -> 5e408 -> 61f08 -> 3dd82 -> 5f5cc -> 95d5b -> 97f
15 -> 304bf -> 2db1d -> e9a50 -> 5aad7 -> 86e93 -> fb049 -> b97cd -> 9654a -> 09250 -> 15c85 -> ffe9e -> a7bac -> f0ed5 -> 73b8a -> 22d14 -> c9b4d -> 2e6f4 -> cc9f3 -> 0428d -> 2cf9b -
> b5bfd -> 19308 -> 9410e -> e1b9b -> 916a4 -> 8d4c3 -> 27e0c -> cec4c -> 6b450 -> d7b8b -> ead04 -> c7f7f -> ed4c1 -> 10f92 -> 33902 -> 6a7d5 -> 69f18 -> 7713f -> 421bf -> 4972e -> 8a
2d7 -> 5b318 -> d9536 -> 9a790 -> 57aa7 -> f4751 -> a0def -> a661b -> c5f90 -> 3a074 -> a7ebe -> 5b793 -> 92504 -> 3ffb2 -> 35c3d -> 97aa5 -> 97951 -> 67ce6 -> 02fe1 -> 7283d -> 7367c
-> 4709f -> 29645 -> ccae0 -> a5dd1 -> ea5e7 -> 79045 -> 22e48 -> b82a2 -> 44fb2 -> fdf68 -> 46b61 -> 67403 -> 5099b -> 50ed1 -> d3daf -> d2f04 -> ba3e8 -> e93d1 -> 6616d -> 0c987 -> 7
a64e -> f37cc -> ba063 -> 3db46 -> ff1b8 -> ba74e -> 2972a -> 9e51c -> 95f4d -> a4cdd -> 29946 -> b50ca -> 59add -> a02b4 -> ae31c -> 486aa -> 4afb3 -> d15fd -> 6c3a7 -> b61cf -> 57060
 -> 801c9 -> 4dffe -> 19884 -> ba542 -> 6dc46 -> d8bd1 -> 2ccf2 -> 62a5f -> 527ca -> e8c89 -> 3f894 -> 5439a -> 395c9 -> 6742c -> 54e6b -> 56bcd -> a0647 -> b4447 -> b51e0 -> 63cb4 ->
ea746 -> 5f5b4 -> 90125 -> a30f9 -> 05905 -> 96939 -> 67e12 -> ab447 -> afcc4 -> 29943 -> a47d7 -> c0b73 -> 30d1a -> bdafb -> 551e0 -> 93619 -> c7b1e -> 50735 -> 824ba -> 9182f -> dbd6
7 -> 6ed74 -> afe1b -> f1a4e -> 67275 -> 5ffeb -> 7b357 -> 44dd1 -> 8e105 -> a517b -> bf103 -> 803cc -> 36cae -> 789d9 -> 3d168 -> 0b647 -> e71b5 -> 497b9 -> faea3 -> 43aa8 -> 3a633 ->
 c232d -> 95ba0 -> bb3a6 -> 829fb -> fcbbe -> 03c99 -> 78be7 -> 2cc42 -> fa4e7 -> 0d210 -> c51e6 -> 780a4 -> 8fb58 -> f47a9 -> 87975 -> d102d -> 814da -> 594af -> 1eaf0 -> 58d22 -> d2c
1d -> 31a00 -> 394fd -> 8bdfb -> 3a16f -> 7c283 -> b9086 -> 1f724 -> af637 -> faeaa -> eac9b -> 5291f -> 29847 -> b37e5 -> 1c795 -> 38c6e -> f5b0f -> b1bf9 -> 10c55 -> ba8bd -> da011 -
> 810b6 -> bff1e -> 0598c -> bf0de -> 6c267 -> 480e3 -> 15f40 -> e5239 -> 53135 -> 987b5 -> cdfcf -> 4ee9a -> d9298 -> 3e026 -> 35c5c -> 74a15 -> b4b46 -> e8c5b -> 0868c -> cad0c -> 01
c6c -> 0951a -> f00de -> 4cf86 -> 09a87 -> 31838 -> 372a3 -> 846ba -> 4dd65 -> af4ce -> 5da55 -> 5fe9b -> a14f0 -> 0ad97 -> 3361b -> f76fa -> 83178 -> 436ec -> db60f -> e269e -> e1cd5
-> 8d330 -> 1512a -> 2bf09 -> 9fe9a -> 15e84 -> 08086 -> 9526d -> ba3ae -> dc716 -> 1c7e4 -> f2703 -> f9816 -> bd42e -> 4fc27 -> a7b0f
Tour length:: 11144539.27 m
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  13:36 min
Finished at: 2023-04-18T17:09:59-04:00
------------------------------------------------------------------------
```

Genetic algorithm



```
cd D:\Sem2\PSA\PSA_final_project; "JAVA_HOME=C:\\Program Files\\Java\\jdk-19" cmd /c "\"C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven\\bin\\mvn.cmd\" -Dexec.vmArgs= \"-Dexec.ar
gs=${exec.vmArgs} -classpath %classpath ${exec.mainClass} ${exec.appArgs}\" \"-Dexec.executable=C:\\Program Files\\Java\\jdk-19\\bin\\java.exe\" -Dexec.mainClass=com.mycompany.tsp.TSPG
eneticAlgorithm -Dexec.classpathScope=runtime -Dexec.appArgs= \"-Dmaven.ext.class.path=C:\\Program Files\\NetBeans-15\\netbeans\\java\\maven-nblib\\netbeans-eventspy.jar\" org.codehaus
.mojo:exec-maven-plugin:3.0.0:exec"
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects (with Compile on Save turned on) will be used instead of their jar
artifacts.
Scanning for projects...

------------------< com.mycompany:PSA_final_project >-------------------
Building PSA_final_project 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSA_final_project ---
Best route:70700->ceae0->4fc27->a7ca1->6a7d5->937c0->67275->afcc4->44234->c7f7f->faea3->f4a14->f4751->ca598->62a5f->0bcd0->987b5->9e57f->95f4d->a661b->7fa41->98adb->677a4->8a2d7->b8ffc
->73c88->0b647->3db46->03f24->c543e->2cc42->46b61->dded9->b50c4->6616d->458d6->1a8db->9fe9a->ceaeb->cbe21->38c6e->09a87->dc716->7b357->3a074->9bdee->7fe42->29847->15ff6->7d2d7->2972a->
527ca->3f894->328a2->57bc7->88b5c->c3faf->ba542->ba8bd->436ec->23239->b61cf->aa7ba->8c81e->846ba->0ee10->801c9->e93d1->97aa5->d0c60->74aa7->cc9f3->b97cd->fe864->f37cc->4dd65->8fb58->af
637->22d14->50ed1->c9b4d->cec4c->d2f04->1a5df->2db1d->3ff9d->395c9->1fc21->6b010->9686e->6e8c2->15c85->d0993->9e912->a47d7->b37e5->3d168->874e3->8c1e3->4ab5a->8df36->fe7ae->67e12->63cb
4->ba3e8->573ab->400a2->5e408->f2fb7->5fe9b->19cb3->b1bf9->e269e->60f63->0ad98->0f199->cce32->73b8a->4bcd6->d5aeb->36cae->049d1->bb562->789d9->af279->d9536->b1b9b->f7d41->2cf9b->e0a06-
>6cb16->55f0d->d3daf->e8c89->480e3->d816e->95134->0951a->0868c->b50ca->f5094->f1a4e->ba3ae->09250->ea5e7->64e7a->f5b0f->5ffeb->5db1a->66ceb->db60f->7fcd4->a0def->d2c1d->824ba->e5239->d
7b8b->e8c5b->292c6->bf103->8e6f9->61f08->bad43->8d2d7->d8c20->86e93->da011->7a64e->0ad97->41072->8e3a4->3eee1->cad0c->0c956->4afb3->a7b0f->31a00->99d29->6e85a->4a714->810b6->4ee9a->eac
9b->b559f->7dfe2->22e48->36116->ff988->c5c16->46e26->604fc->fdcf4->d7bfd->8d4c3->7cd8b->54e6b->fdf68->63f62->a7ebe->afe1b->9526d->3e026->35c3d->7a813->e85ee->6ed74->3ce3f->814da->4e77b
->681a5->b560c->0c7b9->9302e->5b318->f22f2->18474->394fd->ba063->a0647->2bf09->ff749->19884->05905->44dd1->3ba9e->b46af->c8783->1c795->29946->c51e6->98ae6->31fc0->f68b8->35be3->4709f->
15ede->7713f->ab447->f0621->3a496->d9ff6->4b883->2b603->15f40->fd53c->1512a->1c7e4->08086->f4456->d8bd1->a7bac->e0ca5->58d22->10777->bbb28->12325->af4ce->fb049->0c987->35c5c->99bdc->4d
caf->c232d->110be->e53f8->7ac7b->6389b->6c267->5f5b4->68ff5->d6ebf->10f92->0428d->9be80->a6608->61e92->198db->5bfe3->1ff43->51f69->916a4->cd848->bff1e->d4299->1eb20->454af->f98e7->fcbb
e->14b0b->15e84->a1727->9db58->d70ae->4cf86->fd1d4->5da87->5da55->90290->93956->e924d->3f4fb->c80d0->c3298->8d330->fa4e7->4bdb7->50735->53257->ec3b1->ceab2->b82a2->e8d70->bb3a6->67ce6-
>af3c2->0637b->3dd82->7be72->71eab->e61cc->c80c3->10c55->bf0de->cdfcf->b71c9->bc97f->8064f->7c283->d9396->f068b->56bcd->9654a->63b73->36aca->90125->44fb2->d58f6->144e9->02057->26d4a->a
ae64->09e85->78be7->8f6cc->eae5e->551e0->b7969->861e4->b4b46->ba3d9->595f8->9a790->5042a->7040f->cfcbe->e9a50->7773f->c2856->a517b->f384c->d1ba6->cfec3->2d0ae->53135->3849e->0a7a9->f81
84->c7b1e->662e9->f2703->a4cdd->6afd9->04f3f->c0b73->0598c->2cc86->dbd67->803cc->ae1e1->5099b->c4459->957cf->9c5a3->a75a5->b06c7->ead04->6c3a7->97951->e228c->bebb5->57060->f3ee1->1e779
->4dddd->780a4->3a16f->adc1f->3f884->19308->bdafb->85357->92504->f6f39->b85d9->f02c2->42ba6->5b793->ca409->5291f->1f4d0->d1eaa->6b450->43aa8->790fd->6742c->ea746->5aad7->84393->76697->
c59bd->4972e->1eaf0->7367c->c3355->57aa7->f9816->2ccf2->69f18->61a04->3a633->ef0b1->c5f90->60f32->a30f9->f47a9->116db->8e105->71547->ffe9e->95ba0->c15c0->f4d9e->a38d8->b5bfd->a833d->e2
80f->31838->faeaa->6d4ba->bc3a7->fb56c->3b420->7e786->b5df9->29645->e701b->b51e0->12ddb->0d210->829fb->fdde0->79045->42640->e1b9b->da0ba->bd449->60d52->79146->9410e->ae31c->87975->c726
b->304bf->97f15->a7f6f->f76fa->3361b->3ffb2->d24e7->ed4c1->27e0c->e70a8->e5200->1f724->67403->f5d8a->ff1b8->54520->b8962->5f5cc->4d2c0->1b369->33902->486aa->d102d->8b9eb->a8c25->ba74e-
>d15fd->cee38->cfdb9->0e16a->e71b5->421bf->594af->5439a->7283d->96939->9b30d->83178->2d349->d9298->bb937->2e6f4->e6000->03c99->0db30->a14f0->02fe1->22e22->a5dd1->95d5b->497b9->4fa0b->b
9086->7a2b1->70cf0->1d259->ca21a->30704->96dd4->f546d->647af->3927f->bd42e->01c6c->f0ed5->ccae0->cbe84->80a9e->ffb25->09e90->74a15->7fc2d->4b986->6dc46->d4615->e71b5->703b4->34272->ccf
12->9e51c->b465f->cd05f->d972b->4521a->36c1d->b7681->e1cd5->c0da2->c828a->b4447->63a49->93619->2a25c->6b397->9182f->59add->47823->a02b4->48feb->29943->f1507->ac2e5->49fc4->7cb4d->3f416
->f00de->30d1a->e2d42->8bdfb->ec311->7d25b->4dffe->372a3->70700->Tour length:: 8485469.35 m
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  02:24 min
Finished at: 2023-04-18T16:58:53-04:00
------------------------------------------------------------------------
```

Conclusion

From the graph above and after testing five different algorithms we conclude that, 2 opt method gives the least distance (64878.5m), hence is the best route. While, on the other hand, Simulated Annealing returns the highest distance (112266889.71m) and is the worst path to be chosen.

References

- **https://en.wikipedia.org/wiki/Christofides_algorithm**
- **https://bochang.me/blog/posts/tsp/**
- **https://slowandsteadybrain.medium.com/traveling-salesman-problem-ce78187cf1f3**
- **https://journalofbigdata.springeropen.com/articles/10.1186/s40537-018-0122-y#:~:text=Random%20swap%20algorithm%20aims%20at,simple%20to%20implement%20and%20efficient**
- **https://towardsdatascience.com/how-to-solve-travelling-salesman-problem-with-simulated-annealing-c248447a8bcd**
- **https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5676484/**