This project aims to plan the concert tours of The Weeknd using heuristic algorithms in the Java programming language. The main goal is to create an efficient route to visit the target cities starting from the starting route, New York, and selecting the best set of songs for each location.

Knapsack Algorithm

First, we make a list of songs by calling them. Then, like in the bottom up technique, we generate a 2-dimensional array with the maximum values being v[][]. v provides the most value in each capacity. v computes the popularity value by picking the songs with the highest popularity for each performance length. By repeating this procedure for each city, we arrive at the overall popularity. It is achieved by estimating the greatest song combination that each concert may have while staying under time limits. We have N songs and the maximum capacity of a concert is W. The complexity of the Knapsack method is O(WN).

Knapsack Complexity

W= Maximum capacity

N= Number of songs

$$\boldsymbol{O(WN)}$$

TSP Algorithm

The algorithm used in this project to generate the tour route is the Nearest Neighbor Heuristic Algorithm for the Travelling Salesman Problem (TSP). The algorithm starts in New York, the city where The Weeknd is currently based, and marks this city as the initial route. Then, among the cities that have not yet been visited, the algorithm identifies the city that is the nearest distance away and adds that city to the tour list as the next visited city. This process continues until all cities have been visited. Each time, the city that is the nearest distance away from the last visited city is found and added to the tour list as the next visited city. After all cities have been visited, the tour is completed by returning to the starting city. The time complexity of this algorithm is O(N^2). N represents the number of cities, and the algorithm contains a nested loop. When we examine the algorithm in more detail, the first step, determining the starting city, is a fixed process and the time complexity of this step is one. Then, the outer loop then continues running until all cities have been visited and runs N-1 times. The inner loop runs N times for each iteration of the outer loop. The nested loop finds the nearest neighbor for each city. Within the nested loop, constant-time operations are performed to calculate the distance between two cities and to find the shortest distance. As a result, the total number of operations is N * (N-1) * O (1). It is dominated by the main loop, which has an overall running time of O(n^2). Other constant-time operations add to the overall efficiency but do not change the dominant term. Therefore, the time complexity of this algorithm is expressed as O(N^2).

TSP Complexity

N=Number of cities

$$\boldsymbol{O}(\boldsymbol{N}^{\mathbf{2}})$$

Results

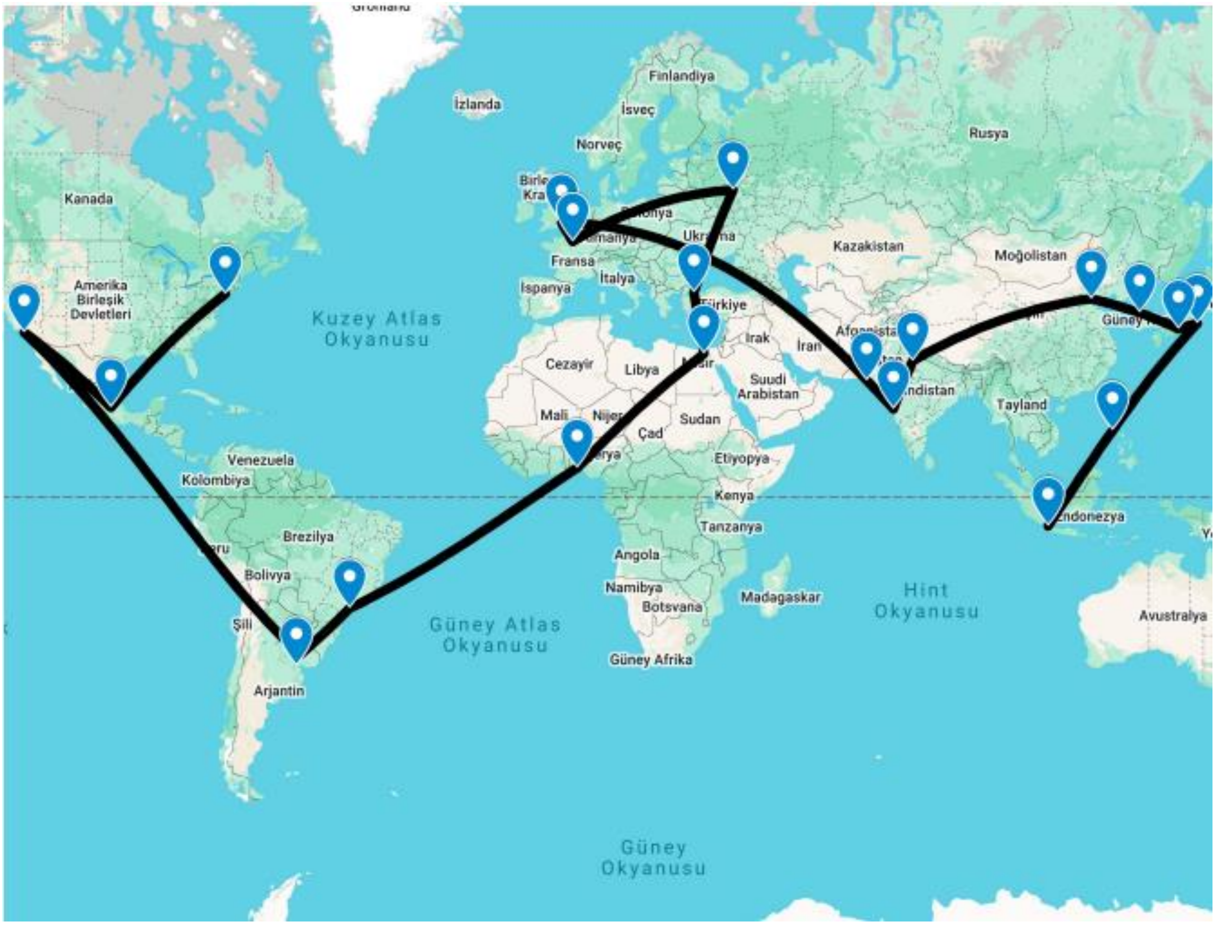
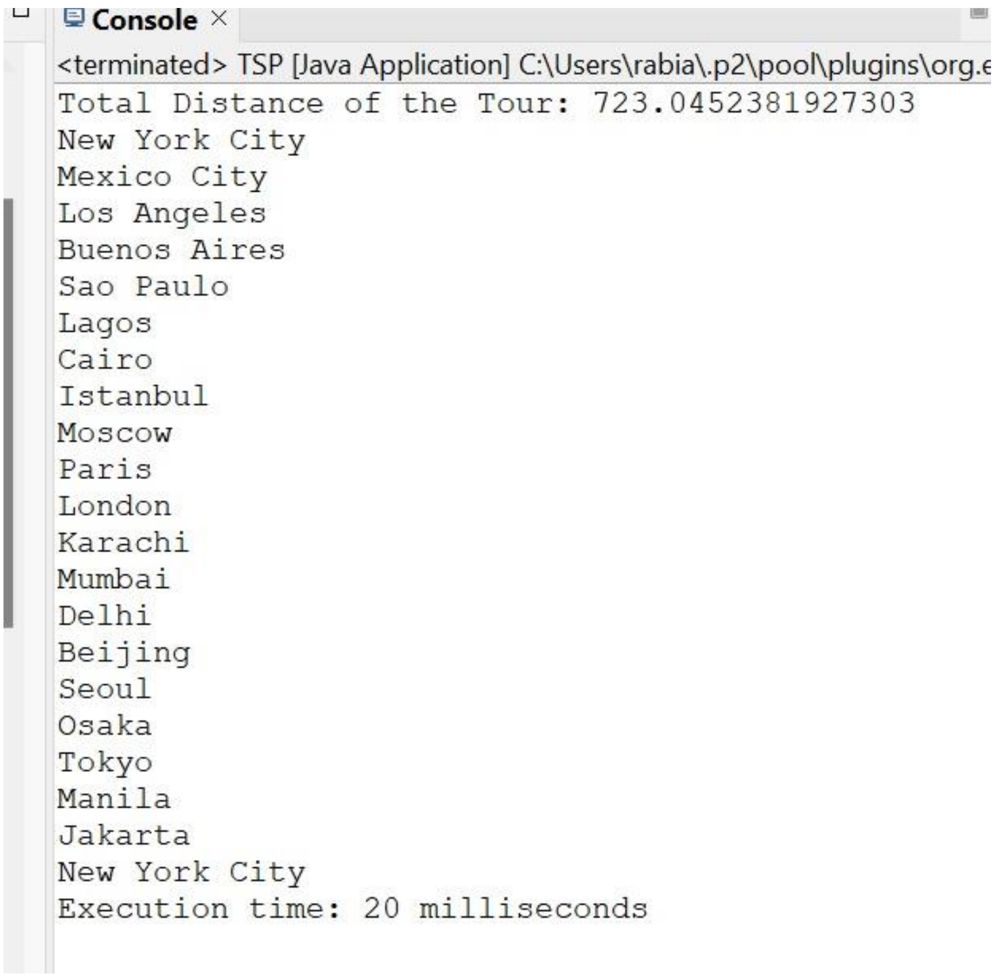
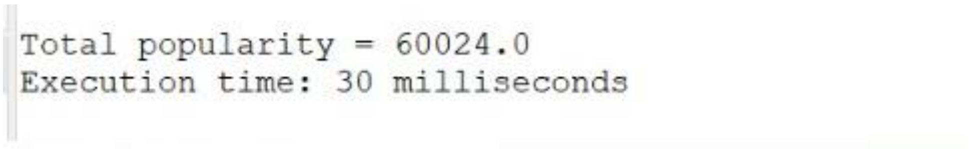


Figure 1: TSP Algorithm Route



```
<terminated> TSP [Java Application] C:\Users\rabia\.p2\pool\plugins\org.e
Total Distance of the Tour: 723.0452381927303
New York City
Mexico City
Los Angeles
Buenos Aires
Sao Paulo
Lagos
Cairo
Istanbul
Moscow
Paris
London
Karachi
Mumbai
Delhi
Beijing
Seoul
Osaka
Tokyo
Manila
Jakarta
New York City
Execution time: 20 milliseconds
```

Figure 2: TSP Algorithm Result



```
Total popularity = 60024.0
Execution time: 30 milliseconds
```

Figure 3: Knapsack Algorithm Result