Exercise # 3

A. Consider the following Travelling Salesman Problem.

An aspiring politician wants to be the mayor of his town in the next local election. As early as now, he is already planning to visit three barangays around the town on the first day of the campaign period. On that day, he will be in his headquarters (home) and must visit barangays A, B, and C and then return to his headquarters. He wants to minimize the total cost of his trip. In what order should he visit the cities? Will this trip be possible if his allotted fund for this campaign is Php 500,000 only? The following table shows the estimated campaign costs (in hundred pesos) between the four places. (Don't use swapping of nodes as a definition for a neighborhood of a solution.)

	Home	A	B	C
Home	-	1559	921	1334
A	1559	-	809	1397
B	921	809	-	521
C	1334	1397	521	-

Objective Function:

For this problem, our objective is to minimize the total cost of the aspiring politician's trip.

Constraints:

The constraint is that his allotted fund for the campaign is Php 500,000 only.

Search Space/Domain:

Since the path is symmetric with only 4 nodes, the search space is $\frac{(n-1)!}{2}$.

Solution:

Using an approximate method, in particular, a hill-climbing algorithm while also utilizing the Python code, the best path for the trip is [0, 1, 2, 3, 0] or Home-Barangay A-Barangay B-Barangay C-Home. The total cost of this trip is only Php 422,300 which is less than the allotted fund and is enough to complete the trip without overspending.

In the Python program, the stopping criterion used is if the new path has a higher cost or if it is not within the allotted fund. Nevertheless, if the new path has a lower cost and is within the allotted fund then the iteration will continue. Since the search domain is very small, the solution obtained is the optimal solution.

B. Consider the following Knapsack Problem.

With a calorie budget of 750 calories, choose an optimal set of foods from the menu.

Food	wine	beer	pizza	burger	fries	coke	apple	donut
Value	89	90	30	50	90	79	90	10
calorie s	123	154	258	354	365	150	95	195

Objective Function:

For this problem, we want to minimize the value while keeping in mind that we have at least 750 calorie budget. Using this analysis, we want to make a binary variable (x_i) where x_i takes the values of either 1 (the food will be chosen) or 0 (otherwise) where i is from 1 to 8 assigning each food to its corresponding number respectively, that will help decide which food to eat. Thus, the objective function is to

$$minimize\ z = 89x_1 + 90x_2 + 30x_3 + 50x_4 + 90x_5 + 79x_6 + 90x_7 + 10x_8$$

Constraints:

Also, the constraints are

$$subject\ to:\ 123x_1 + 154x_2 + 258x_3 + 354x_4 + 365x_5 + 150x_6 + 95x_7 + 195x_8 \geq 750$$
$$x_i\ is\ a\ binary\ variable\ where\ i = 1, 2, ..., 8$$

Search Space/Domain:

Since x_i is a binary variable, its domain is (0,1). Furthermore, since we have 8 binary variables the search space is 2^n or simply $2^8 = 256$ possible solutions.

Solution:

Using Python to implement the constructive method used, specifically hill-climbing algorithm. The solution shows that the optimal solution is [0, 0, 1, 1, 0, 0, 0, 1] or simply the best foods to eat are pizza, burger, and donut. This selection only has a total of 807 calories with a value of only 90.

In the Python code, the stopping criterion is if there is no neighborhood that has a lower objective value. The neighborhood is obtained by flipping a single bit or changing 0 to 1 or vice versa. Since this is a random output, it may have different outputs. However, I personally picked the output with the lowest value as discussed earlier. The solution that I highlighted here is the optimal feasible solution.

C. Consider the following unconstrained nonlinear optimization problem.

$$Min\ f(x,y)\ =\ x^2 + y^2 - 6x + 4y$$

Objective Function:

Obviously, given in this problem is the objective function

$$Min\ f(x,y)\ =\ x^2 + y^2 - 6x + 4y$$

Constraints:

Since this an unconstrained nonlinear optimization problem, this obviously has no constraints also.

Search Space/Domain:

The domain of this problem is all real numbers.

Solution:

Using Python to solve the problem applying hill-climbing algorithm, the output shows a minimum point (x, y) = (3.0000000000000013, -2.0000000000000004) while the minimum function value is -13.0. The stopping criterion used in the program is the number of maximum iterations which is 1000. In finding the optimal solution, I tried increasing the number of iterations to determine if there are changes in the solution. However, since it shows the same answer, the solution I got is the optimal solution for the problem.

NOTE: The documentation of the Python code is incorporated inside the .ipynb file together with the code.