Report on using GRASP to find max cut of a graph
Name: Kazi Istiak Uddin Toriqe
Student ID: 1905104

Complexity :

- Semi Greedy & Greedy = $O(n^2)$
- Local Search = $c\,O(n^2)$; 10 <= c <= 50
- Grasp = $1000\,O(n^2) = d\,O(n^2)$; $100 <= d <= 1000$

Randomized:
We have randomly put the vertices in either of the sets with ½ probability and then calculated the max cut . As average result cannot be a desired cost(because there might be possibility that that cost would never come).So we avoid taking the average

Greedy:
We first select the two connecting nodes from the largest cost edges .For greedy we have chosen α = 1 . And used mentioned heuristic to calculate the cost.A each step we are inserting a vertex in our one of our two sets .For this problem in most of the cases ,greedy solution was better than semi-greedy or randomized ones . Somehow greedy solution find a solution to nearby global maxima.

Semigreedy:
We utilized a value-based approach to form a restricted candidate list (RCL) from a set of edges. In this case, a random real value, denoted as α [0 ≤ α ≤ 1], was generated and multiplied by the maximum edge cost. Any edges with a weight greater than or equal to this value were included in the RCL. Subsequently, a random edge from the RCL was selected, and its two endpoints were placed in separate sets. This process was repeated until all vertices were assigned to either Set X or Set Y. The following steps were followed:

1. Candidate elements refer to vertices that have not yet been assigned to either Set X or Set Y.

2. For each candidate vertex, we computed the cost it would contribute to the maximum cut if added to Set X or Set Y.

3. A random cutoff value, which is α times the maximum potential contribution of a vertex, was generated.

4. Vertices meeting or exceeding this cutoff value were assigned to another RCL.

5. A random vertex from this new list was chosen and added to the set that maximized the value of the maximum cut.

To account for variations in the results based on the value of α, we repeated this procedure ten times and recorded the average value of the maximum cuts.

Local Search:
Following the construction phase, we implemented a process known as local search. During this phase, we attempted to improve the value of the maximum cut by switching the placement of one vertex from one set to the other. We continued this process until reaching a point where the maximum cut had reached its maximum value. At that stage, we terminated the process and recorded the achieved maximum cut.

Performance Analysis:

The randomized algorithm performs very poorly in case of negative edges .In semi-greedy algorithm we are not guaranteed to get stuck on local maxima (in grasp) because at each step the semi greedy solution provides some randomness.So if it gets stuck in local maxima there's a higher probability that it can reach nearer to peak in other iterations.As a result in GRASP procedure we get maximum possible cost as we iterate them for 1000 times randomness.But generally after 50 iterations it doesn't change much in cost.That means increasing iteration in certain level is good.But due to computational complexity we had to avoid much more iterations.SO local search did good on this problem.For this problem in most of the cases ,greedy solution was better than semi-greedy or randomized ones . Somehow greedy solution find a solution to nearby global maxima.But after iteration the semigreedy performs better than the greedy one!