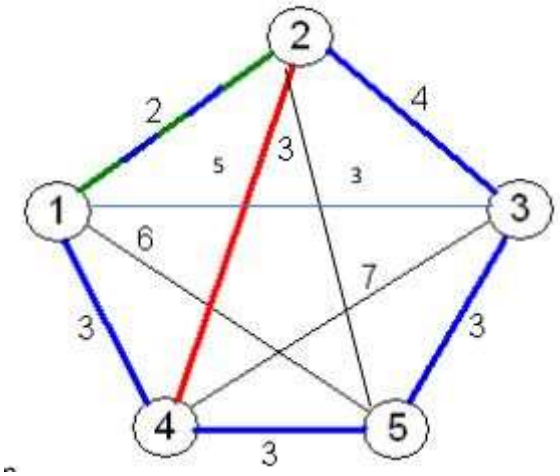
TSP Using Simulated Annealing



In [3]:

```
tsp = [
    [0,2,3,3,6],
    [2,0,4,5,3],
    [3,4,0,7,3],
    [3,5,7,0,3],
    [6,3,3,3,0]
]
```

To calculate path cost

In [4]:

```
def pathCost(path):
    cost = 0
    for i in range(len(path)):
        cost += tsp[path[i - 1]][path[i]]
    return cost
```

To generate random initial path

In [5]:

```
import random
initial_route = []
temp = [i for i in range(len(tsp))]
for i in range(len(tsp)):
    randomCity = temp[random.randint(0, len(temp)-1)]
    initial_route.append(randomCity)
    temp.remove(randomCity)
```

To generate new route by swapping two cities

In [13]:

```
def swap(route,i,j):
    route[i],route[j]=route[j],route[i]
    return route
```

Algo

In [58]:

```
import math
temp = 150
cur_route = initial_route
cur_cost = pathCost(initial_route)
print("Initial path : {0} and it's cost: {1}".format([i+1 for i in cur_route],cur_cost)
while(temp>0):
    for i in range(10):
        r1 = random.randint(0,4)
        r2 = random.randint(0,4)
        while(r1==r2):
            r2 = random.randint(0,4)
        next_route = swap(cur_route,r1,r2)
        next_cost = pathCost(next_route)
        D = next_cost - cur_cost
        if( D < 0 ):
            cur_cost = next_cost
            cur_route = next_route
        elif(random.random() > (math.exp((-1*(D/temp))))):
            cur_cost = next_cost
            cur_route = next_route
    temp-=10
print("Final path : {0} and it's cost: {1}".format([i+1 for i in cur_route],cur_cost))
```

```
Initial path : [1, 3, 5, 2, 4] and it's cost: 17
Final path : [3, 1, 5, 4, 2] and it's cost: 21
```