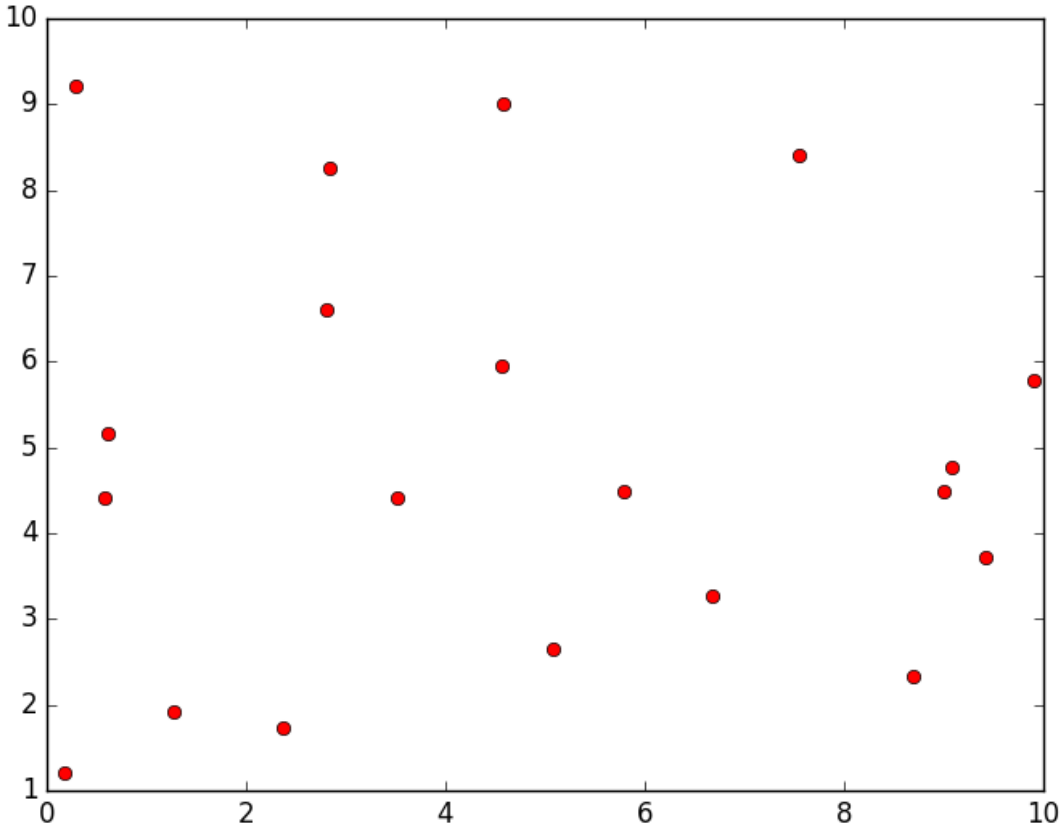
1. Create 20 cities



2. Calculate distances between them:

NOTE: Table is not ready yet...

Source code:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
__author__ = 'ar1'
import random
import math
import matplotlib.pyplot as plt
import itertools
def main():
    cities = gen_cities(20, 0, 10, 0, 10)
    print(cities)
```

```
    plot_cities(cities)
    distances = calc_distances(cities)
    print(distances)
    print(calc_all_allowed_ways_and_their_distances(cities, distances))
def calc_all_allowed_ways_and_their_distances(cities, distances):
    ids = [city['id'] for city in cities]
    ans = []
    for cities_idx in itertools.permutations(ids, len(cities)):
        ans.append(cities_idx)
    ways = [list(way) for way in ans[0:len(cities) - 1]]
    for way in ways:
        way.append(way[0])
    print(ways)
    results = []
    for way_city in ways:
        dist = 0
        for city_id in xrange(len(way_city) - 1):
            for distance in distances:
                if (way_city[city_id] is distance['from'] and way_city[city_id +
1] is distance['to']) or (
                        way_city[city_id] is distance['to'] and way_city[city_id
+ 1] is distance['from']):
                    dist += distance['dist']
                    # print("Step", way_city[city_id], way_city[city_id + 1])
                    break
        print(dist)
        results.append({'way': way_city, "dist": dist})
    return results
def plot_cities(cities):
    x = [city['x'] for city in cities]
    y = [city['y'] for city in cities]
    plt.plot(x, y, 'ro')
    # for x_c, y_c in zip(x, y):
    #     plt.text(x_c, y_c, "%6.2f : %6.2f" % x_c, y_c)
    plt.show()
def gen_cities(N, from_x, to_x, from_y, to_y):
    cities = []
    id = 'Z'
    for i in xrange(N):
        cities.append({"x": random.uniform(from_x, to_x), "y":
random.uniform(from_y, to_y), "id": id})
        if id is 'Z':
            id = 'A'
            continue
        id = chr(ord(id) + 1)
    return cities
def calc_dist_between_points(point_1, point_2):
    return math.sqrt(math.pow(point_1['x'] - point_2['x'], 2) +
math.pow(point_1['y'] - point_2['y'], 2))
def calc_distances(cities):
    dists = []
    n = len(cities)
    for i in xrange(n):
        for j in xrange(i + 1):
            r = calc_dist_between_points(cities[i], cities[j])
            print(r, i, j, cities[i], cities[j])
            dists.append({'from': cities[j]['id'], 'to': cities[i]['id'],
"dist": r})
    return dists
```

```
if __name__ == '__main__':
    main()
```