UNIVERSITAS INDONESIA

FARTHEST TO NEAREST METHOD

Recommendation for Improving the Initial Solution Method for the Vehicle Routing Problem (VRP)

FAUZIAH AROFAH
NPM. 1606842316

ASMA ROSYIDAH
NPM. 1306369895

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK INDUSTRI
DEPOK
2018

Abstrak

Vehicle Routing Problem (VRP) dapat memberikan solusi rute distribusi berdasarkan geografis titik tujuan dengan mengoptimalkan jarak tempuh serta kapasitas. Dalam mencari solusi inisiasi VRP, terdapat dua metode yang dapat digunakan; *Random* dan *Nearest Neighboar* (NN). Metode *random* terbentuk dengan menghubungkan nodes secara acak yang disesuaikan dengan kapasitas maksimum dan demand setiap nodes. Terlihat bahwa metode *random* sangat lemah dalam konsistensi setiap running. Metode NN terbentuk dengan menghubungakan nodes berjarak terdekat dengan node sebelumnya yang juga disesuaikan dengan kapasitas maksimum dandemand setiap nodes, yang berarti, *node* dengan letak terjauh dari depot pusat diperhitungkan paling akhir. Padahal *node* terjauh mempunyai peranan penting dalam menentukan rute optimal.

Kelemahan dari metode *random* dan NN melahirkan metode FTN yang mana mengutamakan node terjauh, batas geografis, dan pengisian node dari dua sisi sehingga dapat membentuk *cycle* yang lebih baik. Dari segala kekurangan dan kelebihan yang diberikan oleh metode *random*, NN, dan FTN, metode FTN memberikan performansi terbaik pada 15 *study cases* pengujian. Bahkan metode FTN memberikan perbedaan yang sangat signifikan pada *cases* dengan jumlah *nodes* diatas 100. Hal tersebut dapat memberikan kesimpulan bahwa *nodes* dengan jarak terjauh, batas, geografis, serta pembentukan rute *cycling* merupakan faktor yang berpengaruh dalam pembentukan solusi inisiasi yang lebih baik.

DAFTAR ISI

Abstrak i
DAFTAR ISI ii
I. PENDAHULUAN 1
II. METODE FARTHEST TO NEAREST (FTN) 2
III. PEMBAHASAN 4
IV. KESIMPULAN 16
DAFTAR PUSTAKA 17
FLOWCHART FTN 18
MANUAL BOOK 20
STUDY CASES 26
RUNNING PROGRAM 39
HASIL *RUNNING PROGRAM* 49
HASIL *RUNNING* VRP *SOLVER* 62
SOURCE CODE PROGRAM 66

I. PENDAHULUAN

Vehicle Routing Problem (VRP), merupakan solusi dalam proses distribusi dengan lingkup besar. VRP dapat memberikan solusi rute untuk mendistribusikan produk berdasarkan geografis konsumen, dengan mengoptimalkan jarak tempuh, serta mengoptimalkan kapasitas, yang berarti akan meminimumkan *cost* perusahaan. Hal tersebut membuat konsep optimalitas dalam *operation research* (yang mana mengandung unsur pengambilan keputusan berdasarkan ilmu pengetahuan, matematika, dan logika), menjadi sangat dibutuhkan dalam metode VRP. Saat ini terdapat beragam jenis VRP berdasarkan faktor yang dipertimbangkan. Namun, tulisan ini hanya akan membahas VRP dengan faktor kapasitas kendaraan.

Dalam VRP, terdapat dua buah tahap penyelesaian; inisiasi dan iterasi. Tahap insiasi merupakan tahap pencarian solusi inisiasi yang bertujuan untuk mendapatkan rute yang layak. Dengan solusi inisiasi yang tepat, maka solusi optimal akan lebih mudah didapatkan. Setelah solusi inisiasi didapatkan, dilakukan tahap iterasi. Tahap iterasi merupakan tahap dimana dilakukan evaluasi terhadap solusi inisiasi. Umumnya pada tahap ini dilakukan *switch nodes* antar kendaraan yang dilakukan secara berulang kali. Setelah itu data tersebut akan di *record* dalam suatu wadah (tergantung pada metode pendekatannya) dan akan dilakukan perbandingan berdasarkan jarak atau *cost* sehingga akan didapatkan hasil optimum untuk rute tersebut. Pada pembahasan berikutnya penyusun akan fokus pada tahap inisiasi. Tedapat beberapa algoritma yang digunakan dalam menentukan solusi inisiasi dari VRP, yaitu:

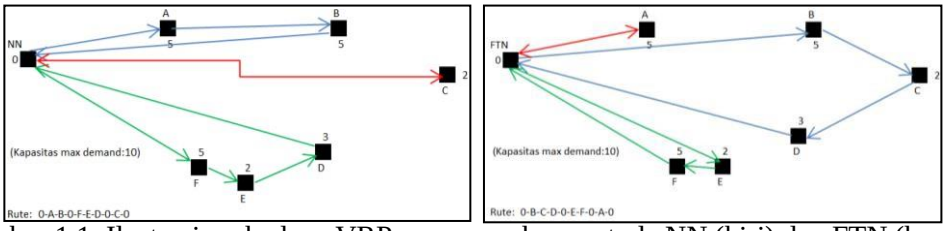
1. *Random*
   Metode penentuan solusi inisiasi VRP dengan algoritma *random* diawali dengan melakukan pemilihan secara acak *nodes* yang ingin dihubungkan dari suatu depot pusat hingga terjadi kekurangan kapasitas muatan kendaraan pengangkut dimana permintaan selanjutnya tidak dapat diangkut. Ketika hal tersebut terjadi, rute kendaraan diarahkan kembali ke depot pusat. Kemudian, dilakukan kembali pemilihan secara acak *nodes* yang belum dikunjungi melalui cara penentuan rute yang sama. Langkah tersebut diulang hingga seluruh *node* terlewati. Metode *random* merupakan metode yang paling tidak konsisten karena tidak ada aturan dalam pemilihan *node* menjadi sebuah rute. Buktinya, hasil *running* metode ini bisa berbeda-beda pada setiap *running*.

2. *Nearest Neighbor* (NN)
   Pada algoritma ini, penentuan *node* tujuan dalam pembuatan rute dilakukan berdasarkan jarak dengan *node* terdekat yang belum dikunjungi. Pada langkah pertama, dilakukan penentuan *node* pertama melalui pencarian jarak terdekat antara depot pusat dengan *node* tujuan yang menjadi kandidat rute. Berangkat dari *node* pertama, dilakukan kembali pencarian jarak terdekat dari *node* tersebut untuk *node* selanjutnya yang belum dikunjungi. Hal ini terus dilakukan sampai terjadi kekurangan kapasitas muatan kendaraan pengangkut dimana permintaan selanjutnya tidak dapat diangkut. Ketika hal tersebut terjadi, rute kendaraan diarahkan kembali ke titik depot pusat. Kemudian, dilakukan lagi penentuan rute untuk distribusi ke *node* yang belum dikunjungi melalui cara yang sama. Langkah tersebut diulang hingga seluruh konsumen terlayani. Metode ini lebih baik jika dibandingkan dengan metode *random*. Namun, *node* terjauh akan menjadi *node* yang paling akhir diperhitungkan. Padahal, *node* terjauh merupakan *node* yang paling membutuhkan perhatian karena memerlukan bahan bakar yang paling banyak serta mempengaruhi jarak dalam sebuah rute.

Terinspirasi dari kekurangan pada metode NN, penyusun memiliki gagasan untuk membentuk tahap inisiasi dengan langkah mundur, yaitu mencari *node* (kota tujuan/ letak konsumen) terjauh dari depot pusat (titik pusat) terlebih dahulu. Hal ini bertujuan untuk mengoptiomalkan rute menuju *node* terjauh tersebut serta mengantisipasi *node* pencilan dari *nodes* yang ada. Metode tersebut dinamakan *Farthest to Nearest (FTN).* Perbedaan antara NN dan FTN diilustrasikan oleh Gambar 1.1.

Pada Gambar 1.1. terlihat bahwa metode NN menghasilkan jarak yang kurang optimum jika dibandingkan dengan FTN. Pada NN, perjalanan menjadi kurang optimum ketika mencapai *nodes* C,

karena rute harus menuju *nodes* terjauh tanpa mengunjungi *nodes* lainnya. Secara logika, hal tersebut dapat membuat *cost* menjadi lebih mahal jika dibandingkan dengan menempuh *nodes* terjauh dengan menyertakan kunjungan ke *nodes* penghubung lainya seperti pada metode FTN. Pada FTN juga terlihat bahwa *nodes* yang tersisa adalah *nodes* A yang mana letaknya lebih dekat dari depot pusat. Dengan FTN, solusi inisiasi akan lebih optimal dalam jarak tempuh sehingga waktu yang diperlukan tahap berikutnya diharapkan akan lebih cepat.



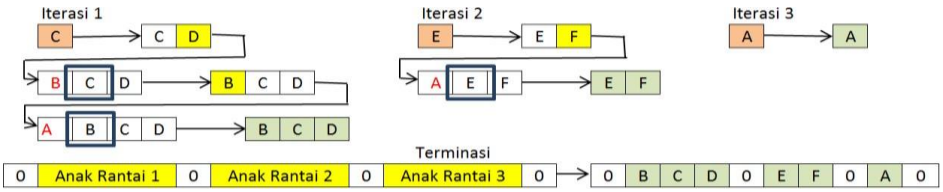
Gambar 1.1. Ilustrasi perbedaan VRP menggunakan metode NN (kiri) dan FTN (kanan). Rute biru= iterasi 1; rute hijau= iterasi 2; rute merah= iterasi3.

II. METODE FARTHEST TO NEAREST (FTN)

Metode terdahulu (NN) pada permasalahan solusi awal rute VRP, cenderung menitikberatkan *node* terdekat didalam himpunan sebagai titik utama. Sedangkan, *node* terjauh diperhitungkan setelahnya, bahkan cenderung kurang diperhitungkan sebagai suatu value yang memiliki kontribusi terhadap efisiensi. Hal tersebut memberikan dorongan dalam pencarian suatu metode pencarian solusi awal yang lebih baik agar pergerakan antar *node* dapat terdistrubusi merata. Melalui pendekatan FTN diharapkan kecenderungan terhadap *node* terjauh yang diabaikan dapat dibalik menjadi sesuatu yang memiliki nilai lebih dalam mengembangkan solusi awal permasalahan VRP yang lebih optimal.

FTN merupakan suatu metode pencarian solusi awal (*initial solution*) dimana mempertimbangkan *node* terjauh dari depot pusat (titik pusat) sebagai landasan pembentukan rute distribusi pada permasalahan VRP. Pembentukan titik sebelum dan selanjutnya dilakukan dengan menentukan *node* terdekat dari *node* terjauh yang sudah berdasarkan depot pusat. *Node* dengan jarak terdekat dari *node* terjauh dan memiliki *demand* yang masih sesuai kapasitas maksimal akan terpilih. Penyusun menginisiasi pembentukan rute dengan membandingkan dua buah calon *node* (kanan dan kiri) yang akan dijadikan acuan *node* berikutnya, sehingga dapat membentuk sebuah kesatuan rute yang memutar dari dan menuju depot pusat. Dengan kata lain untuk setiap titik telah dikunjungi, rantai yang terbentuk akan menghubungkan titik pusat sebagai titik awal, titik akhir, dan titik penghubung antar *nodes* untuk menghasilkan satu kesatuan atau rute sebagai solusi awal dalam permasalahan VRP. Metode ini akan melakukan looping iterasi sampai dengan *node* tersebut ditemukan dan membentuk sebuah anak rantai. Jika sudah sampai diambang batas looping namun *node* yang masih masuk dalam kriteria tidak ditemukan, maka akan dilakukan iterasi pembuatan rute baru dengan cara yang sama (untuk *nodes* yang belum terpilih), yang nantinya akan diterminasikan dengan iterasi sebelumnya, seperti divisualisasikan oleh Gambar 2.1.

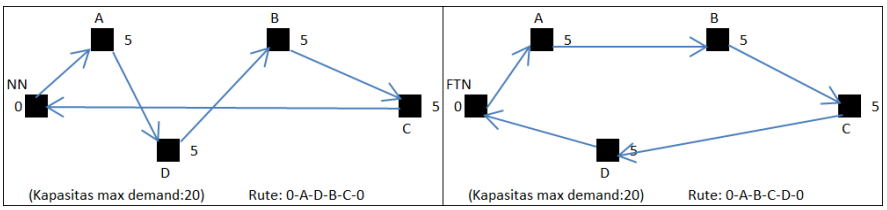
Pada Gambar 2.1., *node* C merupakan *node* terjauh dalam himpunan, sehingga C menjadi *node* awal. Kapasitas maksimum harus dikurangi dengan demand C, sehingga dapat diketahui kapasitas sisanya. Selanjutnya metode FTN akan mencari *node* yang terdekat dengan C, yaitu D. Lalu dilakukan pengecekan demand D, karena demand D tidak melebihi kapasitas sisa, maka D menjadi *node* setelah C, dan dilakukan perhitungan kapasitas sisa. Tahap berikutnya,



Gambar 2.1. Visualisasi tahap terminasi metode FTN

membandingkan antara C dan D, *node* mana yang lebih jauh dari depot pusat. Ternyata C lebih jauh dibandingkan D. Maka dilakukan pencari *node* terdekat dengan C dan didapatkan B. Lalu dilakukan pengecekan demand B, karena demand B tidak melebihi kapasitas sisa, maka B menjadi *node* sebelum C, dan dilakukan perhitungan kapasitas sisa. Tahap berikutnya, membandingkan antara B dan D, *node* mana yang lebih jauh dari depot pusat. Ternyata B lebih jauh dibandingkan D. Maka dilakukan pencari *node* terdekat dengan B dan didapatkan A. Lalu dilakukan pengecekan demand A, ternyata demand A melebihi kapasitas sisa, maka dilakukan pencarian *node* terdekat dengan B yang demandnya tidak melebihi kapasitas sisa, ternyata tidak ada lagi. Maka B-C-D membentuk anak rantai 1 dan dilakukan iterasi selanjutnya dengan cara yang sama hingga semua *nodes* terlewati. Setelah semua *nodes* terlewati, dilakukan terminasi antara anak rantai yang terbentuk sehingga menjadi sebuah kesatuan rute.

Hal diatas membuat sebuah perbedaan lagi dengan metode NN. Tahap membandingkan jarak antara dua *nodes* yang terpilih dengan depot pusat lalu memilih *node* terjauh diantaranya



Gambar 2.2. Ilustrasi pembentukan rute VRP menggunakan metode NN dan FTN

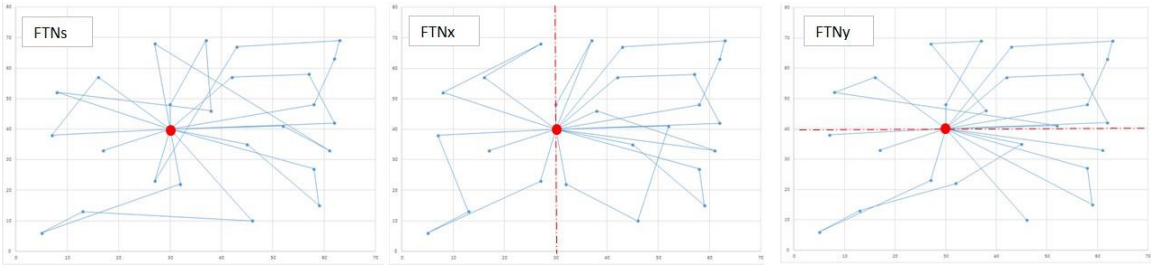
sebagai acuan *node* tujuan, menjadikan FTN dapat membentuk rute *cycle* seperti diilustrasikan dengan Gambar 2.2. Pada Gambar 2.2, terlihat bahwa titik C pada NN harus ditempuh terakhir dan secara kasat mata memberikan pengaruh yang besar dalam kalkulasi jarak. Selain itu, metode NN dapat memperbesar terjadinya kres antar dua garis dalam satu anak rantai. Sedangkan pada metode FTN, pembentukan rute dengan mengisi *node* dari dua arah (kanan atau kiri) menjadikan metode ini dapat membentuk rute dengan cycle yang lebh baik dari NN.

Pada perjalanan testing metode FTN, penyusun menemukan faktor lain yang dapat membuat solusi FTN bisa lebih baik. Pada kenyataannya, baik dalam metode NN maupun FTN diatas, tidak mempertimbangkan pesebaran geografis *nodes* berada pada algoritmanya. Padahal letak geografis dari *nodes* yang didapatkan dari koordinat x dan y, menjadi sebuah faktor penentuan batas pencarian *nodes* selanjutnya yang akan masuk ke dalam rute. Penyusun membuat beberapa algoritma yang mempertimbangkan letak geografis *nodes*; FTN (FTNs) yang tidak mempertimbangkan pesebaran *node*, FTN (FTNx) yang terbagi dalam *cluster*: kanan dan kiri (koordinat x dari depot pusat menjadi batas *nodes* dalam pembentukan rute), serta FTN (FTNy) yang terbagi dalam *cluster*: atas dan bawah (koordinat y dari depot pusat menjadi batas *nodes* dalam pembentukan rute).

Sebagai contoh, *study case* in3 (*nodes* tujuan berjumlah 25), jarak tempuh total yang dihasilkan oleh FTNs adalah 745,46 sedangkan pada FTNx 708,41. Terbukti bahwa batas koordinat x depot pusat memiliki pengaruh positif terhadap total jarak tempuh sebuah rute, jika dibandingkan dengan tidak memperhitungkan pesebaran *nodes*. Namun, tidak selamanya hal tersebut memberikan hal positif, ada kalanya ketika memperhitungkan batas koordinat depot pusat, total jarak tempuh yang dihasilkan lebih besar. Hal ini terlihat ketika membandingkan total jarak dari FTNs dengan FTNy. Seperti yang disebutkan sebelumnya, FTNs menghasilkan jarak 745,46 sedangkan pada FTNy 758,45. Dari hasil tersebut terlihat bahwa batas koordinat y depot pusat tidak memberi pengaruh positif terhadap total jarak tempuh sebuah rute dibandingkan dengan tidak memperhitungkan pesebaran *nodes*.

Dari ketiga skenario diatas terlihat bahwa FTNx memiliki solusi inisiasi yang paling baik. Namun FTNx belum tentu selalu lebih baik dari FTNs dan FTNy. Hal tersebut dibuktikan ketika penyusun *running* program untuk lima belas kasus. Adakalanya FTNs menjadi solusi yang terbaik begitupula dengan FTNy. Hasil antara ketiganya dapat dikatakan bergantung kepada kecendrungan posisi dimana *nodes* berada, pada hal ini penyusun hanya menggolongkan kecendrungan tersebut pada batas

koordinat x dan y. Batas tersebut akan mempengaruhi rute yang terbentuk dari persoalan yang ada, yang dapat dilihat pada Gambar 2.2. Oleh karena itu, penyusun membuat sebuah algoritma utuh yang membandingkan performansi yang dihasilkan oleh ketiga algoritma tersebut. Dimana akan menampilkan solusi terbaik diantara ketiganya. Dari hal ini, diharapkan eksperimen selanjutnya dapat meneliti mengenai koordinat batas *nodes* yang lebih optimum sebagai batas rute *nodes*.



Gambar 2.3. Perbedaan rute yang dihasilkan oleh FTNs, FTNx, dan FTNy

Untuk mempermudah pemahaman algoritma FTN secara utuh dibuat sebuah *flowchart* yang dapat dilihat pada lampiran.

III. PEMBAHASAN

Untuk dapat mengetahui performa metode FTN, dilakukanlah pengujian metode terhadap 15 kasus VRP. Kelima belas kasus tersebut memiliki *nodes* yang menggambarkan kota dengan permintaan masing-masing yang harus dikunjungi oleh kendaraan dengan kapasitas tertentu yang berasal dari depot. Kelima belas kasus tersebut memiliki lokasi depot, jumlah dan lokasi *nodes*, permintaan *nodes*, serta kapasitas kendaraan yang beragam. Lokasi depot dan nodes dideskripsikan dengan koordinat yang berdasarkan pada koordinat kartesian. Penentuan nilai variabel-variabel dalam setiap kasus ditentukan secara acak. Kelima belas kasus tersebut dapat dilihat pada lampiran bagian *study cases*.

Tahapan penentuan solusi awal dari VRP dengan menggunakan metode FTN dilakukan dengan mencari opsi rute dengan pendekatan FTNs, FTNx, dan FTNy dan dilanjutkan dengan melakukan perbandingan diantara ketiganya. Pencarian opsi rute dengan masing-masing pendekatan terbagi ke dalam 3 tahap utama, yaitu tahap inisiasi, iterasi, dan terminasi. Tahapan detail metode FTN tersebut dijelaskan dengan menggunakan kasus 2 sebagai contoh. Berikut merupakan spesifikasi dari kasus 2 dengan depot tampilkan sebagai kota 0.

Tabel 3.1. Spesifikasi Kasus 2

Kapasitas Kendaraan = 10											
Kota	X	Y	*Demand*	Kota	X	Y	*Demand*	Kota	X	Y	*Demand*
0	-5	5	0	6	20	6	5	12	3	3	4
1	8	-8	2	7	-23	2	5	13	4	4	1
2	2	0	5	8	6	-30	4	14	-3	-3	1
3	-2	-10	5	9	8	7	2	15	-2	-2	1
4	8	-10	4	10	1	1	5	16	-1	-1	1
5	41	1	2	11	2	2	5				

Berikut adalah tahapan penentuan solusi awal VRP pada kasus 3 dengan metode FTN.

1. Pencarian opsi rute dengan pendekatan FTNs

- Inisiasi

  Pada tahap ini, dilakukan pembentukan matriks yang menggambarkan jarak antar node. Pada kasus 2, matriks antar node yang dihasilkan dapat dilihat pada Tabel 3.2.

Tabel 3.2. Matriks Jarak Antar Node

Kota	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0,0	18,4	8,6	15,3	19,8	46,2	25,0	18,2	36,7	13,2	7,2	7,6	8,2	9,1	8,2	7,6	7,2
1	18,4	0,0	10,0	10,2	2,0	34,2	18,4	32,6	22,1	15,0	11,4	11,7	12,1	12,6	12,1	11,7	11,4
2	8,6	10,0	0,0	10,8	11,7	39,0	19,0	25,1	30,3	9,2	1,4	2,0	3,2	4,5	5,8	4,5	3,2

3	15,3	10,2	10,8	0,0	10,0	44,4	27,2	24,2	21,5	19,7	11,4	12,6	13,9	15,2	7,1	8,0	9,1
4	19,8	2,0	11,7	10,0	0,0	34,8	20,0	33,2	20,1	17,0	13,0	13,4	13,9	14,6	13,0	12,8	12,7
5	46,2	34,2	39,0	44,4	34,8	0,0	21,6	64,0	46,8	33,5	40,0	39,0	38,1	37,1	44,2	43,1	42,0
6	25,0	18,4	19,0	27,2	20,0	21,6	0,0	43,2	38,6	12,0	19,6	18,4	17,3	16,1	24,7	23,4	22,1
7	18,2	32,6	25,1	24,2	33,2	64,0	43,2	0,0	43,2	31,4	24,0	25,0	26,0	27,1	20,6	21,4	22,2
8	36,7	22,1	30,3	21,5	20,1	46,8	38,6	43,2	0,0	37,1	31,4	32,2	33,1	34,1	28,5	29,1	29,8
9	13,2	15,0	9,2	19,7	17,0	33,5	12,0	31,4	37,1	0,0	9,2	7,8	6,4	5,0	14,9	13,5	12,0
10	7,2	11,4	1,4	11,4	13,0	40,0	19,6	24,0	31,4	9,2	0,0	1,4	2,8	4,2	5,7	4,2	2,8
11	7,6	11,7	2,0	12,6	13,4	39,0	18,4	25,0	32,2	7,8	1,4	0,0	1,4	2,8	7,1	5,7	4,2
12	8,2	12,1	3,2	13,9	13,9	38,1	17,3	26,0	33,1	6,4	2,8	1,4	0,0	1,4	8,5	7,1	5,7
13	9,1	12,6	4,5	15,2	14,6	37,1	16,1	27,1	34,1	5,0	4,2	2,8	1,4	0,0	9,9	8,5	7,1
14	8,2	12,1	5,8	7,1	13,0	44,2	24,7	20,6	28,5	14,9	5,7	7,1	8,5	9,9	0,0	1,4	2,8
15	7,6	11,7	4,5	8,0	12,8	43,1	23,4	21,4	29,1	13,5	4,2	5,7	7,1	8,5	1,4	0,0	1,4
16	7,2	11,4	3,2	9,1	12,7	42,0	22,1	22,2	29,8	12,0	2,8	4,2	5,7	7,1	2,8	1,4	0,0

- Iterasi 1
  - Dilakukan pencarian titik terjauh dari titik pusat. Node 5 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 5, dimana sisa kapasitas yang tersedia adalah 8 (Kapasitas – Permintaan Node 5 = 10 – 2 = 8). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 5.
  - Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 5. Titik tersebut adalah Node 6. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 6. Karena sisa kapasitas yang tersedia bernilai 3 (8-5=3), Node 6 diletakkan pada ruang node di sisi Node 5.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 5 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 5 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 5). Node yang terdekat selanjutnya adalah Node 9. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 1 (3-2=1), Node 9 diletakkan pada ruang node sisi Node 5.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 6 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 6 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 6). Node yang terdekat selanjutnya adalah Node 13. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 0 (1-1=0), Node 13 diletakkan pada ruang node sisi Node 6.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 9 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 9 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 9). Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
  - Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 1 berdasarkan node-node pengisi yang ada, yaitu 9-5-6-13.
- Iterasi 2
  - Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 8 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 8, dimana sisa kapasitas yang tersedia adalah 6 (Kapasitas – Permintaan Node 8 = 10 – 4 = 6). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 8.
  - Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 8. Titik tersebut adalah Node 4. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 4. Karena sisa kapasitas yang tersedia bernilai 2 (8-4=2), Node 4 diletakkan pada ruang node di sisi Node 6.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 8 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 8 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 8). Node yang terdekat selanjutnya adalah Node 3. Namun, karena kapasitas yang tersisa lebih sedikit bila dibandingkan dengan permintaan yang ada, maka Node 3 tidak dimasukkan ke dalam ruang pada anak rantai. Selanjutnya, dilakukan pencarian node terdekat lainnya dengan berurutan

berdasarkan kedekatan terhadap Node 8 hingga menemukan node terdekat ke-n yang memiliki permintaan ≤ sisa kapasitas kendaraan yang tersedia (dalam tahap ini, node pusat tidak diperhitungkan). Node terdekat dari Node 8 yang memenuhi persyaratan tersebut adalah Node 1. Karena sisa kapasitas yang tersedia bernilai 0 (2-2=0), Node 1 diletakkan pada ruang node sisi Node 8.

- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 4 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 4 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 4). Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 2 berdasarkan node-node pengisi yang ada, yaitu 1-8-4.

- Iterasi 3

- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 7 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 7, dimana sisa kapasitas yang tersedia adalah 5 (Kapasitas – Permintaan Node 7 = 10 – 5 = 5). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 7.
- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 7. Titik tersebut adalah Node 14. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 14. Karena sisa kapasitas yang tersedia bernilai 4 (5-1=4), Node 14 diletakkan pada ruang node di sisi Node 7.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 7 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 7 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 7). Node yang terdekat selanjutnya adalah Node 15. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 3 (4-1=3), Node 15 diletakkan pada ruang node sisi Node 7.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 14 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 14 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 14). Node yang terdekat selanjutnya adalah Node 16. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 2 (3-1=2), Node 16 diletakkan pada ruang node sisi Node 14.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 15 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 15 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 15). Node yang terdekat selanjutnya adalah Node 10. Namun, karena kapasitas yang tersisa lebih sedikit bila dibandingkan dengan permintaan yang ada, maka Node 10 tidak dimasukkan ke dalam ruang pada anak rantai. Selanjutnya, dilakukan pencarian node terdekat lainnya dengan berurutan berdasarkan kedekatan terhadap Node 15 hingga menemukan node terdekat ke-n yang memiliki permintaan ≤ sisa kapasitas kendaraan yang tersedia (dalam tahap ini, node pusat tidak diperhitungkan). Apabila terdapat node yang memenuhi persyaratan tersebut, maka node tersebut akan dimasukkan kedalam ruang di sebelah ruang Node 15. Kemudian dilakukan kembali komparasi jarak terjauh antara node pusat dengan noda di kedua ujung rantai dan dilanjutkan dengan pencarian node yang terdekat dari node terjauh tersebut dengan cara yang sama. Namun, karena tidak ditemukan node yang memenuhi persyaratan tersebut, maka ruang kosong dirantai tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 3 berdasarkan node-node pengisi yang ada, yaitu 15-7-14-16.

- Iterasi 4

- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 3 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 3, dimana sisa kapasitas yang tersedia adalah 5 (Kapasitas – Permintaan Node 3 = 10 – 5 = 5). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 3.

- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 3. Titik tersebut adalah Node 2. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 2. Karena sisa kapasitas yang tersedia bernilai 0 (5-5=0), Node 2 diletakkan pada ruang node di sisi Node 3.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 3 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 3 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 2). Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 4 berdasarkan node-node pengisi yang ada, yaitu 3-2.

- Iterasi 5
- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 12 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 12, dimana sisa kapasitas yang tersedia adalah 6 (Kapasitas – Permintaan Node 12 = 10 – 4 = 6). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 12.
- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 12. Titik tersebut adalah Node 11. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 11. Karena sisa kapasitas yang tersedia bernilai 1 (6-5=1), Node 11 diletakkan pada ruang node di sisi Node 12.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 12 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 12 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 12). Node yang terdekat selanjutnya adalah Node 10. Namun, karena kapasitas yang tersisa lebih sedikit bila dibandingkan dengan permintaan yang ada, maka Node 10 tidak dimasukkan ke dalam ruang pada anak rantai. Selanjutnya, karena sudah tidak ada lagi node selain Node 10 yang belum digunakan, maka tidak dapat dilakukan pencarian titik lainnya yang memenuhi persyaratan tersebut. Oleh karena itu, ruang kosong dirantai tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 5 berdasarkan node-node pengisi yang ada, yaitu 12-11.

- Iterasi 6
- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Karena hanya Node 10 yang belum digunakan, maka didefinisikan Anak Rantai 6, yaitu 10.

- Terminasi

  Tahap terminasi dilakukan karena setiap node telah termasuk ke dalam semua anak rantai yang terbentuk. Tahap ini dilakukan dengan menyusun seluruh anak rantai menjadi sebuah rantai panjang dimana titik pusat berperan sebagai titik awal, titik akhir, dan penghubung antar anak rantai. Melalui tahap terminasi ini, dihasilkan opsi rute FTNs pada kasus 2 VRP, yaitu 0-9-5-6-13-0-1-8-4-0-15-7-14-16-0-3-2-0-12-11-0-10-0.

Gambar 3.2 memperlihatkan visualisasi dari tahapan iterasi dan terminasi pendekatan FTNs untuk menghasilkan opsi rute bagi kasus 2 permasalahan VRP.

2. Pencarian opsi rute dengan pendekatan FTNx

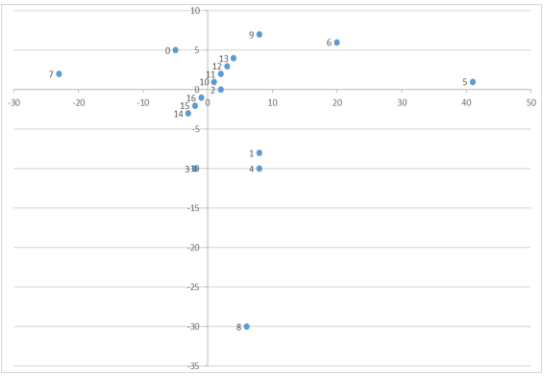
- Inisiasi

  Pada tahap ini, dilakukan pembentukan matriks yang menggambarkan jarak antar node seperti tahap inisiasi pendekatan FTNs yang terlihat pada Tabel 3.2. Selain itu, dilakukan pula pembentukan garis batas $X=X_0$ yang membagi nodes ke dalam 2 klaster: kanan dan kiri. Dalam kasus 2, garis batas tersebut bernilai X= -5.

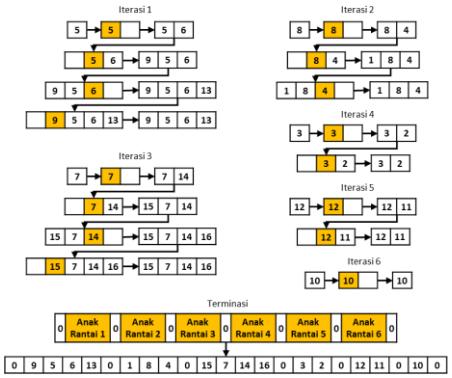
- Iterasi 1
- Dilakukan pencarian titik terjauh dari titik pusat. Node 5 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 5, dimana sisa kapasitas yang

tersedia adalah 8 (Kapasitas – Permintaan Node 5 = 10 – 2 = 8). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 5. Karena Node 5 berada pada klaster wilayah kanan, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 1 hanyalah node yang berada pada klaster wilayah kanan.

- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 5. Titik tersebut adalah Node 6. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 6. Karena sisa kapasitas yang tersedia bernilai 3 (8-5=3), Node 6 diletakkan pada ruang node di sisi Node 5.



Gambar 3.1. Visualisasi Lokasi *Nodes*



Gambar 3.2. Visualisasi Tahapan FTNs

- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 5 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 5 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 5). Node yang terdekat selanjutnya adalah Node 9. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 1 (3-2=1), Node 9 diletakkan pada ruang node sisi Node 5.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 6 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 6 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 6). Node yang terdekat selanjutnya adalah Node 13. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 0 (1-1=0), Node 13 diletakkan pada ruang node sisi Node 6.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 9 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 9 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 9). Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 1 berdasarkan node-node pengisi yang ada, yaitu 9-5-6-13.

• Iterasi 2

- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 8 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 8, dimana sisa kapasitas yang tersedia adalah 6 (Kapasitas – Permintaan Node 8 = 10 – 4 = 6). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 8. Karena Node 8 berada pada klaster wilayah kanan, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 2 hanyalah node yang berada pada klaster wilayah kanan.
- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 8. Titik tersebut adalah Node 4. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 4. Karena sisa kapasitas yang tersedia bernilai 2 (6-4=2), Node 4 diletakkan pada ruang node di sisi Node 6.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 8 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 8 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 8). Node yang terdekat selanjutnya adalah Node 3. Namun, karena kapasitas yang tersisa lebih sedikit bila dibandingkan dengan permintaan yang ada, maka Node 3 tidak dimasukkan ke dalam ruang pada

anak rantai. Selanjutnya, dilakukan pencarian node klaster wilayah kanan terdekat lainnya dengan berurutan berdasarkan kedekatan terhadap Node 8 hingga menemukan node terdekat ke-n yang memiliki permintaan ≤ sisa kapasitas kendaraan yang tersedia (dalam tahap ini, node pusat tidak diperhitungkan). Node terdekat dari Node 8 yang memenuhi persyaratan tersebut adalah Node 1. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 0 (2-2=0), Node 1 diletakkan pada ruang node sisi Node 8.

- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 4 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 4 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 4). Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 2 berdasarkan node-node pengisi yang ada, yaitu 1-8-4.

- Iterasi 3
  - Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 7 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 7, dimana sisa kapasitas yang tersedia adalah 5 (Kapasitas – Permintaan Node 7 = 10 – 5 = 5). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 7. Karena Node 7 berada pada klaster wilayah kiri, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 3 hanyalah node yang berada pada klaster wilayah kiri. Namun, karena seluruh node di klaster kiri telah digunakan ruang kosong pada rantai tersebut dihilangkan.
  - Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 3 berdasarkan node-node pengisi yang ada, yaitu 7.
- Iterasi 4
  - Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 3 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 3, dimana sisa kapasitas yang tersedia adalah 5 (Kapasitas – Permintaan Node 7 = 10 – 5 = 5). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 3. Karena Node 3 berada pada klaster wilayah kanan, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 4 hanyalah node yang berada pada klaster wilayah kanan.
  - Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 3. Titik tersebut adalah Node 14. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 14. Karena sisa kapasitas yang tersedia bernilai 4 (5-1=4), Node 14 diletakkan pada ruang node di sisi Node 3.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 3 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 3 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 3). Node yang terdekat selanjutnya adalah Node 15. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 3 (4-1=3), Node 15 diletakkan pada ruang node sisi Node 3.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 14 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 14 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 14). Node yang terdekat selanjutnya adalah Node 16. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 2 (3-1=2), Node 16 diletakkan pada ruang node sisi Node 14.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 15 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 15 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 15). Node yang terdekat selanjutnya adalah Node 10. Namun, karena kapasitas yang tersisa lebih sedikit bila dibandingkan dengan permintaan yang ada, maka Node 10 tidak dimasukkan ke dalam ruang pada anak rantai. Selanjutnya, dilakukan pencarian node klaster kanan terdekat lainnya dengan berurutan berdasarkan kedekatan terhadap Node 15 hingga menemukan node terdekat ke-n yang memiliki permintaan ≤ sisa kapasitas kendaraan yang tersedia (dalam tahap ini, node pusat tidak diperhitungkan). Apabila terdapat node di wilayah klaster kanan yang memenuhi persyaratan

tersebut, maka node tersebut akan dimasukkan kedalam ruang di sebelah ruang Node 15. Kemudian dilakukan kembali komparasi jarak terjauh antara node pusat dengan noda di kedua ujung rantai dan dilanjutkan dengan pencarian node yang terdekat dari node terjauh tersebut dengan cara yang sama. Namun, karena tidak ditemukan node yang memenuhi persyaratan tersebut, maka ruang kosong dirantai tersebut dihilangkan.

- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 4 berdasarkan node-node pengisi yang ada, yaitu 15-3-14-16.

- Iterasi 5
  - Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 2 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 2, dimana sisa kapasitas yang tersedia adalah 5 (Kapasitas – Permintaan Node 2 = 10 – 5 = 5). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 2. Karena Node 2 berada pada klaster wilayah kanan, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 5 hanyalah node yang berada pada klaster wilayah kanan.
  - Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 2. Titik tersebut adalah Node 10. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 10. Karena sisa kapasitas yang tersedia bernilai 0 (5-5=0), Node 10 diletakkan pada ruang node di sisi Node 2.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 2 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 2. Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
  - Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 5 berdasarkan node-node pengisi yang ada, yaitu 2-10.
- Iterasi 6
  - Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 12 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 12, dimana sisa kapasitas yang tersedia adalah 6 (Kapasitas – Permintaan Node 12 = 10 – 4 = 6). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 12. Karena Node 12 berada pada klaster wilayah kanan, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 1 hanyalah node yang berada pada klaster wilayah kanan.
  - Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 12. Titik tersebut adalah Node 11. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 11. Karena sisa kapasitas yang tersedia bernilai 1 (6-5=1), Node 11 diletakkan pada ruang node di sisi Node 12.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 12 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 12. Karena seluruh node telah digunakan, ruang kosong dirantai tersebut dihilangkan.
  - Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 6 berdasarkan node-node pengisi yang ada, yaitu 12-11.
- Terminasi

  Tahap terminasi dilakukan karena setiap node telah termasuk ke dalam semua anak rantai yang terbentuk. Tahap ini dilakukan dengan menyusun seluruh anak rantai menjadi sebuah rantai panjang dimana titik pusat berperan sebagai titik awal, titik akhir, dan penghubung antar anak rantai. Melalui tahap terminasi ini, dihasilkan opsi rute FTNx pada kasus 2 VRP, yaitu 0-9-5-6-13-0-1-8-4-0-7-0-15-3-14-16-0-2-10-0-12-11-0.

Gambar 3.4 memperlihatkan visualisasi dari tahapan iterasi dan terminasi pendekatan FTNx untuk menghasilkan opsi rute bagi kasus 2 permasalahan VRP.

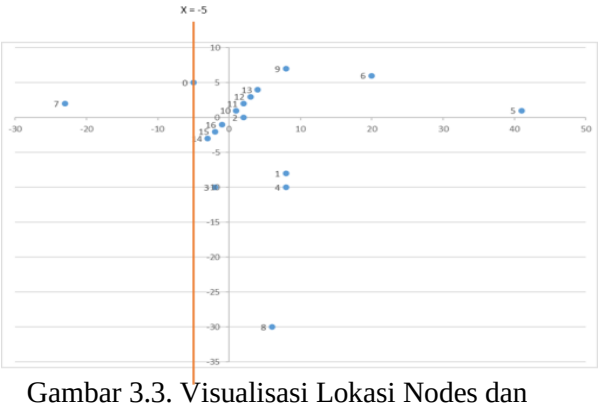
3. Pencarian opsi rute dengan pendekatan FTNy

- Inisiasi

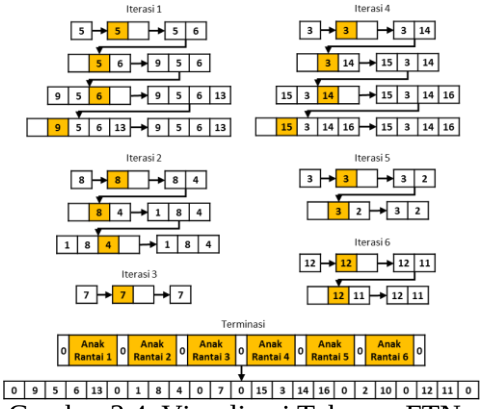
Pada tahap ini, dilakukan pembentukan matriks yang menggambarkan jarak antar node seperti tahap inisiasi pendekatan FTNs yang terlihat pada Tabel 3.2. Selain itu, dilakukan pula pembentukan garis batas $Y=Y_0$ yang membagi nodes ke dalam 2 klaster: atas dan bawah. Dalam kasus 2, garis batas tersebut bernilai Y=5.

- Iterasi 1

- Dilakukan pencarian titik terjauh dari titik pusat. Node 5 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 5, dimana sisa kapasitas yang tersedia adalah 8 (Kapasitas – Permintaan Node 5 = 10 – 2 = 8). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 5. Karena Node 5 berada pada klaster wilayah bawah, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 1 hanyalah node yang berada pada klaster wilayah bawah.



Gambar 3.3. Visualisasi Lokasi Nodes dan Garis Batas X



Gambar 3.4. Visualisasi Tahapan FTNx

- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 5. Titik tersebut adalah Node 1. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 1. Karena sisa kapasitas yang tersedia bernilai 6 (8-2=6), Node 1 diletakkan pada ruang node di sisi Node 5.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 5 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 5 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 5). Node yang terdekat selanjutnya adalah Node 4. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 2 (6-4=2), Node 4 diletakkan pada ruang node sisi Node 5.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 4 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 4 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 4). Node yang terdekat selanjutnya adalah Node 3. Namun, karena kapasitas yang tersisa lebih sedikit bila dibandingkan dengan permintaan yang ada, maka Node 3 tidak dimasukkan ke dalam ruang pada anak rantai. Selanjutnya, dilakukan pencarian node klaster wilayah bawah terdekat lainnya dengan berurutan berdasarkan kedekatan terhadap Node 4 hingga menemukan node terdekat ke-n yang memiliki permintaan ≤ sisa kapasitas kendaraan yang tersedia (dalam tahap ini, node pusat tidak diperhitungkan). Node terdekat dari Node 4 yang memenuhi persyaratan tersebut adalah Node 16. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 1 (2-1=1), Node 16 diletakkan pada ruang node sisi Node 4.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 1 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 1 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 1).
- Node yang terdekat selanjutnya adalah Node 2. Namun, karena kapasitas yang tersisa lebih sedikit bila dibandingkan dengan permintaan yang ada, maka Node 2 tidak dimasukkan ke dalam ruang pada anak rantai. Selanjutnya, dilakukan pencarian node klaster wilayah bawah terdekat lainnya dengan berurutan berdasarkan kedekatan terhadap Node 1 hingga menemukan node

terdekat ke-n yang memiliki permintaan ≤ sisa kapasitas kendaraan yang tersedia (dalam tahap ini, node pusat tidak diperhitungkan). Node terdekat dari Node 1 yang memenuhi persyaratan tersebut adalah Node 15. Karena sisa kapasitas yang tersedia bila melalui node tersebut bernilai 0 (1-1=0), Node 15 diletakkan pada ruang node sisi Node 1.

- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 15 memiliki jarak terjauh, maka dilakukan pembuatan ruang node di sisi Node 15 dan pencarian titik yang belum digunakan diilihat kedekatannya berdasarkan titik terjauh tersebut (Node 15). Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 1 berdasarkan node-node pengisi yang ada, yaitu 16-4-5-1-15.

- Iterasi 2

- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 8 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 8, dimana sisa kapasitas yang tersedia adalah 6 (Kapasitas – Permintaan Node 8 = 10 – 4 = 6). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 8. Karena Node 8 berada pada klaster wilayah bawah, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 2 hanyalah node yang berada pada klaster wilayah bawah.
- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 8. Titik tersebut adalah Node 3. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 3. Karena sisa kapasitas yang tersedia bernilai 1 (6-5=1), Node 3 diletakkan pada ruang node di sisi Node 8.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 8 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 8 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 8). Node yang terdekat selanjutnya adalah Node 14. Karena sisa kapasitas yang tersedia bila melalui titik tersebut bernilai 0 (5-5=0), Node 14 diletakkan pada ruang node di sisi Node 8.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 3 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 3. Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 2 berdasarkan node-node pengisi yang ada, yaitu 14-8-3.

- Iterasi 3

- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 6 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 6, dimana sisa kapasitas yang tersedia adalah 5 (Kapasitas – Permintaan Node 2 = 10 – 5 = 5). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 6. Karena Node 6 berada pada klaster wilayah atas, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 3 hanyalah node yang berada pada klaster wilayah atas.
- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 6. Titik tersebut adalah Node 9. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 9. Karena sisa kapasitas yang tersedia bernilai 3 (5-2=3), Node 9 diletakkan pada ruang node di sisi Node 6.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 6 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 2. Namun, karena seluruh node yang berada pada klaster wilayah atas telah digunakan, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 5 berdasarkan node-node pengisi yang ada, yaitu 6-9.

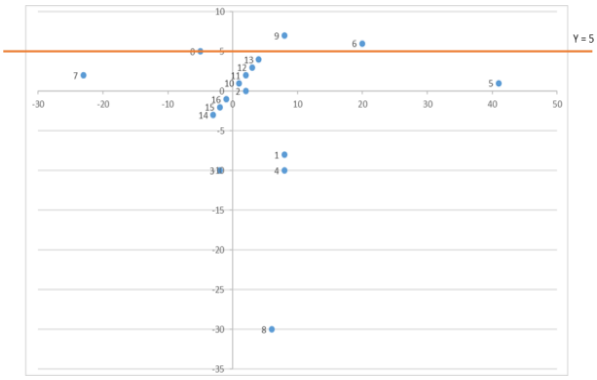
- Iterasi 4

- Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 7 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 7, dimana sisa kapasitas yang tersedia adalah 5 (Kapasitas – Permintaan Node 7 = 10 – 5 = 5). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 7. Karena Node 7 berada pada klaster wilayah bawah, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 4 hanyalah node yang berada pada klaster wilayah bawah.
- Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 7. Titik tersebut adalah Node 10. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 10. Karena sisa kapasitas yang tersedia bernilai 0 (5-5=0), Node 10 diletakkan pada ruang node di sisi Node 7.
- Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 7 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 7. Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
- Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 4 berdasarkan node-node pengisi yang ada, yaitu 7-10.

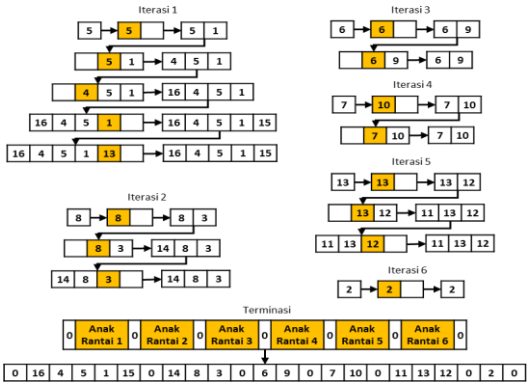
- Iterasi 5
  - Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Node 13 memiliki titik terjauh dari pusat, sehingga pencarian selanjutnya dimulai dari titik ini. Kemudian, dilakukan perhitungan sisa kapasitas yang tersedia akibat mengangkut beban pada Node 13, dimana sisa kapasitas yang tersedia adalah 9 (Kapasitas – Permintaan Node 13 = 10 – 1 = 9). Lalu, dilakukan pembuatan ruang node di sisi kanan Node 13. Karena Node 13 berada pada klaster wilayah bawah, node-node lain yang dapat mengisi anak rantai yang terbentuk pada iterasi 5 hanyalah node yang berada pada klaster wilayah bawah.
  - Selanjutnya dilakukan pencarian titik terdekat yang belum digunakan dari Node 13. Titik tersebut adalah Node 12. Kemudian, dilakukan kembali perhitungan sisa kapasitas apabila memilih Node 12. Karena sisa kapasitas yang tersedia bernilai 5 (9-4=5), Node 12 diletakkan pada ruang node di sisi Node 13.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 13 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 13 dan pencarian titik terdekat yang belum digunakan dari titik terjauh tersebut (Node 13). Node yang terdekat selanjutnya adalah Node 11. Karena sisa kapasitas yang tersedia bila melalui titik tersebut bernilai 0 (5-5=0), Node 11 diletakkan pada ruang node di sisi Node 13.
  - Dilakukan komparasi jarak antara node pusat dengan node pada kedua ujung anak rantai. Karena Node 12 memiliki jarak terjauh, maka dilakukan pembuatan ruang node disisi Node 12. Namun, karena sisa kapasitas telah habis atau bernilai 0, ruang kosong yang terbentuk tersebut dihilangkan.
  - Ketika tidak terdapat ruang kosong di kedua ujung anak rantai, maka didefinisikan Anak Rantai 2 berdasarkan node-node pengisi yang ada, yaitu 11-13-12.
- Iterasi 6
  - Dilakukan pencarian titik terjauh diantara titik-titik yang belum terpakai dari titik pusat. Karena Node 2 menjadi satu-satunya node yang belum digunakan, maka didefinisikan Anak Rantai 6, yaitu 2.
- Terminasi

  Tahap terminasi dilakukan karena setiap node telah termasuk ke dalam semua anak rantai yang terbentuk. Tahap ini dilakukan dengan menyusun seluruh anak rantai menjadi sebuah rantai panjang dimana titik pusat berperan sebagai titik awal, titik akhir, dan penghubung antar anak rantai. Melalui tahap terminasi ini, dihasilkan opsi rute FTNy pada kasus 2 VRP, yaitu 0-16-4-5-1-15-0-14-8-3-0-6-9-0-7-10- 0-11-13-12-0-2-0.

Gambar 3.6 memperlihatkan visualisasi dari tahapan iterasi dan terminasi pendekatan FTNy untuk menghasilkan opsi rute bagi kasus 2 permasalahan VRP.



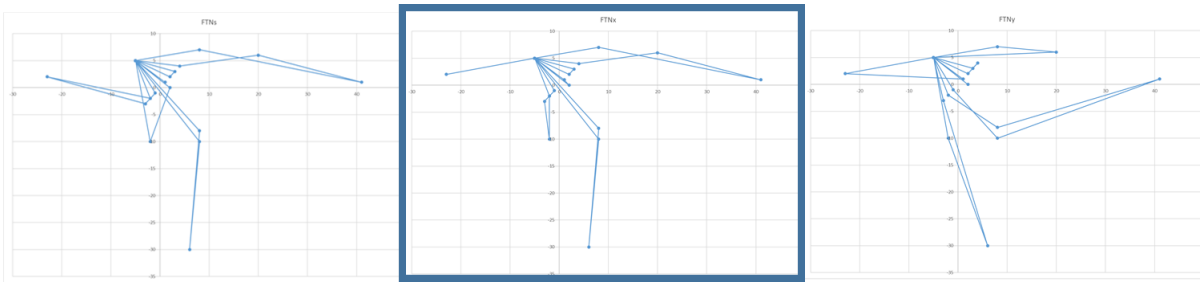
Gambar 3.5. Visualisasi Lokasi Nodes dan Garis Batas Y



Gambar 3.6. Visualisasi Tahapan FTNy

4. Perbandingan opsi rute dan penetapan solusi inisial FTN
   Setelah terbentuk ketiga opsi rute melalui pendekatan FTNs, FTNx, dan FTNy, dilakukan perhitungan total jarak ketiga opsi rute tersebut. Berdasarkan hasil perhitungan, rute FTNs memiliki total jarak 299,902100, FTNx memiliki total jarak 277,612396, dan FTNy memiliki total jarak 318,755463. Setelah itu, dilakukan perbandingan nilai total jarak dari ketiga opsi dan dipilih opsi rute dengan total jarak terpendek. Oleh karena itu, rute FTNx (0-9-5-6-13-0-1-8-4-0-7-0-15-3-14-16-0-2-10-0-12-11-0) dengan total jarak 277,612396 dipilih sebagai solusi inisial metode FTN dari kasus 2 VRP.

Gambar 3.7 memperlihatkan visualisasi opsi rute FTNs, FTNx, FTNy dan solusi inisial FTN yang terbentuk untuk menyelesaikan permasalahan pada kasus 2 VRP.

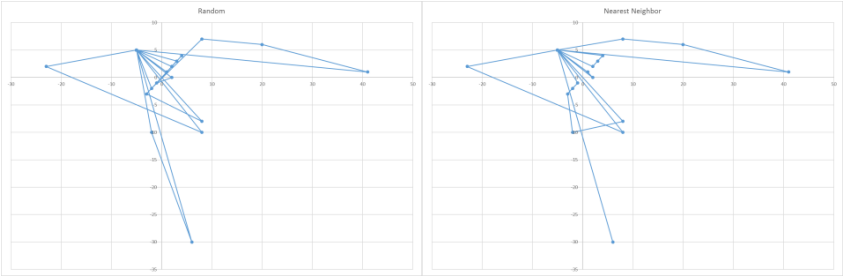


Gambar 3.7. Visualisasi Solusi Inisial FTN pada Kasus 2 VRP

Kelima belas kasus VRP yang ada diolah menggunakan *executable source code* yang telah dibuat. *Source code* metode FTN tersebut dapat dilihat pada lampiran bagian *source code*. Tahap menjalankan *executable source code* tersebut dijelaskan dalam lampiran bagian *manual book*. Hasil *running* program untuk kelima belas studi kasus tersebut diperlihatkan dalam lampiran berupa *running program*.Pada bagian tersebut, dapat pula terlihat solusi initial berupa rute dan total jarak yang dihasilkan.

Untuk melihat performa dari metode FTN ini, dilakukan pencarian solusi awal dari 15 kasus VRP tersebut dengan metode sebelumnya, yaitu *random* dan NN. Pencarian solusi awal tersebut dilakukan menggunakan *executable source code* yang juga telah dibuat. Hasil solusi awal dari metode FTN kemudian dibandingkan dengan hasil solusi awal yang didapatkan melalui kedua metode tersebut.

Pada contoh kasus 2 VRP, metode random menghasilkan solusi awal 0-2-16-13-14-1-0- 5-6-9-15-0-8-3-0-10-12-0-7-4-0-11-0 yang memiliki total jarak 338,456787. Sedangkan metode NN menghasilkan solusi awal 0-10-2-0-16-15-14-3-1-0-11-12-13-0-9-6-5-0-7-4-0-8-0 yang memiliki total jarak 320,090790. Visualisasi solusi inisial dari kedua metode tersebut dapat terlihat pada Gambar 3.8. Berdasarkan hasil tersebut, terlihat bahwa metode FTN dapat menghasilkan solusi awal dengan total jarak yang lebih kecil bila dibandingkan dengan kedua metode lainnya.



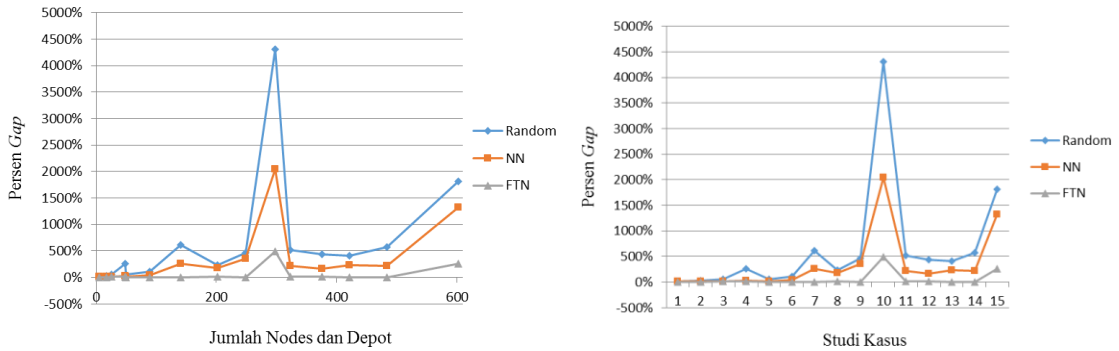
Gambar 3.8. Visualisasi Solusi Inisial dengan Metode *Random* dan *Nearest Neighbor*

Rangkuman dari hasil *running* program untuk menghasilkan solusi inisial dari 15 studi kasus VRP menggunakan metode *random*, NN, dan FTN dapat dilihat pada Tabel 3.3 Hasil *running* tersebut memperlihatkan total jarak yang dihasilkan dari rute yang diperoleh.

Tabel 3.3. Hasil *Running* 15 *Study Cases* dengan Metode *Random*, NN, dan FTN

Study case	**Jumlah *Nodes*+Depot**	**Random**	**NN**	**FTN**
in	5	83,99	82,13	74,90
in2	17	338,46	320,09	277,61
in3	26	935,04	719,32	708,42
in4	48	147.000,67	52.006,22	45.841,71
in5	50	1.568.417,75	1.164.397,88	1.038.446,88
in6	89	4.321,05	2.731,56	2.029,84
in7	141	20.262,85	10.289,80	2.845,39
in8	202	4.419,46	3.711,70	1.498,27
in9	249	4.759,54	3.802,24	908,82
in10	298	49.135,43	23.871,89	6.660,97
in11	323	4.825,87	2.446,51	850,80
in12	376	5.917,86	2.847,68	1.194,05
in13	421	9.710,07	6.528,46	2.013,91
in14	484	8.191,31	3.884,63	1.193,84
in15	602	12.729,46	9.436,83	2.355,43

Berdasarkan hasil tersebut, terlihat bahwa solusi awal yang dihasilkan melalui FTN memiliki nilai total jarak yang lebih kecil. Hal tersebut mengindikasikan bahwa metode FTN memiliki performa yang lebih baik dari kedua metode sebelumnya. Selain itu, dilakukan pula perhitungan persen *gap* antara hasil solusi awal yang dihasilkan dengan metode ini dan solusi yang dihasilkan melalui VRP *solver* yang dikembangkan oleh *Snyder* (2003) dan menerapkan versi randomisasi dari algoritma *Clarke-Wright* untuk VRP. Persen *gap* dihitung sebagai selisih total jarak solusi inisial masing-masing metode (*random*, NN, dan FTN) dengan solusi optimal VRP *Solver* yang dibandingkan terhadap sokusi optimal VRP *Solver* tersebut. Solusi yang dihasilkan melalui VRP solver tersebut dapat dilihat pada lampiran hasil VRP Solver. Gambar 3.9 memperlihatkan grafik persen *gap* dari solusi awal yang dihasilkan melalui metode *random*, NN, dan FTN terhadap solusi VRP *Solver Snyder*.



Gambar 3.9. Grafik persen *gap* berdasarkan jumlah nodes dan depot serta berdasarkan studi kasus

Seperti yang dapat dilihat dalam Gambar 3.9, metode FTN cenderung lebih konsisten dalam memberikan hasil yang mendekati nilai optimal ditandai dengan persen *gap* yang selalu menunjukan nilai yang terkecil dan disusul dengan metode nearest neighbor dan random secara berurutan. Hal tersebut menunjukan bahwa berdasarkan 15 studi kasus yang dilakukan, metode FTN merupakan metode yang terbaik dalam memberikan solusi awal VRP yang paling mendekati hasil optimal yang dapat dicapai bila dibandingkan ketiga metode lainnya. Bahkan dalam kasus tertentu (kasus 7 dan 14), solusi awal yang dihasilkan menggunakan metode FTN memiliki total jarak yang lebih kecil bila dibandingkan dengan total jarak pada solusi optimal VRP *Solver*.

IV. KESIMPULAN

Solusi inisiasi merupakan suatu hal yang menjadi acuan solusi optimum, baik dengan metode *random* ataupun metode NN. Solusi inisiasi yang mendekati optimum diharpakan membuat tahap iterasi menjadi lebih cepat, karena dapat mengurangi jumlah iterasi pada tahap tersebut. Pada metode *random*, sangat terlihat bahwa metode ini sangat lemah dalam konsistensi dan jarang mendapatkan hasil yang lebih baik dari NN. Namun, pada metode NN pun, *node* dengan letak terjauh dari depot pusat diperhitungkan paling akhir. Padahal *node* terjauh mempunyai peranan penting dalam menentukan rute optimal. Selain itu, pada NN, *node* terakhir yang terpilih langsung dihubungkan dengan depot pusatnya. Hal ini membuat rute yang dibentuk NN tidak menjadi *cycle* yang baik karena jarak dari *node* terjauh ke depot pusat dapat dipastikan memboroskan rute. Metode NN kemungkinan akan menjadi lebih baik ketika digabungkan dengan algoritma sorting *node* pada setiap anak rantai yang terpilih. Penggabungan tersebut dapat dijadikan bahan penelitian berikutnya.

Kelemahan dari metode *random* dan NN telah melahirkan metode FTN yang mana mengutamakan faktor jarak node terjauh dan tanpa menggunakan algotirma sorting sudah dapat membentuk *cycle* yang lebih baik. Walaupun pada kenyataannya, FTN masih belum bisa mengatasi *kress* yang terjadi antara dua garis dalam satu rute. Hal tersebut dapat menjadi bahan pembelajaran untuk penelitian selanjutnya. FTN juga memperhitungkan letak geografis dari *nodes* yang berada dalam himpunan, yang dikelompokan dalam FTNs (tidak mempertimbangkan pesebaran *node)*, FTNx (koordinat x dari depot pusat menjadi batas *nodes* dalam pembentukan rute), serta FTNy (koordinat y dari depot pusat menjadi batas *nodes* dalam pembentukan rute). Namun, pengelompokan tersebut hanya berdasarkan koordinat x dan y pada depot pusat saja. Padahal koordinat penentu pengelompokan *nodes* sangat mempengaruhi hasil yang didapatkan. Hal ini juga dapat dijadikan sebagai bahan perbaikan pada penelitian selanjutnya.

Dari segala kekurangan dan kelebihan yang diberikan oleh metode *random*, NN, dan FTN, jika dibandingkan, terlihat bahwa metode FTN memberikan performansi terbaik pada 15 *study cases* pengujian. Bahkan metode FTN memberikan perbedaan yang sangat signifikan pada *cases* dengan jumlah *nodes* diatas 100. Hal tersebut dapat memberikan kesimpulan bahwa *nodes* dengan jarak terjauh, geografis, serta pembentukan rute *cycling* merupakan faktor yang berpengaruh dalam pembentukan solusi inisiasi yang lebih baik.

Program yang dirancang oleh penyusun merupakan program yang hanya dapat mengatasi persoalan dengan jumlah nodes tujuan maksimum sebesar 1000 nodes. Selain itu, demand dari setiap nodes tidak boleh melebihi kapasitas maksimum yang ada atau dengan kata lain, setiap nodes hanya bisa dikunjungi satu kali. Penelitian selanjutnya diharapkan dapat mengatasi faktor batasan yang penyusun rancang saat ini.

DAFTAR PUSTAKA

Braekers, Kris,. Ramaekers, Katrien., dan Nieuwenhuyse, Inneke V. 2016. *The Vehicle Routing Problem: State of The Art Classification and Review.* Computers & Industrial Engineering 99: 300–313.

Cahyaningsih, Wahyu K., Sari, Eminugroho R., Dan Hernawati, Kuswari. 2014. Penyelesaian *Capacitated Vehicle Routing Problem* (CVRP) Menggunakan Algoritma *Sweep* Untuk Optimasi Rute Distribusi Surat Kabar Kedaulatan Rakyat. Seminar Nasional Matematika Dan Pendidikan Matematika UNY.

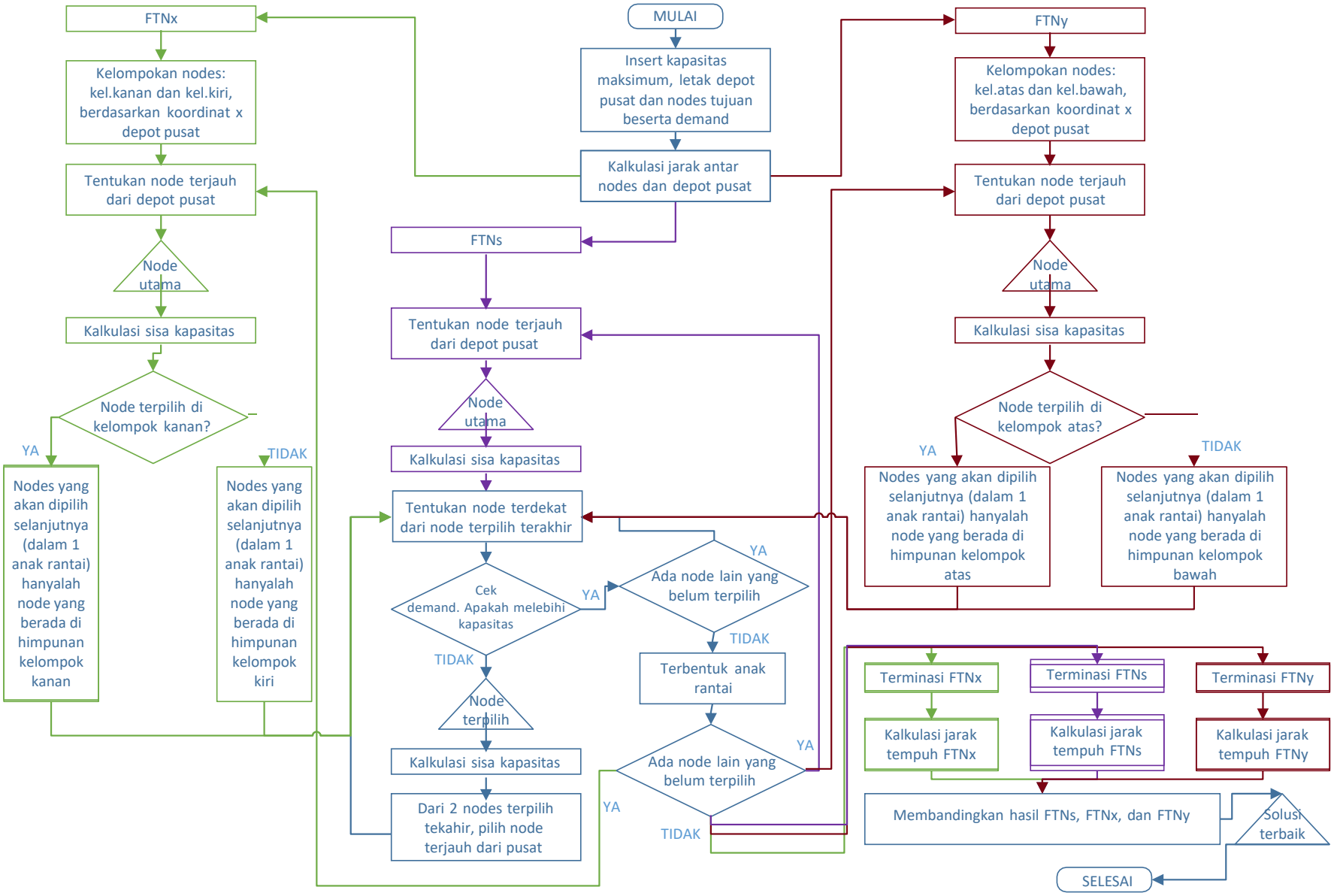
Carotenuto, Pasquale. , Giordani, Stefano,. Massari, Simone., dan Vagaggini, Fabrizio. 2015. *Periodic capacitated vehicle routing for retail distribution of fuel oils.* Transportation Research Procedia, Vol. 10: 735 – 744.

Faulina, Javier., Juan, Angel., Lera, Fernando., dan Grasman Scott. 2011. *Solving the Capacitated Vehicle Routing Problem with Environmental Criteria Based on Real Estimations in Road Transportation: A Case Study.* Procedia Social and Behavioral Sciences, Vol. 20: 323 – 334.

Satria, Wijaya., Siallagan, Manahan., dan Novani, Santi. 2004. Penerapan Metode Algoritma Genetik Untuk Memecahkan Masalah Penentuan Rute Kendaraan Berkendala Kapasitas.

Subramanian, Anand., Uchoa, Eduardo,. Ochi, Luiz S. 2013. *A Hybrid Algorithm for a Class of Vehicle Routing Problems*. Computers & Operations Research 40: 2519–2531.

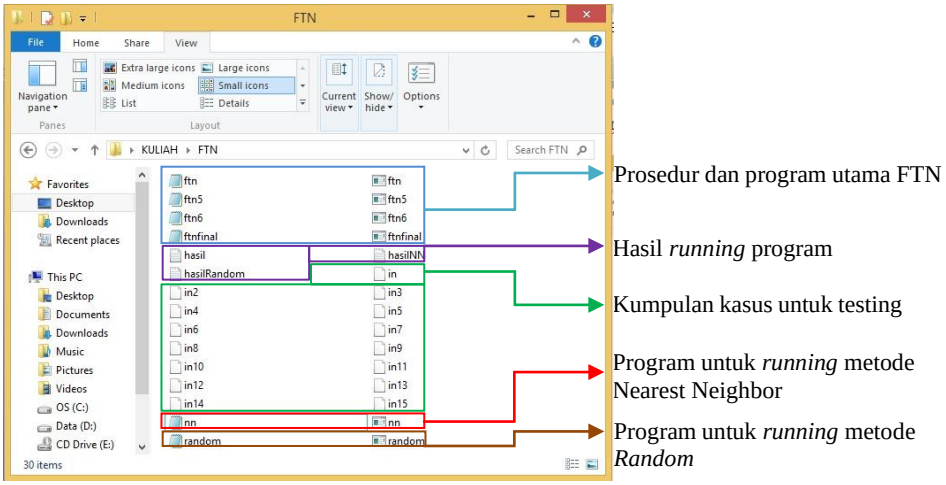
FLOWCHART FTN


MULAI
Insert kapasitas maksimum, letak depot pusat dan nodes tujuan beserta demand
Kalkulasi jarak antar nodes dan depot pusat
FTNx
Kelompokan nodes: kel.kanan dan kel.kiri, berdasarkan koordinat x depot pusat
Tentukan node terjauh dari depot pusat
Node utama
Kalkulasi sisa kapasitas
Node terpilih di kelompok kanan?
YA
TIDAK
Nodes yang akan dipilih selanjutnya (dalam 1 anak rantai) hanyalah node yang berada di himpunan kelompok kanan
Nodes yang akan dipilih selanjutnya (dalam 1 anak rantai) hanyalah node yang berada di himpunan kelompok kiri
FTNs
Tentukan node terjauh dari depot pusat
Node utama
Kalkulasi sisa kapasitas
Tentukan node terdekat dari node terpilih terakhir
Cek demand. Apakah melebihi kapasitas
YA
TIDAK
Node terpilih
Kalkulasi sisa kapasitas
Dari 2 nodes terpilih tekahir, pilih node terjauh dari pusat
Ada node lain yang belum terpilih
YA
TIDAK
Terbentuk anak rantai
Ada node lain yang belum terpilih
YA
TIDAK
FTNy
Kelompokan nodes: kel.atas dan kel.bawah, berdasarkan koordinat x depot pusat
Tentukan node terjauh dari depot pusat
Node utama
Kalkulasi sisa kapasitas
Node terpilih di kelompok atas?
YA
TIDAK
Nodes yang akan dipilih selanjutnya (dalam 1 anak rantai) hanyalah node yang berada di himpunan kelompok atas
Nodes yang akan dipilih selanjutnya (dalam 1 anak rantai) hanyalah node yang berada di himpunan kelompok bawah
Terminasi FTNx
Terminasi FTNs
Terminasi FTNy
Kalkulasi jarak tempuh FTNx
Kalkulasi jarak tempuh FTNs
Kalkulasi jarak tempuh FTNy
Membandingkan hasil FTNs, FTNx, dan FTNy
Solusi terbaik
SELESAI

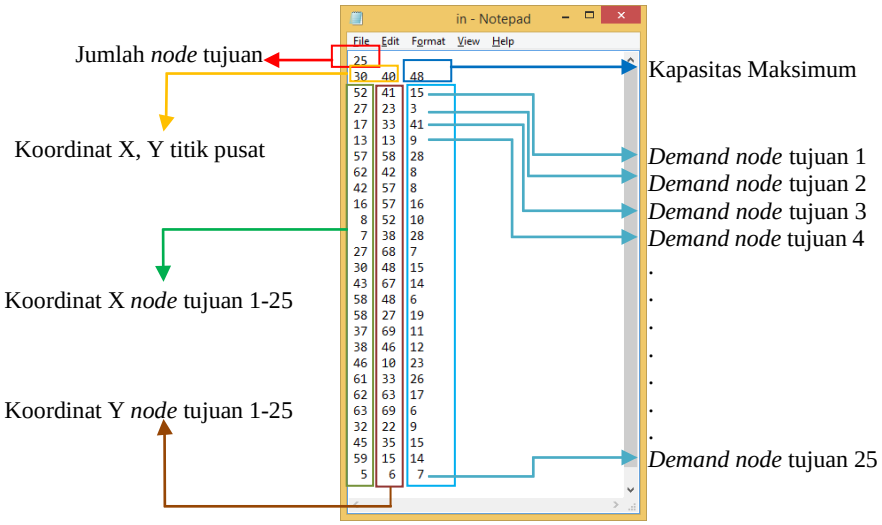
MANUAL BOOK

MANUAL BOOK

Bagian ini menjelaskan mengenai cara *running* program FTN. Sebelum memulai, pastikan bahwa semua file (harus semua file pada Gambar 1.) berada dalam satu folder, seperti terlihat pada Gambar 1. *Study case* yang akan diuji performanya berada pada file in, in2, in3,.., in15 dengan format penulisan *case* digambarkan pada Gambar 2. *Case* dibuat terpisah guna mempermudah *copy paste cases* dari excel.



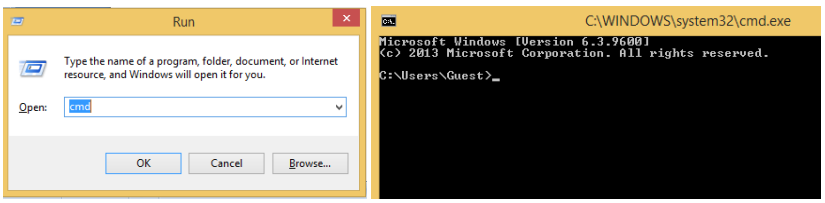
Gambar 1. File (dalam satu folder) yang diperlukan untuk eksperimen/ *running* program



Gambar 2. Format penulisan *case* yang akan diuji

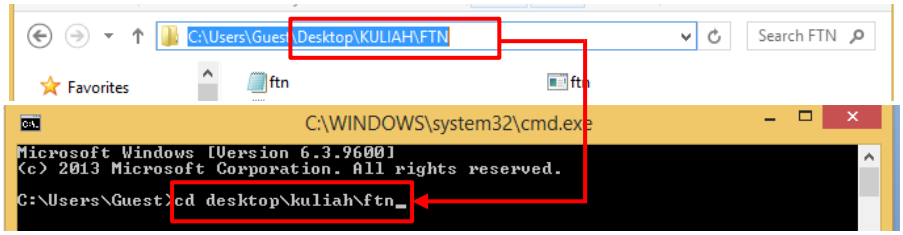
Untuk me-*running* program, diperlukan beberapa langkah sebagai berikut:

1. Buka *command prompt* dengan cara menekan ⊞+R secara bersamaan. Setelah itu, ketik cmd lalu klik OK. Seperti pada Gambar 3a dan Gambar 3b.



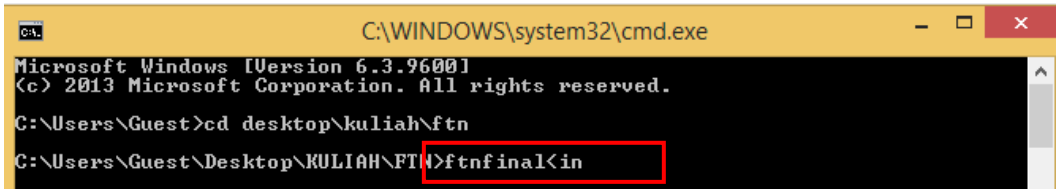
Gambar 3a dan 3b. Langkah pertama membuka command promt

2. Sesuaikan directory *command prompt* dengan letak dimana folder yang akan diakses. Ketik cd (spasi) lalu ketikan *directory* folder yang memuat file-file yang akan di*running*. Seperti pada Gambar 4.



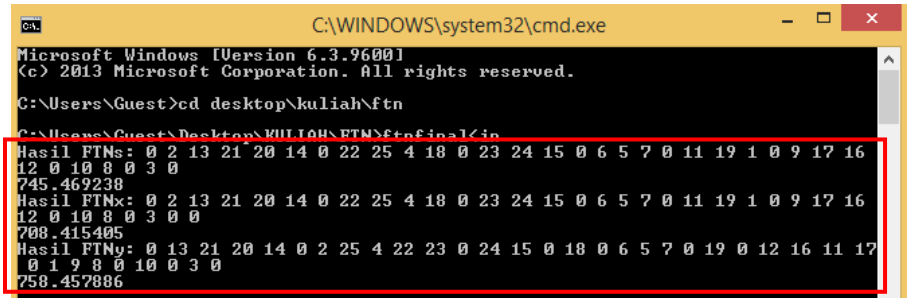
Gambar 4. Langkah kedua, menyesuaikan directory

3. *Running* program dengan cara mengetikan nama file yang memuat program ("ftnfinal" untuk *running* metode *Farthest to Nearest*; "nn" untuk *running* metode *Nearest Neighbor*; dan "*random*" untuk *running* metode *random*) diikuti dengan lambing lebih kecil (<) beserta judul *study case* yang akan di*running*. Misalnya akan diuji hasil *study case* berjudul "in" menggunakan metode FTN, maka yang harus diketikan adalah "ftnfinal<in", seperti ditunjukkan pada Gambar 5.



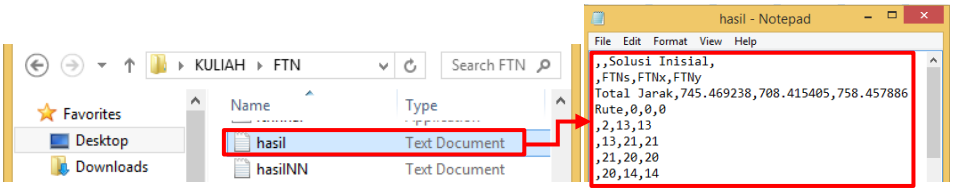
Gambar 5. Mengakes/ *running* metode FTN untuk *study case* berjudul "in"

4. Setelah langkah tiga selesai, tekan enter, maka akan didapatkan hasil pada *command prompt* seperti pada Gambar 6.



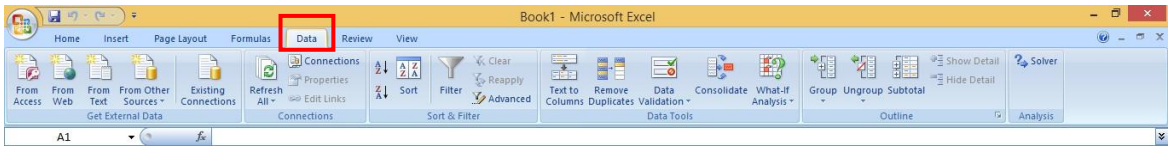
Gambar 6. Hasil *running* metode FTN untuk *study case* berjudul "in" pada *command prompt*

5. Untuk mempermudah pengolahan data ke dalam excel, maka hasil pada Gambar 6. diimport kedalam file hasil berbentuk .txt ("hasil.txt" untuk hasil *running* metode FTN; "hasilNN" untuk hasil *running* metode NN; dan "hasil*Random*" untuk hasil *running* metode *Random*). Perlu diingat bahwa, pada saat *running* program pada command promt, file hasil.txt ini tidak boleh dalam keadaan dibuka. File ini akan menampilkan hasil *running* ketika dibuka setelah *running* pada command promt. Isi dari file hasil setelah *running command prompt* terlihat pada Gambar 7.

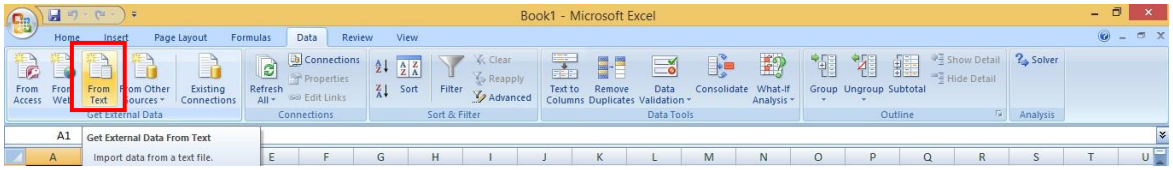


Gambar 7. Hasil file "hasil.txt" pada notepad

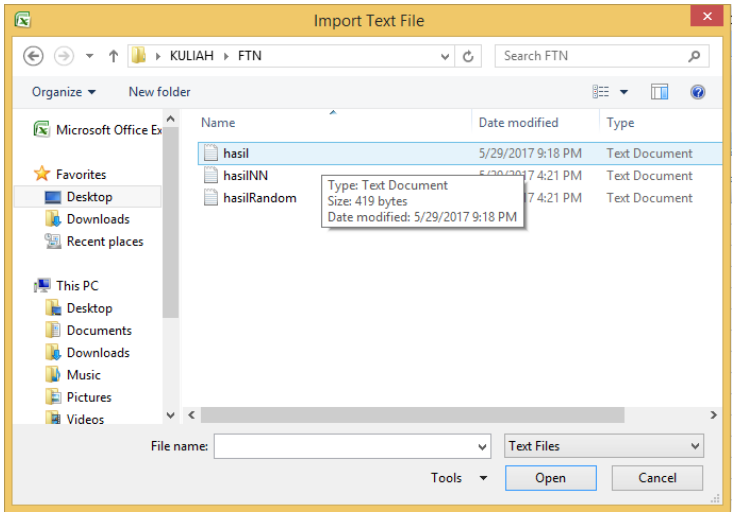
6. Seperti yang disebutkan pada langkah lima, program ini dirancang dan disusun sedemikian rupa, agar mudah untuk diakses dan diolah dalam excel. Oleh karena itu, hasil pada Gambar 7., akan lebih mudah dianalisa ketika diimport dalam excel. Cara mengimport file .txt kedalam excel dapat dilihat pada Gambar 8. sampai dengan Gambar 15.



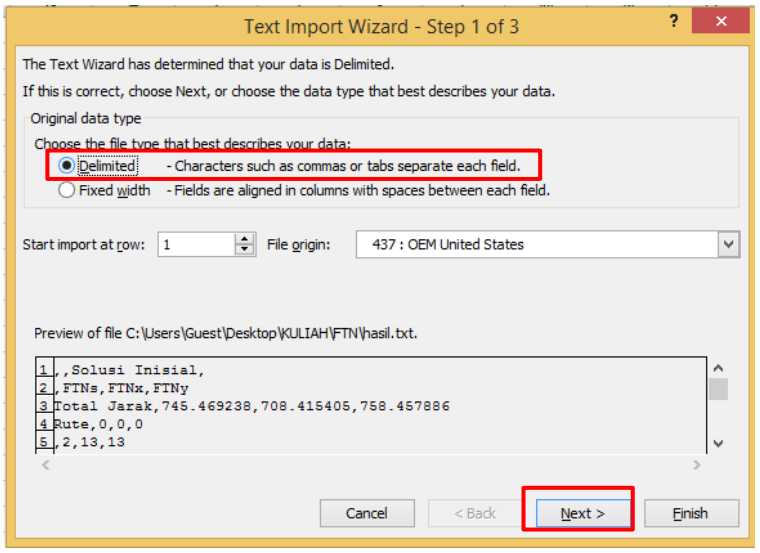
Gambar 8. Buka Ms. Excel, lalu klik menu bar "Data"



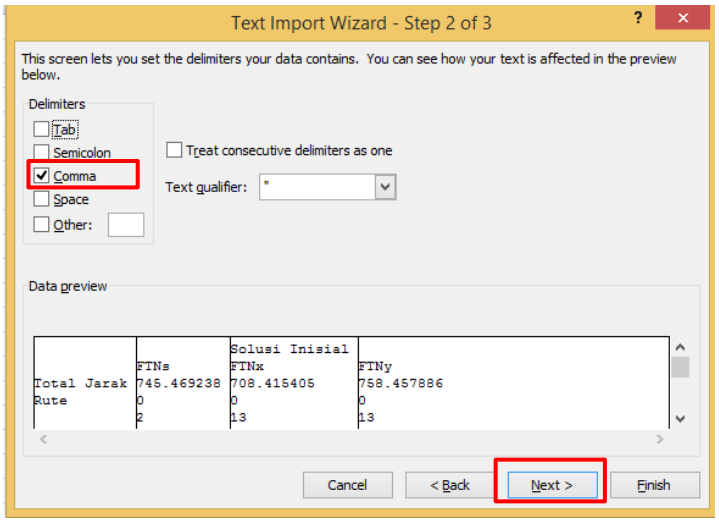
Gambar 9. Pilih tool bar "From Text", lalu klik tool bar tersebut untuk mengimport file .txt



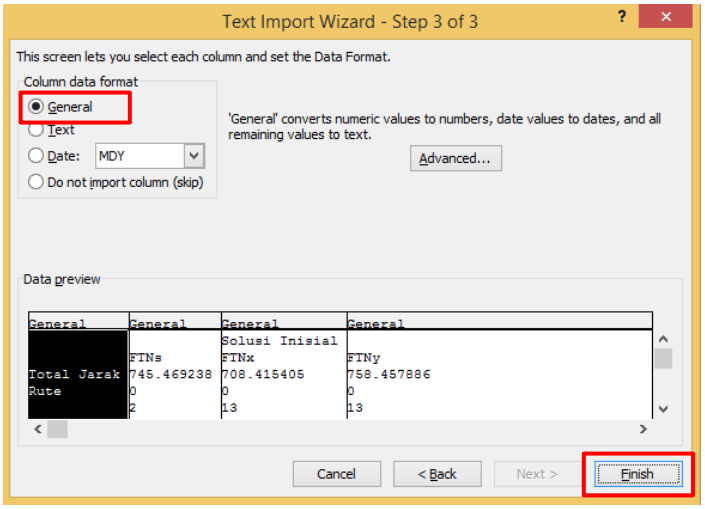
Gambar 10. Pilih judul file .txt yang akan diimport, klik file tersebut lalu enter



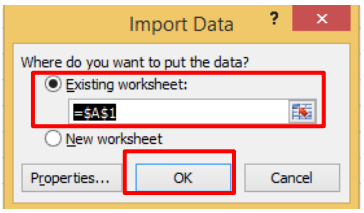
Gambar 11. Klik "Delimited" lalu klik "Next>"



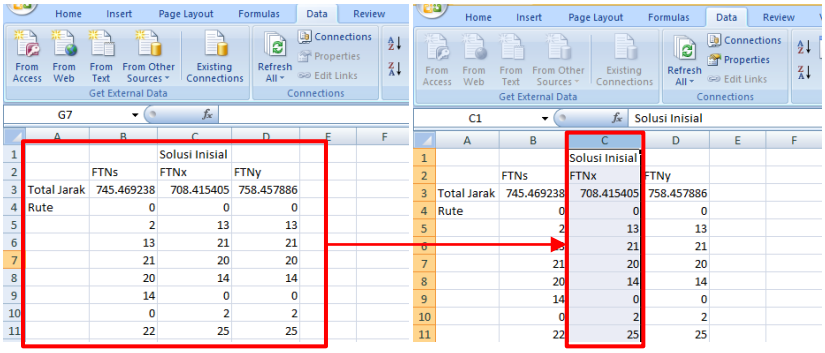
Gambar 12. Pilih delimiters “Comma” lalu klik “Next>”



Gambar 13. Pilih format kolom “General” lalu klik “Finish”



Gambar 14. Tentukan kolom dan baris mana yang akan menjadi space pertama data lalu klik “OK”



Gambar 15a. Hasil setelah hasil.txt diimport. Maka terlihat bahwa solusi inisial FTN adalah kolom C

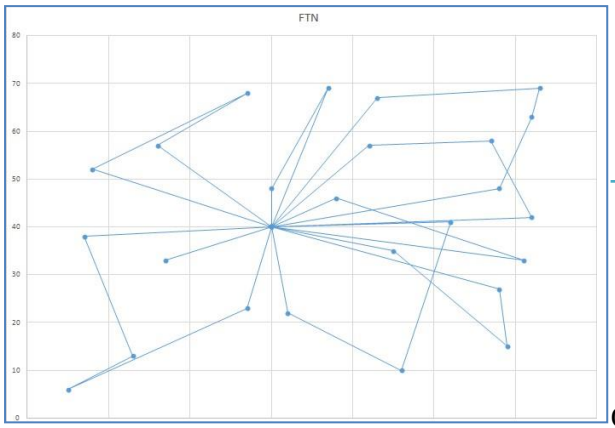
	A	B	C	D
1		NN		
2	Total Jarak	719.31897		
3	Rute	0		
4		12		
5		17		
6		7		
7		16		
8		0		
9		3		
10		2		
11		0		
12		23		
13		1		
14		14		
15		6		

Gambar 15b. Hasil setelah hasilNN.txt diimport. Maka terlihat solusi inisial NN

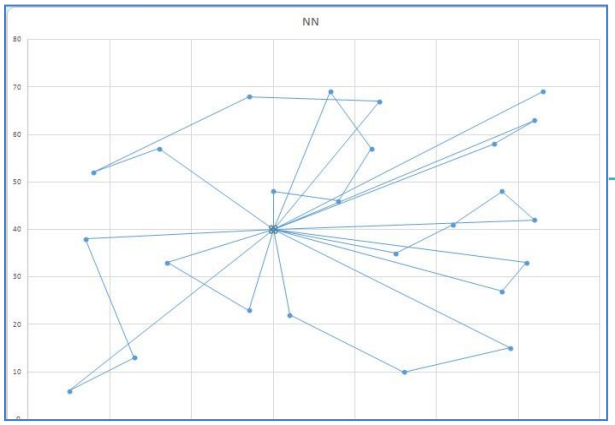
	A	B	C	D
1		Random		
2	Total Jarak	1088.486694		
3	Rute	0		
4		15		
5		12		
6		21		
7		14		
8		0		
9		22		
10		11		
11		1		
12		16		
13		2		
14		0		
15		23		

Gambar 15C. Hasil setelah hasil*Random*.txt diimport. Maka terlihat solusi inisial *Random*

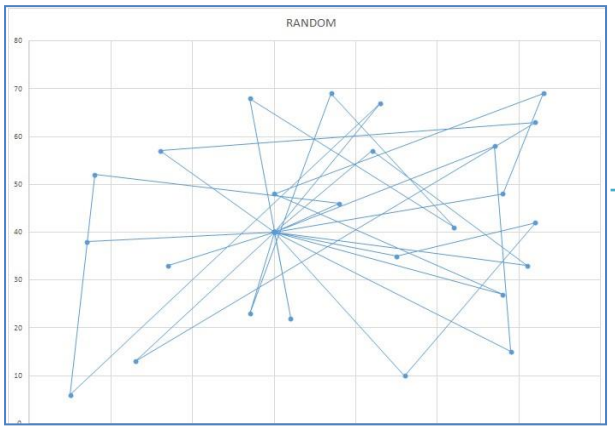
7. Penyusun merancang akses hasil dan study case program melalui excel agar mempermudah mendapatkan hasil tanpa menulis ulang hasil dari program berjalan serta mempermudah pengguna dalam pengolahan data. Sehingga dari excel dapat dilihat perbedaannya melalui grafik rute yang diilustrasikan pada Gambar 16a, 16b, dan 16c.



Gambar 16a. Grafik rute FTN, case “in3”



Gambar 16a. Grafik rute NN, case “in3”



Gambar 16a. Grafik rute *Random*, case “in3”

STUDY CASES

STUDY CASES

Bagian ini melampirkan 15 *study case* dalam testing program mencari solusi inisiasi. Dengan kota 0 sebagai depot pusat.

***Study case* in**

Kapasitas: 10

Kota	x	y	*demand*
0	-5	5	0
1	8	-8	2
2	2	0	5
3	-2	-10	5
4	8	-10	4

***Study case* in2**

Kapasitas: 10

Kota	x	y	*demand*	Kota	x	y	*demand*
0	-5	5	0	9	8	7	2
1	8	-8	2	10	1	1	5
2	2	0	5	11	2	2	5
3	-2	-10	5	12	3	3	4
4	8	-10	4	13	4	4	1
5	41	1	2	14	-3	-3	1
6	20	6	5	15	-2	-2	1
7	-23	2	5	16	-1	-1	1
8	6	-30	4				

***Study case* in3**

Kapasitas: 48

Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*
0	30	40	0	7	42	57	8	14	58	48	6	21	63	69	6
1	52	41	15	8	16	57	16	15	58	27	19	22	32	22	9
2	27	23	3	9	8	52	10	16	37	69	11	23	45	35	15
3	17	33	41	10	7	38	28	17	38	46	12	24	59	15	14
4	13	13	9	11	27	68	7	18	46	10	23	25	5	6	7
5	57	58	28	12	30	48	15	19	61	33	26				
6	62	42	8	13	43	67	14	20	62	63	17				

***Study case* in4**

Kapasitas: 15

Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*
0	6823	4674	0	13	6471	4275	1	26	4326	1426	1	39	7010	2710	1
1	7692	2247	1	14	7110	4369	1	27	5164	1440	1	40	9005	3996	1
2	9135	6748	1	15	6462	2634	1	28	8389	5804	1	41	7576	7065	1
3	7721	3451	1	16	8476	2874	1	29	4639	1629	1	42	4246	1701	1
4	8304	8580	1	17	3961	1370	1	30	6344	1436	1	43	5906	1472	1
5	7501	5899	1	18	5555	1519	1	31	5840	5736	1	44	6469	8971	1
6	4687	1373	1	19	4422	1249	1	32	5972	2555	1	45	6152	2174	1
7	5429	1408	1	20	5584	3081	1	33	7947	4373	1	46	5887	3796	1
8	7877	1716	1	21	5776	4498	1	34	6929	8958	1	47	7203	5958	1
9	7260	2083	1	22	8035	2880	1	35	5366	1733	1				
10	7096	7869	1	23	6963	3782	1	36	4550	1219	1				
11	6539	3513	1	24	6336	7348	1	37	6901	1589	1				
12	6272	2992	1	25	8139	8306	1	38	6316	5497	1				

***Study case* in5**

Kapasitas: 15

Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*
0	67893	3	0	13	281	-12	3	26	9812	-3	3	39	38	16	1
1	9812	4	3	14	38	-11	3	27	213	-4	1	40	392	17	3
2	213	5	3	15	392	-10	1	28	3328	-5	3	41	3	18	3
3	3328	6	1	16	3	-9	3	29	21	-6	3	42	1735	19	3
4	21	7	3	17	1735	-8	3	30	23	-7	3	43	9090	20	1
5	23	8	3	18	9090	-7	3	31	3221	-8	1	44	15312	21	3
6	3221	9	3	19	15312	-6	1	32	1234	-9	3	45	127	22	3
7	1234	10	1	20	127	-5	3	33	21213	-10	3	46	32373	23	3
8	21213	11	3	21	32373	-4	3	34	890	-11	3	47	5676	24	1
9	890	12	3	22	5676	-3	3	35	970	12	1	48	454	25	3
10	970	13	3	23	454	-2	1	36	8	13	3	49	34	26	1
11	8	-14	1	24	34	-1	3	37	462	14	3				
12	462	-13	3	25	67893	-2	3	38	281	15	3				

***Study case* in6**

Kapasitas: 10

Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*
0	0	0	0	23	23	-22	3	45	-44	44	5	67	-22	22	2
1	1	-44	1	24	24	-21	4	46	-43	43	1	68	-21	21	3
2	2	-43	2	25	25	-20	5	47	-42	42	2	69	-20	20	4
3	3	-42	3	26	26	-19	1	48	-41	41	3	70	-19	19	5
4	4	-41	4	27	27	-18	2	49	-40	40	4	71	-18	18	1
5	5	-40	5	28	28	-17	3	50	-39	39	5	72	-17	17	2
6	6	-39	1	29	29	-16	4	51	-38	38	1	73	-16	16	3
7	7	-38	2	30	30	-15	5	52	-37	37	2	74	-15	15	4
8	8	-37	3	31	31	-14	1	53	-36	36	3	75	-14	14	5
9	9	-36	4	32	32	-13	2	54	-35	35	4	76	-13	13	1
10	10	-35	5	33	33	-12	3	55	-34	34	5	77	-12	12	2
11	11	-34	1	34	34	-11	4	56	-33	33	1	78	-11	11	3
12	12	-33	2	35	35	-10	5	57	-32	32	2	79	-10	10	4
13	13	-32	3	36	36	-9	1	58	-31	31	3	80	-9	9	5
14	14	-31	4	37	37	-8	2	59	-30	30	4	81	-8	8	1
15	15	-30	5	38	38	-7	3	60	-29	29	5	82	-7	7	2
16	16	-29	1	39	39	-6	4	61	-28	28	1	83	-6	6	3
17	17	-28	2	40	40	-5	5	62	-27	27	2	84	-5	5	4
18	18	-27	3	41	41	-4	1	63	-26	26	3	85	-4	4	5
19	19	-26	4	42	42	-3	2	64	-25	25	4	86	-3	3	1
20	20	-25	5	43	43	-2	3	65	-24	24	5	87	-2	2	2
21	21	-24	1	44	44	-1	4	66	-23	23	1	88	-1	1	3
22	22	-23	2												

***Study case* in7**

Kapasitas: 100

Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*	Kota	x	y	*demand*
0	1	2	0	36	73	74	1	71	143	144	7	106	213	214	1
1	3	4	7	37	75	76	1	72	145	146	7	107	215	216	1
2	5	6	7	38	77	78	1	73	147	148	7	108	217	218	1
3	7	8	7	39	79	80	1	74	149	150	7	109	219	220	1
4	9	10	7	40	81	82	1	75	151	152	7	110	221	222	7
5	11	12	7	41	83	84	1	76	153	154	7	111	223	224	7
6	13	14	7	42	85	86	1	77	155	156	7	112	225	226	7
7	15	16	7	43	87	88	1	78	157	158	1	113	227	228	7
8	17	18	7	44	89	90	7	79	159	160	1	114	229	230	7
9	19	20	7	45	91	92	7	80	161	162	1	115	231	232	7
10	21	22	7	46	93	94	7	81	163	164	1	116	233	234	7
11	23	24	7	47	95	96	7	82	165	166	1	117	235	236	7
12	25	26	1	48	97	98	7	83	167	168	1	118	237	238	7
13	27	28	1	49	99	100	7	84	169	170	1	119	239	240	7
14	29	30	1	50	101	102	7	85	171	172	1	120	241	242	7
15	31	32	1	51	103	104	7	86	173	174	1	121	243	244	7
16	33	34	1	52	105	106	7	87	175	176	1	122	245	246	1
17	35	36	1	53	107	108	7	88	177	178	7	123	247	248	1
18	37	38	1	54	109	110	7	89	179	180	7	124	249	250	1
19	39	40	1	55	111	112	7	90	181	182	7	125	251	252	1
20	41	42	1	56	113	114	1	91	183	184	7	126	253	254	1
21	43	44	1	57	115	116	1	92	185	186	7	127	255	256	1
22	45	46	7	58	117	118	1	93	187	188	7	128	257	258	1
23	47	48	7	59	119	120	1	94	189	190	7	129	259	260	1
24	49	50	7	60	121	122	1	95	191	192	7	130	261	262	1
25	51	52	7	61	123	124	1	96	193	194	7	131	263	264	1
26	53	54	7	62	125	126	1	97	195	196	7	132	265	266	7
27	55	56	7	63	127	128	1	98	197	198	7	133	267	268	7
28	57	58	7	64	129	130	1	99	199	200	7	134	269	270	7
29	59	60	7	65	131	132	1	100	201	202	1	135	271	272	7
30	61	62	7	66	133	134	7	101	203	204	1	136	273	274	7
31	63	64	7	67	135	136	7	102	205	206	1	137	275	276	7
32	65	66	7	68	137	138	7	103	207	208	1	138	277	278	7
33	67	68	7	69	139	140	7	104	209	210	1	139	279	280	7
34	69	70	1	70	141	142	7	105	211	212	1	140	281	282	7
35	71	72	1												

***Study case* in8**

Kapasitas: 80

Kota	X	Y	*Demand*	Kota	X	Y	*Demand*	Kota	X	Y	*Demand*	Kota	X	Y	*Demand*
0	-30.2832	-14.3051	0	51	-4.5105	-4.5105	4	102	3	-6	9	153	9	-4	3
1	-30.2832	-30.2832	4	52	-4.1497	-5.3125	8	103	3	3	6	154	9	-6	6
2	-27.5302	-9.5367	2	53	-4.1497	-4.1497	1	104	3.1574	-7.8125	7	155	9	9	6
3	-27.5302	-27.5302	5	54	-3.6084	-6.25	2	105	3.1574	3.1574	10	156	9.0211	-6.25	9
4	-20.3723	-2.861	2	55	-3.6084	-3.6084	5	106	3.3198	-4.25	8	157	9.0211	-7.8125	3
5	-20.3723	-20.3723	5	56	-3.5239	-18.3105	1	107	3.3198	3.3198	1	158	9.0211	9.0211	2
6	-19.3813	9.1553	3	57	-3.5239	18.3105	2	108	3.5239	-18.3105	7	159	10.1487	-1.9531	10
7	-19.3813	-19.3813	6	58	-3.5239	-3.5239	5	109	3.5239	18.3105	8	160	10.1487	10.1487	3
8	-16.2979	2.2888	4	59	-3.3198	-4.25	9	110	3.5239	3.5239	10	161	10.4306	-1.4648	1
9	-16.2979	-16.2979	7	60	-3.3198	-3.3198	2	111	3.6084	-6.25	4	162	10.4306	10.4306	4
10	-15.8574	3.0518	5	61	-3.1574	-7.8125	10	112	3.6084	3.6084	7	163	10.8253	-6.25	7
11	-15.8574	-15.8574	8	62	-3.1574	-3.1574	3	113	4	-13	2	164	10.8253	10.8253	10
12	-14.0955	-4.8828	3	63	-2.753	-33.3786	1	114	4	4	5	165	11	-4	1
13	-14.0955	-14.0955	6	64	-2.753	-2.753	4	115	4.1497	-5.3125	9	166	11	-6	4
14	-13.5316	-7.8125	9	65	-2.7063	-7.8125	2	116	4.1497	4.1497	2	167	11	11	4
15	-13.5316	-13.5316	2	66	-2.7063	-2.7063	5	117	4.5106	-7.8125	5	168	11.2764	-7.8125	2
16	-13.0383	1.8311	6	67	-2.5259	-6.25	3	118	4.5106	4.5106	8	169	11.2764	3.9062	4
17	-13.0383	-13.0383	9	68	-2.5259	-2.5259	6	119	5	-4	7	170	11.2764	11.2764	5
18	-12.6859	-2.4414	7	69	-2.2553	-11.7187	3	120	5	-6	6	171	12.404	5.8594	3
19	-12.6859	2.4414	8	70	-2.2553	-2.2553	6	121	5	5	10	172	12.404	12.404	6
20	-12.6859	-12.6859	10	71	-2.1651	-6.25	4	122	5.4127	-6.25	10	173	12.6859	-2.4414	4
21	-12.404	5.8594	9	72	-2.1651	-2.1651	7	123	5.4127	5.4127	3	174	12.6859	2.4414	5
22	-12.404	-12.404	2	73	-1.7619	-21.3623	5	124	5.506	-28.6102	9	175	12.6859	12.6859	7
23	-11.2764	-7.8125	10	74	-1.7619	21.3623	6	125	5.506	28.6102	10	176	13	-4	9
24	-11.2764	-3.9063	4	75	-1.7619	-1.7619	9	126	5.506	5.506	3	177	13	-6	2
25	-11.2764	3.9062	5	76	-1.4434	-7.5	4	127	6	-13	1	178	13	13	2
26	-11.2764	-11.2764	3	77	-1.4434	-1.4434	7	128	6	6	4	179	13.0383	1.8311	6
27	-10.8253	-6.25	10	78	-1.4095	-17.0898	7	129	6.4952	-1.25	1	180	13.0383	13.0383	9
28	-10.8253	-10.8253	3	79	-1.4095	-1.4095	10	130	6.4952	6.4952	4	181	13.5316	-7.8125	8
29	-10.1487	-1.9531	1	80	-1.1276	13.6719	8	131	6.6756	0.9375	2	182	13.5316	13.5316	1
30	-10.1487	-10.1487	4	81	1.1276	13.6719	9	132	6.6756	6.6756	5	183	14.0955	-4.8828	5
31	-9.0211	-7.8125	6	82	1.4095	-17.0898	10	133	6.7658	-7.8125	3	184	14.0955	14.0955	8
32	-9.0211	-9.0211	9	83	1.4095	1.4095	3	134	6.7658	6.7658	6	185	15	-4	8
33	-8.3445	-1.1719	2	84	1.4434	-7.5	5	135	7	-4	5	186	15	-6	10
34	-8.3445	1.1719	3	85	1.4434	1.4434	8	136	7	-6	8	187	15	15	1
35	-8.3445	-8.3445	6	86	1.7619	-21.3623	1	137	7	7	8	188	15.8574	3.0518	7
36	-8.119	-1.5625	4	87	1.7619	21.3623	2	138	7.2169	-6.25	1	189	15.8574	15.8574	10
37	-8.119	-8.119	7	88	1.7619	1.7619	5	139	7.2169	2.5	2	190	16.2979	2.2888	8
38	-7.2169	-6.25	7	89	2	-13	3	140	7.2169	7.2169	4	191	16.2979	16.2979	1
39	-7.2169	2.5	8	90	2	2	6	141	7.2169	7.2169	5	192	17	-4	7
40	-7.2169	-7.2169	1	91	2.1651	-6.25	3	142	8	-13	10	193	17	17	10
41	-6.7658	-7.8125	5	92	2.1651	2.1651	6	143	8	8	3	194	19.3813	9.1553	9
42	-6.7658	-6.7658	8	93	2.2553	-11.7187	6	144	8.1049	-10.376	4	195	19.3813	19.3813	2
43	-6.6756	0.9375	6	94	2.2553	2.2553	9	145	8.1049	8.1049	7	196	20.3723	-2.861	10
44	-6.6756	-6.6756	9	95	2.5259	-6.25	4	146	8.119	-1.5625	5	197	20.3723	20.3723	3
45	-5.506	-28.6102	9	96	2.5259	2.5259	7	147	8.119	8.119	8	198	27.5302	-9.5367	6
46	-5.506	28.6102	10	97	2.7063	-7.8125	5	148	8.2591	-23.8418	6	199	27.5302	27.5302	9
47	-5.506	-5.506	2	98	2.7063	2.7063	8	149	8.2591	8.2591	9	200	30.2832	-14.3051	1
48	-5.4127	-6.25	7	99	2.753	-33.3786	6	150	8.3445	-1.1719	7	201	30.2832	30.2832	4
49	-5.4127	-5.4127	10	100	2.753	2.753	9	151	8.3445	1.1719	8				
50	-4.5105	-7.8125	1	101	3	-4	3	152	8.3445	8.3445	1				

***Study case* in9**

Kapasitas: 350

Kota	X	Y	Dema	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
0	6.4952	1.25	0	63	9	-12	47	126	11.2764	9.7656	21	189	16	5	21
1	6.6072	19.0735	13	64	9	-10	42	127	12	-9	36	190	16.2979	-2.2888	13
2	6.6073	-19.0735	13	65	9	-8	39	128	12	-7	33	191	16.2979	2.2888	13
3	6.6756	-0.9375	13	66	9	-6	36	129	12	-5	31	192	16.5181	-19.0735	13
4	6.6756	0.9375	13	67	9	-4	33	130	12	-3	29	193	16.5181	19.0735	13
5	6.7658	-7.8125	13	68	9	-2	31	131	12	-1	27	194	16.9146	-9.7656	42
6	6.7658	7.8125	13	69	9	0	29	132	12	1	25	195	16.9146	9.7656	42
7	7	-14	60	70	9	2	27	133	12	3	24	196	24.7772	4.7684	13
8	7	-12	52	71	9	4	25	134	12	5	23	197	26.429	-15.2588	42
9	7	-10	47	72	9	6	24	135	12	7	21	198	26.429	15.2588	42
10	7	-8	42	73	9	8	23	136	12	9	21	199	27.5302	-19.0735	13
11	7	-6	39	74	9	10	21	137	12.404	-5.8594	13	200	27.5302	-9.5367	21
12	7	-4	36	75	9	12	21	138	12.404	5.8594	13	201	27.5302	9.5367	21
13	7	-2	33	76	9.0211	-7.8125	21	139	12.6639	-16.2125	13	202	27.5302	19.0735	13
14	7	0	31	77	9.0211	-6.25	13	140	12.6639	16.2125	13	203	30.2832	-14.3051	13
15	7	2	29	78	9.0211	-3.125	21	141	12.6859	-2.4414	13	204	30.2832	14.3051	13
16	7	4	27	79	9.0211	0	42	142	12.6859	2.4414	13	205	33.0362	19.0735	42
17	7	6	25	80	9.0211	3.125	21	143	13	-8	33	206	33.0363	-19.0735	42
18	7	8	24	81	9.0211	6.25	13	144	13	-6	31	207	-33.0362	-19.0735	0
19	7	10	23	82	9.0211	7.8125	21	145	13	-4	29	208	-33.0362	19.0735	42
20	7	12	21	83	9.9232	-4.6875	13	146	13	-2	27	209	-30.2832	-14.3051	13
21	7	14	21	84	9.9232	4.6875	13	147	13	0	25	210	-30.2832	14.3051	13
22	7.0477	-12.207	42	85	10	-11	42	148	13	2	24	211	-27.5302	-19.0735	13
23	7.0477	12.207	42	86	10	-9	39	149	13	4	23	212	-27.5302	-9.5367	21
24	7.2169	-6.25	21	87	10	-7	36	150	13	6	21	213	-27.5302	9.5367	21
25	7.2169	-5	13	88	10	-5	33	151	13	8	21	214	-27.5302	19.0735	13
26	7.2169	-2.5	21	89	10	-3	31	152	13.0383	-1.8311	13	215	-26.429	-15.2588	42
27	7.2169	0	42	90	10	-1	29	153	13.0383	1.8311	13	216	-26.429	15.2588	42
28	7.2169	2.5	21	91	10	1	27	154	13.2145	-15.2588	13	217	-24.7772	-4.7684	13
29	7.2169	5	13	92	10	3	25	155	13.2145	15.2588	13	218	-24.7772	4.7684	13
30	7.2169	6.25	21	93	10	5	24	156	13.5316	-7.8125	42	219	-24.2266	-11.4441	13
31	7.7085	-19.0735	13	94	10	7	23	157	13.5316	7.8125	42	220	-24.2266	11.4441	13
32	7.7085	19.0735	13	95	10	9	21	158	14	-7	31	221	-22.0242	-19.0735	21
33	7.9386	-3.75	13	96	10	11	21	159	14	-5	29	222	-22.0242	-15.2588	13
34	7.9386	3.75	13	97	10.1311	-12.97	13	160	14	-3	27	223	-22.0242	-7.6294	21
35	8	-13	52	98	10.1311	12.97	13	161	14	-1	25	224	-22.0242	0	42
36	8	-11	47	99	10.1487	-1.9531	13	162	14	1	24	225	-22.0242	7.6294	21
37	8	-9	42	100	10.1487	1.9531	13	163	14	3	23	226	-22.0242	15.2588	13
38	8	-7	39	101	10.4306	-1.4648	13	164	14	5	21	227	-22.0242	19.0735	21
39	8	-5	36	102	10.4306	1.4648	13	165	14	7	21	228	-21.1432	-12.207	42
40	8	-3	33	103	10.5716	-12.207	13	166	14.0955	-12.207	21	229	-21.1432	12.207	42
41	8	-1	31	104	10.5716	12.207	13	167	14.0955	-9.7656	13	230	-20.3723	-2.861	13
42	8	1	29	105	10.8253	-6.25	42	168	14.0955	-4.8828	21	231	-20.3723	2.861	13
43	8	3	27	106	10.8253	6.25	42	169	14.0955	0	42	232	-19.8217	-3.8147	13
44	8	5	25	107	11	-10	39	170	14.0955	4.8828	21	233	-19.8217	3.8147	13
45	8	7	24	108	11	-8	36	171	14.0955	9.7656	13	234	-19.3813	-9.1553	13
46	8	9	23	109	11	-6	33	172	14.0955	12.207	21	235	-19.3813	9.1553	13
47	8	11	21	110	11	-4	31	173	15	-6	29	236	-17.6193	-15.2588	21
48	8	13	21	111	11	-2	29	174	15	-4	27	237	-17.6193	-12.207	13
49	8.1049	-10.376	13	112	11	0	27	175	15	-2	25	238	-17.6193	-6.1035	21
50	8.1049	10.376	13	113	11	2	25	176	15	0	24	239	-17.6193	0	42
51	8.119	-1.5625	13	114	11	4	24	177	15	2	23	240	-17.6193	6.1035	21
52	8.119	1.5625	13	115	11	6	23	178	15	4	21	241	-17.6193	12.207	13
53	8.2591	-23.8418	13	116	11	8	21	179	15	6	21	242	-17.6193	15.2588	21
54	8.2591	23.8419	13	117	11	10	21	180	15.505	-7.3242	13	243	-16.9146	-9.7656	42
55	8.3445	-1.1719	13	118	11.0121	-19.0735	42	181	15.505	7.3242	13	244	-16.9146	9.7656	42
56	8.3445	1.1719	13	119	11.0121	19.0735	42	182	15.8574	-3.0518	13	245	-16.5181	-19.0735	13
57	8.4573	-9.7656	13	120	11.2764	-9.7656	21	183	15.8574	3.0518	13	246	-16.5181	19.0735	13
58	8.4573	9.7656	13	121	11.2764	-7.8125	13	184	16	-5	27	247	-16.2979	-2.2888	13
59	8.6603	-5	42	122	11.2764	-3.9062	21	185	16	-3	25	248	-16.2979	2.2888	13
60	8.6603	5	42	123	11.2764	0	42	186	16	-1	24				
61	8.8097	-15.2588	42	124	11.2764	3.9062	21	187	16	1	23				
62	8.8097	15.2588	42	125	11.2764	7.8125	13	188	16	3	21				

***Study case* in10**

Kapasitas: 1400

Kota	X	Y	*Demand*	Kota	X	Y	*Demand*	Kota	X	Y	*Demand*	Kota	X	Y	*Demand*
0	0	0	0	75	75	75	70	150	150	150	70	225	225	225	70
1	1	1	70	76	76	76	70	151	151	151	70	226	226	226	70
2	2	2	70	77	77	77	70	152	152	152	70	227	227	227	70
3	3	3	70	78	78	78	70	153	153	153	70	228	228	228	70
4	4	4	70	79	79	79	70	154	154	154	70	229	229	229	70
5	5	5	70	80	80	80	70	155	155	155	70	230	230	230	70
6	6	6	70	81	81	81	70	156	156	156	70	231	231	231	70
7	7	7	70	82	82	82	70	157	157	157	70	232	232	232	70
8	8	8	70	83	83	83	70	158	158	158	70	233	233	233	70
9	9	9	70	84	84	84	70	159	159	159	70	234	234	234	70
10	10	10	70	85	85	85	70	160	160	160	70	235	235	235	70
11	11	11	70	86	86	86	70	161	161	161	70	236	236	236	70
12	12	12	70	87	87	87	70	162	162	162	70	237	237	237	70
13	13	13	70	88	88	88	70	163	163	163	70	238	238	238	70
14	14	14	70	89	89	89	70	164	164	164	70	239	239	239	70
15	15	15	70	90	90	90	70	165	165	165	70	240	240	240	70
16	16	16	70	91	91	91	70	166	166	166	70	241	241	241	70
17	17	17	70	92	92	92	70	167	167	167	70	242	242	242	70
18	18	18	70	93	93	93	70	168	168	168	70	243	243	243	70
19	19	19	70	94	94	94	70	169	169	169	70	244	244	244	70
20	20	20	70	95	95	95	70	170	170	170	70	245	245	245	70
21	21	21	70	96	96	96	70	171	171	171	70	246	246	246	70
22	22	22	70	97	97	97	70	172	172	172	70	247	247	247	70
23	23	23	70	98	98	98	70	173	173	173	70	248	248	248	70
24	24	24	70	99	99	99	70	174	174	174	70	249	249	249	70
25	25	25	70	100	100	100	70	175	175	175	70	250	250	250	70
26	26	26	70	101	101	101	70	176	176	176	70	251	251	251	70
27	27	27	70	102	102	102	70	177	177	177	70	252	252	252	70
28	28	28	70	103	103	103	70	178	178	178	70	253	253	253	70
29	29	29	70	104	104	104	70	179	179	179	70	254	254	254	70
30	30	30	70	105	105	105	70	180	180	180	70	255	255	255	70
31	31	31	70	106	106	106	70	181	181	181	70	256	256	256	70
32	32	32	70	107	107	107	70	182	182	182	70	257	257	257	70
33	33	33	70	108	108	108	70	183	183	183	70	258	258	258	70
34	34	34	70	109	109	109	70	184	184	184	70	259	259	259	70
35	35	35	70	110	110	110	70	185	185	185	70	260	260	260	70
36	36	36	70	111	111	111	70	186	186	186	70	261	261	261	70
37	37	37	70	112	112	112	70	187	187	187	70	262	262	262	70
38	38	38	70	113	113	113	70	188	188	188	70	263	263	263	70
39	39	39	70	114	114	114	70	189	189	189	70	264	264	264	70
40	40	40	70	115	115	115	70	190	190	190	70	265	265	265	70
41	41	41	70	116	116	116	70	191	191	191	70	266	266	266	70
42	42	42	70	117	117	117	70	192	192	192	70	267	267	267	70
43	43	43	70	118	118	118	70	193	193	193	70	268	268	268	70
44	44	44	70	119	119	119	70	194	194	194	70	269	269	269	70
45	45	45	70	120	120	120	70	195	195	195	70	270	270	270	70
46	46	46	70	121	121	121	70	196	196	196	70	271	271	271	70
47	47	47	70	122	122	122	70	197	197	197	70	272	272	272	70
48	48	48	70	123	123	123	70	198	198	198	70	273	273	273	70
49	49	49	70	124	124	124	70	199	199	199	70	274	274	274	70
50	50	50	70	125	125	125	70	200	200	200	70	275	275	275	70
51	51	51	70	126	126	126	70	201	201	201	70	276	276	276	70
52	52	52	70	127	127	127	70	202	202	202	70	277	277	277	70
53	53	53	70	128	128	128	70	203	203	203	70	278	278	278	70
54	54	54	70	129	129	129	70	204	204	204	70	279	279	279	70
55	55	55	70	130	130	130	70	205	205	205	70	280	280	280	70
56	56	56	70	131	131	131	70	206	206	206	70	281	281	281	70
57	57	57	70	132	132	132	70	207	207	207	70	282	282	282	70
58	58	58	70	133	133	133	70	208	208	208	70	283	283	283	70
59	59	59	70	134	134	134	70	209	209	209	70	284	284	284	70
60	60	60	70	135	135	135	70	210	210	210	70	285	285	285	70
61	61	61	70	136	136	136	70	211	211	211	70	286	286	286	70
62	62	62	70	137	137	137	70	212	212	212	70	287	287	287	70
63	63	63	70	138	138	138	70	213	213	213	70	288	288	288	70
64	64	64	70	139	139	139	70	214	214	214	70	289	289	289	70
65	65	65	70	140	140	140	70	215	215	215	70	290	290	290	70
66	66	66	70	141	141	141	70	216	216	216	70	291	291	291	70
67	67	67	70	142	142	142	70	217	217	217	70	292	292	292	70
68	68	68	70	143	143	143	70	218	218	218	70	293	293	293	70
69	69	69	70	144	144	144	70	219	219	219	70	294	294	294	70
70	70	70	70	145	145	145	70	220	220	220	70	295	295	295	70
71	71	71	70	146	146	146	70	221	221	221	70	296	296	296	70
72	72	72	70	147	147	147	70	222	222	222	70	297	297	297	70
73	73	73	70	148	148	148	70	223	223	223	70				
74	74	74	70	149	149	149	70	224	224	224	70				

***Study case* in11**

Kapasitas: 1000

Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
0	0	-21	0	75	9	-10	42	150	14	-5	29	225	12	-1	27	300	-4	3	31
1	1	-20	300	76	11	-10	39	151	16	-5	27	226	14	-1	25	301	-2	3	33
2	-2	-19	180	77	-12	-9	36	152	-17	-4	25	227	16	-1	24	302	0	3	36
3	0	-19	300	78	-10	-9	39	153	-15	-4	27	228	18	-1	23	303	2	3	33
4	2	-19	180	79	-8	-9	42	154	-13	-4	29	229	20	-1	21	304	4	3	31
5	-3	-18	128	80	-6	-9	47	155	-11	-4	31	230	-21	0	20	305	6	3	29
6	-1	-18	180	81	-4	-9	52	156	-9	-4	33	231	-19	0	21	306	8	3	27
7	1	-18	180	82	-2	-9	60	157	-7	-4	36	232	-17	0	23	307	10	3	25
8	3	-18	128	83	0	-9	69	158	-5	-4	39	233	-15	0	24	308	12	3	24
9	-4	-17	100	84	2	-9	60	159	-3	-4	42	234	-13	0	25	309	14	3	23
10	-2	-17	128	85	4	-9	52	160	-1	-4	47	235	-11	0	27	310	16	3	21
11	0	-17	180	86	6	-9	47	161	1	-4	47	236	-9	0	29	311	18	3	20
12	2	-17	128	87	8	-9	42	162	3	-4	42	237	-7	0	31	312	-17	4	20
13	4	-17	100	88	10	-9	39	163	5	-4	39	238	-5	0	33	313	-15	4	21
14	-5	-16	81	89	12	-9	36	164	7	-4	36	239	-3	0	36	314	-13	4	23
15	-3	-16	100	90	-13	-8	33	165	9	-4	33	240	-1	0	39	315	-11	4	24
16	-1	-16	128	91	-11	-8	36	166	11	-4	31	241	1	0	39	316	-9	4	25
17	1	-16	128	92	-9	-8	39	167	13	-4	29	242	3	0	36	317	-7	4	27
18	3	-16	100	93	-7	-8	42	168	15	-4	27	243	5	0	33	318	-5	4	29
19	5	-16	81	94	-5	-8	47	169	17	-4	25	244	7	0	31	319	-3	4	31
20	-6	-15	69	95	-3	-8	52	170	-18	-3	24	245	9	0	29	320	-1	4	33
21	-4	-15	81	96	-1	-8	60	171	-16	-3	25	246	11	0	27	321	1	4	33
22	-2	-15	100	97	1	-8	60	172	-14	-3	27	247	13	0	25	322	3	4	31
23	0	-15	128	98	3	-8	52	173	-12	-3	29	248	15	0	24				
24	2	-15	100	99	5	-8	47	174	-10	-3	31	249	17	0	23				
25	4	-15	81	100	7	-8	42	175	-8	-3	33	250	19	0	21				
26	6	-15	69	101	9	-8	39	176	-6	-3	36	251	21	0	20				
27	-7	-14	60	102	11	-8	36	177	-4	-3	39	252	-20	1	20				
28	-5	-14	69	103	13	-8	33	178	-2	-3	42	253	-18	1	21				
29	-3	-14	81	104	-14	-7	31	179	0	-3	47	254	-16	1	23				
30	-1	-14	100	105	-12	-7	33	180	2	-3	42	255	-14	1	24				
31	1	-14	100	106	-10	-7	36	181	4	-3	39	256	-12	1	25				
32	3	-14	81	107	-8	-7	39	182	6	-3	36	257	-10	1	27				
33	5	-14	69	108	-6	-7	42	183	8	-3	33	258	-8	1	29				
34	7	-14	60	109	-4	-7	47	184	10	-3	31	259	-6	1	31				
35	-8	-13	52	110	-2	-7	52	185	12	-3	29	260	-4	1	33				
36	-6	-13	60	111	0	-7	60	186	14	-3	27	261	-2	1	36				
37	-4	-13	69	112	2	-7	52	187	16	-3	25	262	0	1	39				
38	-2	-13	81	113	4	-7	47	188	18	-3	24	263	2	1	36				
39	0	-13	100	114	6	-7	42	189	-19	-2	23	264	4	1	33				
40	2	-13	81	115	8	-7	39	190	-17	-2	24	265	6	1	31				
41	4	-13	69	116	10	-7	36	191	-15	-2	25	266	8	1	29				
42	6	-13	60	117	12	-7	33	192	-13	-2	27	267	10	1	27				
43	8	-13	52	118	14	-7	31	193	-11	-2	29	268	12	1	25				
44	-9	-12	47	119	-15	-6	29	194	-9	-2	31	269	14	1	24				
45	-7	-12	52	120	-13	-6	31	195	-7	-2	33	270	16	1	23				
46	-5	-12	60	121	-11	-6	33	196	-5	-2	36	271	18	1	21				
47	-3	-12	69	122	-9	-6	36	197	-3	-2	39	272	20	1	20				
48	-1	-12	81	123	-7	-6	39	198	-1	-2	42	273	-19	2	20				
49	1	-12	81	124	-5	-6	42	199	1	-2	42	274	-17	2	21				
50	3	-12	69	125	-3	-6	47	200	3	-2	39	275	-15	2	23				
51	5	-12	60	126	-1	-6	52	201	5	-2	36	276	-13	2	24				
52	7	-12	52	127	1	-6	52	202	7	-2	33	277	-11	2	25				
53	9	-12	47	128	3	-6	47	203	9	-2	31	278	-9	2	27				
54	#	-11	42	129	5	-6	42	204	11	-2	29	279	-7	2	29				
55	-8	-11	47	130	7	-6	39	205	13	-2	27	280	-5	2	31				
56	-6	-11	52	131	9	-6	36	206	15	-2	25	281	-3	2	33				
57	-4	-11	60	132	11	-6	33	207	17	-2	24	282	-1	2	36				
58	-2	-11	69	133	13	-6	31	208	19	-2	23	283	1	2	36				
59	0	-11	81	134	15	-6	29	209	-20	-1	21	284	3	2	33				
60	2	-11	69	135	-16	-5	27	210	-18	-1	23	285	5	2	31				
61	4	-11	60	136	-14	-5	29	211	-16	-1	24	286	7	2	29				
62	6	-11	52	137	-12	-5	31	212	-14	-1	25	287	9	2	27				
63	8	-11	47	138	-10	-5	33	213	-12	-1	27	288	11	2	25				
64	10	-11	42	139	-8	-5	36	214	-10	-1	29	289	13	2	24				
65	-11	-10	39	140	-6	-5	39	215	-8	-1	31	290	15	2	23				
66	-9	-10	42	141	-4	-5	42	216	-6	-1	33	291	17	2	21				
67	-7	-10	47	142	-2	-5	47	217	-4	-1	36	292	19	2	20				
68	-5	-10	52	143	0	-5	52	218	-2	-1	39	293	-18	3	20				
69	-3	-10	60	144	2	-5	47	219	0	-1	42	294	-16	3	21				
70	-1	-10	69	145	4	-5	42	220	2	-1	39	295	-14	3	23				
71	1	-10	69	146	6	-5	39	221	4	-1	36	296	-12	3	24				
72	3	-10	60	147	8	-5	36	222	6	-1	33	297	-10	3	25				
73	5	-10	52	148	10	-5	33	223	8	-1	31	298	-8	3	27				
74	7	-10	47	149	12	-5	31	224	10	-1	29	299	-6	3	29				

***Study case* in12**

Kapasitas: 880

Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
0	-21	-9	0	75	-10	3	23	150	-3	4	60	225	3	-8	39	300	10	8	42
1	-20	-7	21	76	-10	5	21	151	-3	6	52	226	3	-6	36	301	10	10	39
2	-20	-5	20	77	-9	7	47	152	-3	8	47	227	3	-4	33	302	10	-13	36
3	-19	-3	23	78	-9	9	42	153	-3	-7	42	228	3	-2	31	303	10	-11	33
4	-19	-1	21	79	-9	-12	39	154	-3	-5	39	229	3	0	29	304	10	-9	31
5	-19	1	20	80	-9	-10	36	155	-3	-3	36	230	3	2	27	305	10	-7	29
6	-18	3	24	81	-9	-8	33	156	-3	-1	33	231	4	4	69	306	10	-5	27
7	-18	5	23	82	-9	-6	31	157	-3	1	31	232	4	6	60	307	10	-3	25
8	-18	7	21	83	-9	-4	29	158	-3	3	29	233	4	8	52	308	10	-1	24
9	-18	9	20	84	-9	-2	27	159	-3	5	27	234	4	-9	47	309	10	1	23
10	-17	-12	25	85	-9	0	25	160	-3	7	25	235	4	-7	42	310	10	3	21
11	-17	-10	24	86	-9	2	24	161	-2	-6	81	236	4	-5	39	311	11	5	39
12	-17	-8	23	87	-9	4	23	162	-2	-4	69	237	4	-3	36	312	11	7	36
13	-17	-6	21	88	-9	6	21	163	-2	-2	60	238	4	-1	33	313	11	9	33
14	-17	-4	20	89	-8	8	52	164	-2	0	52	239	4	1	31	314	11	-12	31
15	-16	-2	27	90	-8	-13	47	165	-2	2	47	240	4	3	29	315	11	-10	29
16	-16	0	25	91	-8	-11	42	166	-2	4	42	241	4	5	27	316	11	-8	27
17	-16	2	24	92	-8	-9	39	167	-2	6	39	242	4	7	25	317	11	-6	25
18	-16	4	23	93	-8	-7	36	168	-2	-5	36	243	5	9	60	318	11	-4	24
19	-16	6	21	94	-8	-5	33	169	-2	-3	33	244	5	-10	52	319	11	-2	23
20	-16	8	20	95	-8	-3	31	170	-2	-1	31	245	5	-8	47	320	11	0	21
21	-15	-13	29	96	-8	-1	29	171	-2	1	29	246	5	-6	42	321	12	2	36
22	-15	-11	27	97	-8	1	27	172	-2	3	27	247	5	-4	39	322	12	4	33
23	-15	-9	25	98	-8	3	25	173	-1	5	81	248	5	-2	36	323	12	6	31
24	-15	-7	24	99	-8	5	24	174	-1	-4	69	249	5	0	33	324	12	8	29
25	-15	-5	23	100	-8	7	23	175	-1	-2	60	250	5	2	31	325	12	10	27
26	-15	-3	21	101	-7	9	52	176	-1	0	52	251	5	4	29	326	12	-13	25
27	-15	-1	20	102	-7	-12	47	177	-1	2	47	252	5	6	27	327	12	-11	24
28	-14	1	31	103	-7	-10	42	178	-1	4	42	253	5	8	25	328	12	-9	23
29	-14	3	29	104	-7	-8	39	179	-1	-3	39	254	6	10	60	329	12	-7	21
30	-14	5	27	105	-7	-6	36	180	-1	-1	36	255	6	-11	52	330	12	-5	20
31	-14	7	25	106	-7	-4	33	181	-1	1	33	256	6	-9	47	331	13	-3	33
32	-14	9	24	107	-7	-2	31	182	-1	3	31	257	6	-7	42	332	13	-1	31
33	-14	-12	23	108	-7	0	29	183	-1	-2	29	258	6	-5	39	333	13	1	29
34	-14	-10	21	109	-7	2	27	184	-1	0	27	259	6	-3	36	334	13	3	27
35	-14	-8	20	110	-7	4	25	185	0	2	100	260	6	-1	33	335	13	5	25
36	-13	-6	33	111	-7	6	24	186	0	-1	81	261	6	1	31	336	13	7	24
37	-13	-4	31	112	-7	8	23	187	0	1	69	262	6	3	29	337	13	9	23
38	-13	-2	29	113	-6	-11	60	188	0	0	60	263	6	5	27	338	13	-12	21
39	-13	0	27	114	-6	-9	52	189	0	0	52	264	6	7	25	339	13	-10	20
40	-13	2	25	115	-6	-7	47	190	0	-1	47	265	6	9	24	340	14	-8	31
41	-13	4	24	116	-6	-5	42	191	0	1	42	266	7	-12	52	341	14	-6	29
42	-13	6	23	117	-6	-3	39	192	0	-2	39	267	7	-10	47	342	14	-4	27
43	-13	8	21	118	-6	-1	36	193	0	0	36	268	7	-8	42	343	14	-2	25
44	-13	-13	20	119	-6	1	33	194	0	2	33	269	7	-6	39	344	14	0	24
45	-12	-11	36	120	-6	3	31	195	0	-3	31	270	7	-4	36	345	14	2	23
46	-12	-9	33	121	-6	5	29	196	0	-1	29	271	7	-2	33	346	14	4	21
47	-12	-7	31	122	-6	7	27	197	1	1	81	272	7	0	31	347	14	6	20
48	-12	-5	29	123	-6	9	25	198	1	3	69	273	7	2	29	348	15	8	29
49	-12	-3	27	124	-6	-10	24	199	1	-4	60	274	7	4	27	349	15	10	27
50	-12	-1	25	125	-5	-8	60	200	1	-2	52	275	7	6	25	350	15	-13	25
51	-12	1	24	126	-5	-6	52	201	1	0	47	276	7	8	24	351	15	-11	24
52	-12	3	23	127	-5	-4	47	202	1	2	42	277	8	10	52	352	15	-9	23
53	-12	5	21	128	-5	-2	42	203	1	4	39	278	8	-13	47	353	15	-7	21
54	-12	7	20	129	-5	0	39	204	1	-5	36	279	8	-11	42	354	15	-5	20
55	-11	9	39	130	-5	2	36	205	1	-3	33	280	8	-9	39	355	16	-3	27
56	-11	-12	36	131	-5	4	33	206	1	-1	31	281	8	-7	36	356	16	-1	25
57	-11	-10	33	132	-5	6	31	207	1	1	29	282	8	-5	33	357	16	1	24
58	-11	-8	31	133	-5	8	29	208	2	3	81	283	8	-3	31	358	16	3	23
59	-11	-6	29	134	-5	-9	27	209	2	5	69	284	8	-1	29	359	16	5	21
60	-11	-4	27	135	-5	-7	25	210	2	-6	60	285	8	1	27	360	16	7	20
61	-11	-2	25	136	-5	-5	24	211	2	-4	52	286	8	3	25	361	17	9	25
62	-11	0	24	137	-4	-3	69	212	2	-2	47	287	8	5	24	362	17	-12	24
63	-11	2	23	138	-4	-1	60	213	2	0	42	288	8	7	23	363	17	-10	23
64	-11	4	21	139	-4	1	52	214	2	2	39	289	9	9	47	364	17	-8	21
65	-11	6	20	140	-4	3	47	215	2	4	36	290	9	-12	42	365	17	-6	20
66	-10	8	42	141	-4	5	42	216	2	6	33	291	9	-10	39	366	18	-4	24
67	-10	-13	39	142	-4	7	39	217	2	-7	31	292	9	-8	36	367	18	-2	23
68	-10	-11	36	143	-4	9	36	218	2	-5	29	293	9	-6	33	368	18	0	21
69	-10	-9	33	144	-4	-8	33	219	2	-3	27	294	9	-4	31	369	18	2	20
70	-10	-7	31	145	-4	-6	31	220	3	-1	69	295	9	-2	29	370	19	4	23
71	-10	-5	29	146	-4	-4	29	221	3	1	60	296	9	0	27	371	19	6	21
72	-10	-3	27	147	-4	-2	27	222	3	3	52	297	9	2	25	372	19	8	20
73	-10	-1	25	148	-4	0	25	223	3	5	47	298	9	4	24	373	20	10	21
74	-10	1	24	149	-3	2	69	224	3	7	42	299	9	6	23	374	20	-13	20
																375	21	-11	20

***Study case* in13**

Kapasitas: 200

Kota	x	y	dem	Kota	x	y	dem	Kota	x	y	dem	Kota	x	y	dem
0	0	0	0	75	9.0211	-3.125	20	150	-3.3829	9.7656	10	225	3.9467	-9.7656	10
1	2.8867	5	40	76	9.9232	-4.6875	10	151	-2.2553	11.7187	20	226	-8.4573	-9.7656	10
2	-2.8868	5	40	77	8.119	1.5625	10	152	-1.1276	13.6719	10	227	-11.2764	-9.7656	20
3	-5.7735	0	40	78	9.0211	3.125	20	153	-6.7658	7.8125	10	228	-14.0955	-9.7656	10
4	-2.8867	-5	40	79	9.9232	4.6875	10	154	-9.0211	7.8125	20	229	-4.2286	-12.207	10
5	2.8868	-5	40	80	2.5259	6.25	10	155	-11.2764	7.8125	10	230	-2.8191	-14.6484	20
6	8.6603	5	40	81	-2.5259	6.25	10	156	-8.3445	-1.1719	10	231	-1.4095	-17.0898	10
7	0	10	40	82	5.4127	6.25	10	157	-5.1871	-6.6406	10	232	6.4839	-8.3008	10
8	-8.6603	5	40	83	7.2169	6.25	20	158	-10.1487	1.9531	10	233	10.4306	-1.4648	10
9	-8.6603	-5	40	84	9.0211	6.25	10	159	-11.2764	3.9062	20	234	4.2286	-12.207	10
10	0	-10	40	85	2.7063	7.8125	10	160	-12.404	5.8594	10	235	2.8191	-14.6484	20
11	8.6603	-5	40	86	1.8042	9.375	20	161	-10.1487	-1.9531	10	236	1.4095	-17.0898	10
12	5.3405	0.75	10	87	0.9021	10.9375	10	162	-11.2764	-3.9063	20	237	8.4573	-9.7656	10
13	3.3198	4.25	10	88	-4.1497	5.3125	10	163	-12.404	-5.8594	10	238	11.2764	-9.7656	20
14	6.4952	-1.25	10	89	-6.6756	0.9375	10	164	-3.1574	-7.8125	10	239	14.0955	-9.7656	10
15	7.2169	-2.5	20	90	-2.7063	7.8125	10	165	3.1574	-7.8125	10	240	14.0955	0	40
16	7.9386	-3.75	10	91	-1.8042	9.375	20	166	-6.7658	-7.8125	10	241	7.0477	12.207	40
17	6.4952	1.25	10	92	-0.9021	10.9375	10	167	-9.0211	-7.8125	20	242	-7.0477	12.207	40
18	7.2169	2.5	20	93	-5.4127	6.25	10	168	-11.2764	-7.8125	10	243	-14.0955	0	40
19	7.9386	3.75	10	94	-7.2169	6.25	20	169	-3.3829	-9.7656	10	244	-7.0477	-12.207	40
20	2.0207	5	10	95	-9.0211	6.25	10	170	-2.2553	-11.7187	20	245	7.0477	-12.207	40
21	-2.0207	5	10	96	-6.6756	-0.9375	10	171	-1.1276	-13.6719	10	246	21.1432	12.207	40
22	4.3301	5	10	97	-4.1497	-5.3125	10	172	5.1871	-6.6406	10	247	0	24.4141	40
23	5.7735	5	20	98	-8.119	1.5625	10	173	8.3445	-1.1719	10	248	-21.1432	12.207	40
24	7.2169	5	10	99	-9.0211	3.125	20	174	3.3829	-9.7656	10	249	-21.1432	-12.207	40
25	2.1651	6.25	10	100	-9.9232	4.6875	10	175	2.2553	-11.7187	20	250	0	-24.4141	40
26	1.4434	7.5	20	101	-8.119	-1.5625	10	176	1.1276	-13.6719	10	251	21.1432	-12.207	40
27	0.7217	8.75	10	102	-9.0211	-3.125	20	177	6.7658	-7.8125	10	252	13.0383	1.8311	10
28	-3.3198	4.25	10	103	-9.9232	-4.6875	10	178	9.0211	-7.8125	20	253	8.1049	10.376	10
29	-5.3405	0.75	10	104	-2.5259	-6.25	10	179	11.2764	-7.8125	10	254	15.8574	-3.0518	10
30	-2.1651	6.25	10	105	2.5259	-6.25	10	180	11.2764	0	40	255	17.6193	-6.1035	20
31	-1.4434	7.5	20	106	-5.4127	-6.25	10	181	5.6382	9.7656	40	256	19.3813	-9.1553	10
32	-0.7217	8.75	10	107	-7.2169	-6.25	20	182	-5.6382	9.7656	40	257	15.8574	3.0518	10
33	-4.3301	5	10	108	-9.0211	-6.25	10	183	-11.2764	0	40	258	17.6193	6.1035	20
34	-5.7735	5	20	109	-2.7063	-7.8125	10	184	-5.6382	-9.7656	40	259	19.3813	9.1553	10
35	-7.2169	5	10	110	-1.8042	-9.375	20	185	5.6382	-9.7656	40	260	4.9334	12.207	10
36	-5.3405	-0.75	10	111	-0.9021	-10.9375	10	186	16.9146	9.7656	40	261	-4.9334	12.207	10
37	-3.3198	-4.25	10	112	4.1497	-5.3125	10	187	0	19.5312	40	262	10.5716	12.207	10
38	-6.4952	1.25	10	113	6.6756	-0.9375	10	188	-16.9146	9.7656	40	263	14.0955	12.207	20
39	-7.2169	2.5	20	114	2.7063	-7.8125	10	189	-16.9146	-9.7656	40	264	17.6193	12.207	10
40	-7.9386	3.75	10	115	1.8042	-9.375	20	190	0	-19.5312	40	265	5.2858	15.2588	10
41	-6.4952	-1.25	10	116	0.9021	-10.9375	10	191	16.9146	-9.7656	40	266	3.5239	18.3105	20
42	-7.2169	-2.5	20	117	5.4127	-6.25	10	192	10.4306	1.4648	10	267	1.7619	21.3623	10
43	-7.9386	-3.75	10	118	7.2169	-6.25	20	193	6.4839	8.3008	10	268	-8.1049	10.376	10
44	-2.0207	-5	10	119	9.0211	-6.25	10	194	12.6859	-2.4414	10	269	-13.0383	1.8311	10
45	2.0207	-5	10	120	9.0211	0	40	195	14.0955	-4.8828	20	270	-5.2858	15.2588	10
46	-4.3301	-5	10	121	4.5105	7.8125	40	196	15.505	-7.3242	10	271	-3.5239	18.3105	20
47	-5.7735	-5	20	122	-4.5105	7.8125	40	197	12.6859	2.4414	10	272	-1.7619	21.3623	10
48	-7.2169	-5	10	123	-9.0211	0	40	198	14.0955	4.8828	20	273	-10.5716	12.207	10
49	-2.1651	-6.25	10	124	-4.5105	-7.8125	40	199	15.505	7.3242	10	274	-14.0955	12.207	20
50	-1.4434	-7.5	20	125	4.5106	-7.8125	40	200	3.9467	9.7656	10	275	-17.6193	12.207	10
51	-0.7217	-8.75	10	126	13.5316	7.8125	40	201	-3.9467	9.7656	10	276	-13.0383	-1.8311	10
52	3.3198	-4.25	10	127	0	15.625	40	202	8.4573	9.7656	10	277	-8.1049	-10.376	10
53	5.3405	-0.75	10	128	-13.5316	7.8125	40	203	11.2764	9.7656	20	278	-15.8574	3.0518	10
54	2.1651	-6.25	10	129	-13.5316	-7.8125	40	204	14.0955	9.7656	10	279	-17.6193	6.1035	20
55	1.4434	-7.5	20	130	0	-15.625	40	205	4.2286	12.207	10	280	-19.3813	9.1553	10
56	0.7217	-8.75	10	131	13.5316	-7.8125	40	206	2.8191	14.6484	20	281	-15.8574	-3.0518	10
57	4.3301	-5	10	132	8.3445	1.1719	10	207	1.4095	17.0898	10	282	-17.6193	-6.1035	20
58	5.7735	-5	20	133	5.1871	6.6406	10	208	-6.4839	8.3008	10	283	-19.3813	-9.1553	10
59	7.2169	-5	10	134	10.1487	-1.9531	10	209	-10.4306	1.4648	10	284	-4.9334	-12.207	10
60	7.2169	0	40	135	11.2764	-3.9062	20	210	-4.2286	12.207	10	285	4.9334	-12.207	10
61	3.6084	6.25	40	136	12.404	-5.8594	10	211	-2.8191	14.6484	20	286	-10.5716	-12.207	10
62	-3.6084	6.25	40	137	10.1487	1.9531	10	212	-1.4095	17.0898	10	287	-14.0955	-12.207	20
63	-7.2169	0	40	138	11.2764	3.9062	20	213	-8.4573	9.7656	10	288	-17.6193	-12.207	10
64	-3.6084	-6.25	40	139	12.404	5.8594	10	214	-11.2764	9.7656	20	289	-5.2858	-15.2588	10
65	3.6084	-6.25	40	140	3.1574	7.8125	10	215	-14.0955	9.7656	10	290	-3.5239	-18.3105	20
66	10.8253	6.25	40	141	-3.1574	7.8125	10	216	-10.4306	-1.4648	10	291	-1.7619	-21.3623	10
67	0	12.5	40	142	6.7658	7.8125	10	217	-6.4839	-8.3008	10	292	8.1049	-10.376	10
68	-10.8253	6.25	40	143	9.0211	7.8125	20	218	-12.6859	2.4414	10	293	13.0383	-1.8311	10
69	-10.8253	-6.25	40	144	11.2764	7.8125	10	219	-14.0955	4.8828	20	294	5.2858	-15.2588	10
70	0	-12.5	40	145	3.3829	9.7656	10	220	-15.505	7.3242	10	295	3.5239	-18.3105	20
71	10.8253	-6.25	40	146	2.2553	11.7187	20	221	-12.6859	-2.4414	10	296	1.7619	-21.3623	10
72	6.6756	0.9375	10	147	1.1276	13.6719	10	222	-14.0955	-4.8828	20	297	10.5716	-12.207	10
73	4.1497	5.3125	10	148	-5.1871	6.6406	10	223	-15.505	-7.3242	10	298	14.0955	-12.207	20
74	8.119	-1.5625	10	149	-8.3445	1.1719	10	224	-3.9467	-9.7656	10	299	17.6193	-12.207	10

Kota	x	y	dem	Kota	x	y	dem	Kota	x	y	dem	Kota	x	y	dem
300	17.6193	0	40	331	-4.4048	22.8882	20	362	-11.0121	19.0735	40	393	-16.5181	19.0735	10
301	8.8097	15.2588	40	332	-2.2024	26.7029	10	363	-22.0242	0	40	394	-22.0242	19.0735	20
302	-8.8097	15.2588	40	333	-13.2145	15.2588	10	364	-11.0121	-19.0735	40	395	-27.5302	19.0735	10
303	-17.6193	0	40	334	-17.6193	15.2588	20	365	11.0121	-19.0735	40	396	-20.3723	-2.861	10
304	-8.8097	-15.2588	40	335	-22.0242	15.2588	10	366	33.0362	19.0735	40	397	-12.6639	-16.2125	10
305	8.8097	-15.2588	40	336	-16.2979	-2.2888	10	367	0	38.147	40	398	-24.7772	4.7684	10
306	26.429	15.2588	40	337	-10.1311	-12.97	10	368	-33.0362	19.0735	40	399	-27.5302	9.5367	20
307	0	30.5176	40	338	-19.8217	3.8147	10	369	-33.0362	-19.0735	40	400	-30.2832	14.3051	10
308	-26.429	15.2588	40	339	-22.0242	7.6294	20	370	0	-38.147	40	401	-24.7772	-4.7684	10
309	-26.429	-15.2588	40	340	-24.2266	11.4441	10	371	33.0363	-19.0735	40	402	-27.5302	-9.5367	20
310	0	-30.5176	40	341	-19.8217	-3.8147	10	372	20.3723	2.861	10	403	-30.2832	-14.3051	10
311	26.429	-15.2588	40	342	-22.0242	-7.6294	20	373	12.6639	16.2125	10	404	-7.7085	-19.0735	10
312	16.2979	2.2888	10	343	-24.2266	-11.4441	10	374	24.7772	-4.7684	10	405	7.7085	-19.0735	10
313	10.1311	12.97	10	344	-6.1668	-15.2588	10	375	27.5302	-9.5367	20	406	-16.5181	-19.0735	10
314	19.8217	-3.8147	10	345	6.1668	-15.2588	10	376	30.2832	-14.3051	10	407	-22.0242	-19.0735	20
315	22.0242	-7.6294	20	346	-13.2145	-15.2588	10	377	24.7772	4.7684	10	408	-27.5302	-19.0735	10
316	24.2266	-11.4441	10	347	-17.6193	-15.2588	20	378	27.5302	9.5367	20	409	-8.2591	-23.8419	10
317	19.8217	3.8147	10	348	-22.0242	-15.2588	10	379	30.2832	14.3051	10	410	-5.506	-28.6102	20
318	22.0242	7.6294	20	349	-6.6072	-19.0735	10	380	7.7085	19.0735	10	411	-2.753	-33.3786	10
319	24.2266	11.4441	10	350	-4.4048	-22.8882	20	381	-7.7085	19.0735	10	412	12.6639	-16.2125	10
320	6.1668	15.2588	10	351	-2.2024	-26.7029	10	382	16.5181	19.0735	10	413	20.3723	-2.861	10
321	-6.1668	15.2588	10	352	10.1311	-12.97	10	383	22.0242	19.0735	20	414	8.2591	-23.8418	10
322	13.2145	15.2588	10	353	16.2979	-2.2888	10	384	27.5302	19.0735	10	415	5.506	-28.6102	20
323	17.6193	15.2588	20	354	6.6073	-19.0735	10	385	8.2591	23.8419	10	416	2.753	-33.3786	10
324	22.0242	15.2588	10	355	4.4048	-22.8882	20	386	5.506	28.6102	20	417	16.5181	-19.0735	10
325	6.6072	19.0735	10	356	2.2024	-26.7029	10	387	2.753	33.3786	10	418	22.0242	-19.0735	20
326	4.4048	22.8882	20	357	13.2145	-15.2588	10	388	-12.6639	16.2125	10	419	27.5302	-19.0735	10
327	2.2024	26.7029	10	358	17.6193	-15.2588	20	389	-20.3723	2.861	10	420	5.7735	0	40
328	-10.1311	12.97	10	359	22.0242	-15.2588	10	390	-8.2591	23.8419	10				
329	-16.2979	2.2888	10	360	22.0242	0	40	391	-5.506	28.6102	20				
330	-6.6072	19.0735	10	361	11.0121	19.0735	40	392	-2.753	33.3786	10				

***Study case* in14**

Kapasitas: 1000

Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
0	0	-21	0	41	4	-13	69	82	-2	-9	60	123	-7	-6	39	164	7	-4	36
1	1	-20	300	42	6	-13	60	83	0	-9	69	124	-5	-6	42	165	9	-4	33
2	-2	-19	180	43	8	-13	52	84	2	-9	60	125	-3	-6	47	166	11	-4	31
3	0	-19	300	44	-9	-12	47	85	4	-9	52	126	-1	-6	52	167	13	-4	29
4	2	-19	180	45	-7	-12	52	86	6	-9	47	127	1	-6	52	168	15	-4	27
5	-3	-18	128	46	-5	-12	60	87	8	-9	42	128	3	-6	47	169	17	-4	25
6	-1	-18	180	47	-3	-12	69	88	10	-9	39	129	5	-6	42	170	-18	-3	24
7	1	-18	180	48	-1	-12	81	89	12	-9	36	130	7	-6	39	171	-16	-3	25
8	3	-18	128	49	1	-12	81	90	-13	-8	33	131	9	-6	36	172	-14	-3	27
9	-4	-17	100	50	3	-12	69	91	-11	-8	36	132	11	-6	33	173	-12	-3	29
10	-2	-17	128	51	5	-12	60	92	-9	-8	39	133	13	-6	31	174	-10	-3	31
11	0	-17	180	52	7	-12	52	93	-7	-8	42	134	15	-6	29	175	-8	-3	33
12	2	-17	128	53	9	-12	47	94	-5	-8	47	135	-16	-5	27	176	-6	-3	36
13	4	-17	100	54	-10	-11	42	95	-3	-8	52	136	-14	-5	29	177	-4	-3	39
14	-5	-16	81	55	-8	-11	47	96	-1	-8	60	137	-12	-5	31	178	-2	-3	42
15	-3	-16	100	56	-6	-11	52	97	1	-8	60	138	-10	-5	33	179	0	-3	47
16	-1	-16	128	57	-4	-11	60	98	3	-8	52	139	-8	-5	36	180	2	-3	42
17	1	-16	128	58	-2	-11	69	99	5	-8	47	140	-6	-5	39	181	4	-3	39
18	3	-16	100	59	0	-11	81	100	7	-8	42	141	-4	-5	42	182	6	-3	36
19	5	-16	81	60	2	-11	69	101	9	-8	39	142	-2	-5	47	183	8	-3	33
20	-6	-15	69	61	4	-11	60	102	11	-8	36	143	0	-5	52	184	10	-3	31
21	-4	-15	81	62	6	-11	52	103	13	-8	33	144	2	-5	47	185	12	-3	29
22	-2	-15	100	63	8	-11	47	104	-14	-7	31	145	4	-5	42	186	14	-3	27
23	0	-15	128	64	10	-11	42	105	-12	-7	33	146	6	-5	39	187	16	-3	25
24	2	-15	100	65	-11	-10	39	106	-10	-7	36	147	8	-5	36	188	18	-3	24
25	4	-15	81	66	-9	-10	42	107	-8	-7	39	148	10	-5	33	189	-19	-2	23
26	6	-15	69	67	-7	-10	47	108	-6	-7	42	149	12	-5	31	190	-17	-2	24
27	-7	-14	60	68	-5	-10	52	109	-4	-7	47	150	14	-5	29	191	-15	-2	25
28	-5	-14	69	69	-3	-10	60	110	-2	-7	52	151	16	-5	27	192	-13	-2	27
29	-3	-14	81	70	-1	-10	69	111	0	-7	60	152	-17	-4	25	193	-11	-2	29
30	-1	-14	100	71	1	-10	69	112	2	-7	52	153	-15	-4	27	194	-9	-2	31
31	1	-14	100	72	3	-10	60	113	4	-7	47	154	-13	-4	29	195	-7	-2	33
32	3	-14	81	73	5	-10	52	114	6	-7	42	155	-11	-4	31	196	-5	-2	36
33	5	-14	69	74	7	-10	47	115	8	-7	39	156	-9	-4	33	197	-3	-2	39
34	7	-14	60	75	9	-10	42	116	10	-7	36	157	-7	-4	36	198	-1	-2	42
35	-8	-13	52	76	11	-10	39	117	12	-7	33	158	-5	-4	39	199	1	-2	42
36	-6	-13	60	77	-12	-9	36	118	14	-7	31	159	-3	-4	42	200	3	-2	39
37	-4	-13	69	78	-10	-9	39	119	-15	-6	29	160	-1	-4	47	201	5	-2	36
38	-2	-13	81	79	-8	-9	42	120	-13	-6	31	161	1	-4	47	202	7	-2	33
39	0	-13	100	80	-6	-9	47	121	-11	-6	33	162	3	-4	42	203	9	-2	31
40	2	-13	81	81	-4	-9	52	122	-9	-6	36	163	5	-4	39	204	11	-2	29

Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
205	13	-2	27	261	-2	1	36	317	-7	4	27	373	6	7	25	429	-7	12	21
206	15	-2	25	262	0	1	39	318	-5	4	29	374	8	7	24	430	-5	12	23
207	17	-2	24	263	2	1	36	319	-3	4	31	375	10	7	23	431	-3	12	24
208	19	-2	23	264	4	1	33	320	-1	4	33	376	12	7	21	432	-1	12	25
209	-20	-1	21	265	6	1	31	321	1	4	33	377	14	7	20	433	1	12	25
210	-18	-1	23	266	8	1	29	322	3	4	31	378	-13	8	20	434	3	12	24
211	-16	-1	24	267	10	1	27	323	5	4	29	379	-11	8	21	435	5	12	23
212	-14	-1	25	268	12	1	25	324	7	4	27	380	-9	8	23	436	7	12	21
213	-12	-1	27	269	14	1	24	325	9	4	25	381	-7	8	24	437	9	12	20
214	-10	-1	29	270	16	1	23	326	11	4	24	382	-5	8	25	438	-8	13	20
215	-8	-1	31	271	18	1	21	327	13	4	23	383	-3	8	27	439	-6	13	21
216	-6	-1	33	272	20	1	20	328	15	4	21	384	-1	8	29	440	-4	13	23
217	-4	-1	36	273	-19	2	20	329	17	4	20	385	1	8	29	441	-2	13	24
218	-2	-1	39	274	-17	2	21	330	-16	5	20	386	3	8	27	442	0	13	25
219	0	-1	42	275	-15	2	23	331	-14	5	21	387	5	8	25	443	2	13	24
220	2	-1	39	276	-13	2	24	332	-12	5	23	388	7	8	24	444	4	13	23
221	4	-1	36	277	-11	2	25	333	-10	5	24	389	9	8	23	445	6	13	21
222	6	-1	33	278	-9	2	27	334	-8	5	25	390	11	8	21	446	8	13	20
223	8	-1	31	279	-7	2	29	335	-6	5	27	391	13	8	20	447	-7	14	20
224	10	-1	29	280	-5	2	31	336	-4	5	29	392	-12	9	20	448	-5	14	21
225	12	-1	27	281	-3	2	33	337	-2	5	31	393	-10	9	21	449	-3	14	23
226	14	-1	25	282	-1	2	36	338	0	5	33	394	-8	9	23	450	-1	14	24
227	16	-1	24	283	1	2	36	339	2	5	31	395	-6	9	24	451	1	14	24
228	18	-1	23	284	3	2	33	340	4	5	29	396	-4	9	25	452	3	14	23
229	20	-1	21	285	5	2	31	341	6	5	27	397	-2	9	27	453	5	14	21
230	-21	0	20	286	7	2	29	342	8	5	25	398	0	9	29	454	7	14	20
231	-19	0	21	287	9	2	27	343	10	5	24	399	2	9	27	455	-6	15	20
232	-17	0	23	288	11	2	25	344	12	5	23	400	4	9	25	456	-4	15	21
233	-15	0	24	289	13	2	24	345	14	5	21	401	6	9	24	457	-2	15	23
234	-13	0	25	290	15	2	23	346	16	5	20	402	8	9	23	458	0	15	24
235	-11	0	27	291	17	2	21	347	-15	6	20	403	10	9	21	459	2	15	23
236	-9	0	29	292	19	2	20	348	-13	6	21	404	12	9	20	460	4	15	21
237	-7	0	31	293	-18	3	20	349	-11	6	23	405	-11	10	20	461	6	15	20
238	-5	0	33	294	-16	3	21	350	-9	6	24	406	-9	10	21	462	-5	16	20
239	-3	0	36	295	-14	3	23	351	-7	6	25	407	-7	10	23	463	-3	16	21
240	-1	0	39	296	-12	3	24	352	-5	6	27	408	-5	10	24	464	-1	16	23
241	1	0	39	297	-10	3	25	353	-3	6	29	409	-3	10	25	465	1	16	23
242	3	0	36	298	-8	3	27	354	-1	6	31	410	-1	10	27	466	3	16	21
243	5	0	33	299	-6	3	29	355	1	6	31	411	1	10	27	467	5	16	20
244	7	0	31	300	-4	3	31	356	3	6	29	412	3	10	25	468	-4	17	20
245	9	0	29	301	-2	3	33	357	5	6	27	413	5	10	24	469	-2	17	21
246	11	0	27	302	0	3	36	358	7	6	25	414	7	10	23	470	0	17	23
247	13	0	25	303	2	3	33	359	9	6	24	415	9	10	21	471	2	17	21
248	15	0	24	304	4	3	31	360	11	6	23	416	11	10	20	472	4	17	20
249	17	0	23	305	6	3	29	361	13	6	21	417	-10	11	20	473	-3	18	20
250	19	0	21	306	8	3	27	362	15	6	20	418	-8	11	21	474	-1	18	21
251	21	0	20	307	10	3	25	363	-14	7	20	419	-6	11	23	475	1	18	21
252	-20	1	20	308	12	3	24	364	-12	7	21	420	-4	11	24	476	3	18	20
253	-18	1	21	309	14	3	23	365	-10	7	23	421	-2	11	25	477	-2	19	20
254	-16	1	23	310	16	3	21	366	-8	7	24	422	0	11	27	478	0	19	21
255	-14	1	24	311	18	3	20	367	-6	7	25	423	2	11	25	479	2	19	20
256	-12	1	25	312	-17	4	20	368	-4	7	27	424	4	11	24	480	-1	20	20
257	-10	1	27	313	-15	4	21	369	-2	7	29	425	6	11	23	481	1	20	20
258	-8	1	29	314	-13	4	23	370	0	7	31	426	8	11	21	482	0	21	20
259	-6	1	31	315	-11	4	24	371	2	7	29	427	10	11	20	483	-1	-20	300
260	-4	1	33	316	-9	4	25	372	4	7	27	428	-9	12	20				

***Study case* in15**

Kapasitas: 977

Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
0	-33.0362	-19.0735	0	16	-22.0242	-7.6294	21	32	-17.6193	0	42	48	-14.0955	-4.8828	21
1	-33.0362	19.0735	42	17	-22.0242	0	42	33	-17.6193	6.1035	21	49	-14.0955	0	42
2	-30.2832	-14.3051	13	18	-22.0242	7.6294	21	34	-17.6193	12.207	13	50	-14.0955	4.8828	21
3	-30.2832	14.3051	13	19	-22.0242	15.2588	13	35	-17.6193	15.2588	21	51	-14.0955	9.7656	13
4	-27.5302	-19.0735	13	20	-22.0242	19.0735	21	36	-16.9146	-9.7656	42	52	-13.5316	-7.8125	42
5	-27.5302	-9.5367	21	21	-21.1432	-12.207	42	37	-16.9146	9.7656	42	53	-13.5316	7.8125	42
6	-27.5302	9.5367	21	22	-21.1432	12.207	42	38	-16.5181	-19.0735	13	54	-13.2145	-15.2588	13
7	-27.5302	19.0735	13	23	-20.3723	-2.861	13	39	-16.5181	19.0735	13	55	-13.2145	15.2588	13
8	-26.429	-15.2588	42	24	-20.3723	2.861	13	40	-16.2979	-2.2888	13	56	-13.0383	-1.8311	13
9	-26.429	15.2588	42	25	-19.8217	-3.8147	13	41	-16.2979	2.2888	13	57	-13.0383	1.8311	13
10	-24.7772	-4.7684	13	26	-19.8217	3.8147	13	42	-15.8574	-3.0518	13	58	-12.6859	-2.4414	13
11	-24.7772	4.7684	13	27	-19.3813	-9.1553	13	43	-15.8574	3.0518	13	59	-12.6859	2.4414	13
12	-24.2266	-11.4441	13	28	-19.3813	9.1553	13	44	-15.505	-7.3242	13	60	-12.6639	-16.2125	13
13	-24.2266	11.4441	13	29	-17.6193	-15.2588	21	45	-15.505	7.3242	13	61	-12.6639	16.2125	13
14	-22.0242	-19.0735	21	30	-17.6193	-12.207	13	46	-14.0955	-12.207	21	62	-12.404	-5.8594	13
15	-22.0242	-15.2588	13	31	-17.6193	-6.1035	21	47	-14.0955	-9.7656	13	63	-12.404	5.8594	13

Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
64	-11.2764	-9.7656	21	139	-4.4048	22.8882	21	214	2	1	36	289	4	15	21
65	-11.2764	-7.8125	13	140	-4.3301	-5	13	215	2	3	33	290	4	17	21
66	-11.2764	-3.9063	21	141	-4.3301	5	13	216	2	5	31	291	4.1497	-5.3125	13
67	-11.2764	0	42	142	-4.2286	-12.207	13	217	2	7	29	292	4.1497	5.3125	13
68	-11.2764	3.9062	21	143	-4.2286	12.207	13	218	2	9	27	293	4.2286	-12.207	13
69	-11.2764	7.8125	13	144	-4.1497	-5.3125	13	219	2	11	25	294	4.2286	12.207	13
70	-11.2764	9.7656	21	145	-4.1497	5.3125	13	220	2	13	24	295	4.3301	-5	13
71	-11.0121	-19.0735	42	146	-3.9467	-9.7656	13	221	2	15	23	296	4.3301	5	13
72	-11.0121	19.0735	42	147	-3.9467	9.7656	13	222	2	17	21	297	4.4048	-22.8882	21
73	-10.8253	-6.25	42	148	-3.6084	-6.25	42	223	2	19	21	298	4.4048	22.8882	21
74	-10.8253	6.25	42	149	-3.6084	6.25	42	224	2.0207	-5	13	299	4.5105	7.8125	42
75	-10.4306	1.4648	13	150	-3.5239	-18.3105	21	225	2.0207	5	13	300	4.5106	-7.8125	42
76	-10.1487	-1.9531	13	151	-3.5239	18.3105	21	226	2.1651	-6.25	13	301	4.9334	-12.207	13
77	-10.1487	1.9531	13	152	-3.3829	-9.7656	13	227	2.1651	6.25	13	302	4.9334	12.207	13
78	-10.1311	-12.97	13	153	-3.3829	9.7656	13	228	2.2024	-26.7029	13	303	5	-16	81
79	-10.1311	12.97	13	154	-3.3198	-4.25	13	229	2.2024	26.7029	13	304	5	-14	69
80	-9.9232	-4.6875	13	155	-3.3198	4.25	13	230	2.2553	-11.7187	21	305	5	-12	60
81	-9.9232	4.6875	13	156	-3.1574	-7.8125	13	231	2.2553	11.7187	21	306	5	-10	52
82	-9.0211	-7.8125	21	157	-3.1574	7.8125	13	232	2.5259	-6.25	13	307	5	-8	47
83	-9.0211	0	42	158	-2.8868	5	42	233	2.5259	6.25	13	308	5	-6	42
84	-9.0211	3.125	21	159	-2.8867	-5	42	234	2.7063	-7.8125	13	309	5	-4	39
85	-9.0211	6.25	13	160	-2.8191	-14.6484	21	235	2.7063	7.8125	13	310	5	-2	36
86	-9.0211	7.8125	21	161	-2.8191	14.6484	21	236	2.753	-33.3786	13	311	5	0	33
87	-8.8097	-15.2588	42	162	-2.753	-33.3786	13	237	2.753	33.3786	13	312	5	2	31
88	-8.8097	15.2588	42	163	-2.753	33.3786	13	238	2.8191	-14.6484	21	313	5	4	29
89	-8.6603	-5	42	164	-2.7063	-7.8125	13	239	2.8191	14.6484	21	314	5	6	27
90	-8.6603	5	42	165	-2.7063	7.8125	13	240	2.8867	5	35	315	5	8	25
91	-8.4573	-9.7656	13	166	-2.5259	-6.25	13	241	2.8868	-5	42	316	5	10	24
92	-8.4573	9.7656	13	167	-2.5259	6.25	13	242	3	-18	128	317	5	12	23
93	-8.3445	-1.1719	13	168	-2.2553	-11.7187	21	243	3	-16	130	318	5	14	21
94	-8.3445	1.1719	13	169	-2.2553	11.7187	21	244	3	-14	81	319	5	16	21
95	-8.2591	-23.8419	13	170	-2.2024	-26.7029	13	245	3	-12	69	320	5.1871	-6.6406	13
96	-8.2591	23.8419	13	171	-2.2024	26.7029	13	246	3	-10	60	321	5.1871	6.6406	13
97	-8.119	-1.5625	13	172	-2.1651	-6.25	13	247	3	-8	52	322	5.2858	-15.2588	13
98	-8.119	1.5625	13	173	-2.1651	6.25	13	248	3	-6	47	323	5.2858	15.2588	13
99	-8.1049	10.376	13	174	-2.0207	-5	13	249	3	-4	42	324	5.3405	-0.75	13
100	-7.9386	-3.75	13	175	-2.0207	5	13	250	3	-2	39	325	5.3405	0.75	13
101	-7.9386	3.75	13	176	-1.8042	-9.375	21	251	3	0	36	326	5.4127	-6.25	13
102	-7.7085	-19.0735	13	177	-1.8042	9.375	21	252	3	2	33	327	5.4127	6.25	13
103	-7.7085	19.0735	13	178	-1.7619	-21.3623	13	253	3	4	31	328	5.506	-28.6102	21
104	-7.2169	-6.25	21	179	-1.7619	21.3623	13	254	3	6	29	329	5.506	28.6102	21
105	-7.2169	-2.5	21	180	-1.4434	-7.5	21	255	3	8	27	330	5.6382	-9.7656	42
106	-7.2169	0	42	181	-1.4434	7.5	21	256	3	10	25	331	5.6382	9.7656	42
107	-7.2169	2.5	21	182	-1.4095	-17.0898	13	257	3	12	24	332	5.7735	-5	21
108	-7.2169	6.25	21	183	-1.4095	17.0898	13	258	3	14	23	333	5.7735	5	21
109	-7.0477	-12.207	42	184	-1.1276	-13.6719	13	259	3	16	21	334	6	-15	69
110	-6.7658	-7.8125	13	185	-1.1276	13.6719	13	260	3	18	21	335	6	-13	60
111	-6.7658	7.8125	13	186	-0.9021	-10.9375	13	261	3.1574	-7.8125	13	336	6	-11	52
112	-6.6756	0.9375	13	187	-0.9021	10.9375	13	262	3.1574	7.8125	13	337	6	-9	47
113	-6.6072	-19.0735	13	188	-0.7217	-8.75	13	263	3.3198	-4.25	13	338	6	-7	42
114	-6.6072	19.0735	13	189	-0.7217	8.75	13	264	3.3198	4.25	13	339	6	-5	39
115	-6.4952	1.25	13	190	0.7217	-8.75	13	265	3.3829	-9.7656	13	340	6	-3	36
116	-6.4839	-8.3008	13	191	0.7217	8.75	13	266	3.3829	9.7656	13	341	6	-1	33
117	-6.4839	8.3008	13	192	0.9021	-10.9375	13	267	3.5239	-18.3105	21	342	6	1	31
118	-6.1668	15.2588	13	193	0.9021	10.9375	13	268	3.5239	18.3105	21	343	6	3	29
119	-5.7735	-5	21	194	1.1276	-13.6719	13	269	3.6084	-6.25	42	344	6	5	27
120	-5.7735	0	42	195	1.1276	13.6719	13	270	3.6084	6.25	42	345	6	7	25
121	-5.7735	5	21	196	1.4095	-17.0898	13	271	3.9467	-9.7656	13	346	6	9	24
122	-5.6382	-9.7656	42	197	1.4095	17.0898	13	272	3.9467	9.7656	13	347	6	11	23
123	-5.6382	9.7656	42	198	1.4434	-7.5	21	273	4	-17	130	348	6	13	21
124	-5.506	-28.6102	21	199	1.4434	7.5	21	274	4	-15	81	349	6	15	21
125	-5.506	28.6102	21	200	1.7619	-21.3623	13	275	4	-13	69	350	6.1668	-15.2588	13
126	-5.4127	-6.25	13	201	1.7619	21.3623	13	276	4	-11	60	351	6.1668	15.2588	13
127	-5.4127	6.25	13	202	1.8042	-9.375	21	277	4	-9	52	352	6.4839	-8.3008	13
128	-5.3405	-0.75	13	203	1.8042	9.375	21	278	4	-7	47	353	6.4839	8.3008	13
129	-5.3405	0.75	13	204	2	-19	180	279	4	-5	42	354	6.4952	-1.25	13
130	-5.2858	-15.2588	13	205	2	-17	128	280	4	-3	39	355	6.4952	1.25	13
131	-5.2858	15.2588	13	206	2	-15	130	281	4	-1	36	356	6.6072	19.0735	13
132	-5.1871	-6.6406	13	207	2	-13	81	282	4	1	33	357	6.6073	-19.0735	13
133	-5.1871	6.6406	13	208	2	-11	69	283	4	3	31	358	6.6756	-0.9375	13
134	-4.9334	-12.207	13	209	2	-9	60	284	4	5	29	359	6.6756	0.9375	13
135	-4.9334	12.207	13	210	2	-7	52	285	4	7	27	360	6.7658	-7.8125	13
136	-4.5105	-7.8125	42	211	2	-5	47	286	4	9	25	361	6.7658	7.8125	13
137	-4.5105	7.8125	42	212	2	-3	42	287	4	11	24	362	7	-14	60
138	-4.4048	-22.8882	21	213	2	-1	39	288	4	13	23	363	7	-12	52

Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem	Kota	X	Y	Dem
364	7	-10	47	439	9.9232	4.6875	13	514	14	-5	29	589	24.2266	11.4441	13
365	7	-8	42	440	10	-11	42	515	14	-3	27	590	24.7772	-4.7684	13
366	7	-6	39	441	10	-9	39	516	14	-1	25	591	24.7772	4.7684	13
367	7	-4	36	442	10	-7	36	517	14	1	24	592	26.429	-15.2588	42
368	7	-2	33	443	10	-5	33	518	14	3	23	593	26.429	15.2588	42
369	7	0	31	444	10	-3	31	519	14	5	21	594	27.5302	-19.0735	13
370	7	2	29	445	10	-1	29	520	14	7	21	595	27.5302	-9.5367	21
371	7	4	27	446	10	1	27	521	14.0955	-12.207	21	596	27.5302	9.5367	21
372	7	6	25	447	10	3	25	522	14.0955	-9.7656	13	597	27.5302	19.0735	13
373	7	8	24	448	10	5	24	523	14.0955	-4.8828	21	598	30.2832	-14.3051	13
374	7	10	23	449	10	7	23	524	14.0955	0	42	599	30.2832	14.3051	13
375	7	12	21	450	10	9	21	525	14.0955	4.8828	21	600	33.0362	19.0735	42
376	7	14	21	451	10	11	21	526	14.0955	9.7656	13	601	33.0363	-19.0735	42
377	7.0477	-12.207	42	452	10.1311	-12.97	13	527	14.0955	12.207	21				
378	7.0477	12.207	42	453	10.1311	12.97	13	528	15	-6	29				
379	7.2169	-6.25	21	454	10.1487	-1.9531	13	529	15	-4	27				
380	7.2169	-5	13	455	10.1487	1.9531	13	530	15	-2	25				
381	7.2169	-2.5	21	456	10.4306	-1.4648	13	531	15	0	24				
382	7.2169	0	42	457	10.4306	1.4648	13	532	15	2	23				
383	7.2169	2.5	21	458	10.5716	-12.207	13	533	15	4	21				
384	7.2169	5	13	459	10.5716	12.207	13	534	15	6	21				
385	7.2169	6.25	21	460	10.8253	-6.25	42	535	15.505	-7.3242	13				
386	7.7085	-19.0735	13	461	10.8253	6.25	42	536	15.505	7.3242	13				
387	7.7085	19.0735	13	462	11	-10	39	537	15.8574	-3.0518	13				
388	7.9386	-3.75	13	463	11	-8	36	538	15.8574	3.0518	13				
389	7.9386	3.75	13	464	11	-6	33	539	16	-5	27				
390	8	-13	52	465	11	-4	31	540	16	-3	25				
391	8	-11	47	466	11	-2	29	541	16	-1	24				
392	8	-9	42	467	11	0	27	542	16	1	23				
393	8	-7	39	468	11	2	25	543	16	3	21				
394	8	-5	36	469	11	4	24	544	16	5	21				
395	8	-3	33	470	11	6	23	545	16.2979	-2.2888	13				
396	8	-1	31	471	11	8	21	546	16.2979	2.2888	13				
397	8	1	29	472	11	10	21	547	16.5181	-19.0735	13				
398	8	3	27	473	11.0121	-19.0735	42	548	16.5181	19.0735	13				
399	8	5	25	474	11.0121	19.0735	42	549	16.9146	-9.7656	42				
400	8	7	24	475	11.2764	-9.7656	21	550	16.9146	9.7656	42				
401	8	9	23	476	11.2764	-7.8125	13	551	17	-4	25				
402	8	11	21	477	11.2764	-3.9062	21	552	17	-2	24				
403	8	13	21	478	11.2764	0	42	553	17	0	23				
404	8.1049	-10.376	13	479	11.2764	3.9062	21	554	17	2	21				
405	8.1049	10.376	13	480	11.2764	7.8125	13	555	17	4	21				
406	8.119	-1.5625	13	481	11.2764	9.7656	21	556	17.6193	-15.2588	21				
407	8.119	1.5625	13	482	12	-9	36	557	17.6193	-12.207	13				
408	8.2591	-23.8418	13	483	12	-7	33	558	17.6193	-6.1035	21				
409	8.2591	23.8419	13	484	12	-5	31	559	17.6193	0	42				
410	8.3445	-1.1719	13	485	12	-3	29	560	17.6193	6.1035	21				
411	8.3445	1.1719	13	486	12	-1	27	561	17.6193	12.207	13				
412	8.4573	-9.7656	13	487	12	1	25	562	17.6193	15.2588	21				
413	8.4573	9.7656	13	488	12	3	24	563	18	-3	24				
414	8.6603	-5	42	489	12	5	23	564	18	-1	23				
415	8.6603	5	42	490	12	7	21	565	18	1	21				
416	8.8097	-15.2588	42	491	12	9	21	566	18	3	21				
417	8.8097	15.2588	42	492	12.404	-5.8594	13	567	19	-2	23				
418	9	-12	47	493	12.404	5.8594	13	568	19	0	21				
419	9	-10	42	494	12.6639	-16.2125	13	569	19	2	21				
420	9	-8	39	495	12.6639	16.2125	13	570	19.3813	-9.1553	13				
421	9	-6	36	496	12.6859	-2.4414	13	571	19.3813	9.1553	13				
422	9	-4	33	497	12.6859	2.4414	13	572	19.8217	-3.8147	13				
423	9	-2	31	498	13	-8	33	573	19.8217	3.8147	13				
424	9	0	29	499	13	-6	31	574	20	-1	21				
425	9	2	27	500	13	-4	29	575	20	1	21				
426	9	4	25	501	13	-2	27	576	20.3723	-2.861	13				
427	9	6	24	502	13	0	25	577	20.3723	2.861	13				
428	9	8	23	503	13	2	24	578	21	0	21				
429	9	10	21	504	13	4	23	579	21.1432	-12.207	42				
430	9	12	21	505	13	6	21	580	21.1432	12.207	42				
431	9.0211	-7.8125	21	506	13	8	21	581	22.0242	-19.0735	21				
432	9.0211	-6.25	13	507	13.0383	-1.8311	13	582	22.0242	-15.2588	13				
433	9.0211	-3.125	21	508	13.0383	1.8311	13	583	22.0242	-7.6294	21				
434	9.0211	0	42	509	13.2145	-15.2588	13	584	22.0242	0	42				
435	9.0211	3.125	21	510	13.2145	15.2588	13	585	22.0242	7.6294	21				
436	9.0211	6.25	13	511	13.5316	-7.8125	42	586	22.0242	15.2588	13				
437	9.0211	7.8125	21	512	13.5316	7.8125	42	587	22.0242	19.0735	21				
438	9.9232	-4.6875	13	513	14	-7	31	588	24.2266	-11.4441	13				

RUNNING PROGRAM

RUNNING PROGRAM

Bagian ini akan menampilkan hasil print screen *running* program.

***Study case* In**

Hasil *running* metode *Random*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>random<in
0.000000 18.384777 8.602325 15.297058 19.849434
18.384777 0.000000 10.000000 10.198039 2.000000
8.602325 10.000000 0.000000 10.770329 11.661903
15.297058 10.198039 10.770329 0.000000 10.000000
19.849434 2.000000 11.661903 10.000000 0.000000
0 2 3 1 4
0 1 4 2 3
0 2 1 3 4
0 3 4 1 2
0 4 1 3 2
0 3
0 3 1
0 3 1
0 4
0 4 2
0 4 2
Hasil Random: 0 3 1 0 4 2 0
83.993538
```

Hasil *running* metode *Nearest Neighbor*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>nn<in
0.000000 18.384777 8.602325 15.297058 19.849434
18.384777 0.000000 10.000000 10.198039 2.000000
8.602325 10.000000 0.000000 10.770329 11.661903
15.297058 10.198039 10.770329 0.000000 10.000000
19.849434 2.000000 11.661903 10.000000 0.000000
0 2 3 1 4
0 1 4 2 3
0 2 1 3 4
0 3 4 1 2
0 4 1 3 2
0 2
0 2 1
0 2 1
0 3
0 3 4
0 3 4
Hasil NN: 0 2 1 0 3 4 0
82.133591
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in
Hasil FTNs: 0 4 1 0 3 2 0
74.903923
Hasil FTNx: 0 4 1 0 3 2 0
74.903923
Hasil FTNy: 0 4 1 0 3 2 0
74.903923
```

***Study case* In2**

Hasil *running* metode *Random*:

```
Hasil Random: 0 2 16 13 14 1 0 5 6 9 15 0 8 3 0 10 12 0 7 4 0 11 0
338.456787
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 10 2 0 16 15 14 3 1 0 11 12 13 0 9 6 5 0 7 4 0 8 0
320.090790
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in2
Hasil FTNs: 0 9 5 6 13 0 1 8 4 0 15 7 14 16 0 3 2 0 12 11 0 10 0
299.902100
Hasil FTNx: 0 9 5 6 13 0 1 8 4 0 7 0 15 3 14 16 0 2 10 0 12 11 0
277.612396
Hasil FTNy: 0 16 4 5 1 15 0 14 8 3 0 6 9 0 7 10 0 11 13 12 0 2 0
318.755463
```

***Study case* In3**

Hasil *running* metode *Random*:

```
Hasil Random: 0 25 8 21 9 11 0 6 19 7 14 0 24 15 2 4 0 17 1 20 0 12 13 23 0 5 16
 22 0 10 0 3 0 18 0
935.044006
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 12 17 7 16 0 3 2 0 23 1 14 6 0 22 18 24 0 8 9 11 13 0 10 4 25 0 15 1
9 0 5 20 0 21 0
719.318970
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in3
Hasil FTNs: 0 2 13 21 20 14 0 22 25 4 18 0 23 24 15 0 6 5 7 0 11 19 1 0 9 17 16
12 0 10 8 0 3 0
745.469238
Hasil FTNx: 0 13 21 20 14 0 2 25 4 10 0 23 24 15 0 1 18 22 0 6 5 7 0 19 17 0 16
12 0 9 11 8 0 3 0
708.415405
Hasil FTNy: 0 13 21 20 14 0 2 25 4 22 23 0 24 15 0 18 0 6 5 7 0 19 0 12 16 11 17
 0 1 9 8 0 10 0 3 0
758.457886
```

***Study case* In4**

Hasil *running* metode *Random*:

```
Hasil Random: 0 18 3 35 22 16 39 29 26 21 20 8 6 47 33 23 0 5 7 40 46 30 13 43 1
4 11 44 41 32 24 38 28 0 10 37 36 27 42 15 9 31 2 17 25 4 34 12 19 0 45 1 0
147000.671875
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 14 23 11 12 15 32 45 43 18 7 27 35 29 6 36 0 13 21 46 20 39 9 1 8 37
 30 19 26 42 17 3 0 38 31 47 5 28 2 41 10 24 44 34 25 4 33 40 0 22 16 0
52006.218750
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in4
Hasil FTNs: 0 45 35 29 42 17 26 19 36 6 27 7 18 43 30 37 0 31 38 47 24 10 44 34
25 4 41 5 28 2 33 14 0 13 21 46 20 32 12 15 39 9 8 1 22 16 3 23 0 40 11 0
45841.710938
Hasil FTNx: 0 45 35 29 42 17 26 19 36 6 27 7 18 43 30 32 0 13 20 15 12 11 46 21
31 44 24 38 0 1 16 22 3 33 2 28 5 41 4 25 34 10 47 14 0 40 39 37 8 9 23 0
55599.843750
Hasil FTNy: 0 45 35 29 42 17 26 19 36 6 27 7 18 43 30 37 0 31 47 24 10 44 34 25
4 41 5 28 2 38 0 13 21 46 20 32 12 15 39 9 8 1 22 16 3 23 0 14 40 33 11 0
46155.199219
```

***Study case* In5**

Hasil *running* metode *Random*:

```
Hasil Random: 0 33 3 40 14 46 27 47 0 36 16 4 9 42 0 22 35 2 49 31 12 20 0 41 24
 48 13 28 0 6 45 29 8 34 0 1 37 7 26 30 19 15 0 11 18 43 39 32 21 44 0 5 25 17 2
3 10 0 38 0
1568417.750000
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 25 21 46 8 33 0 19 44 1 26 18 43 47 0 22 28 3 6 31 17 7 0 42 32 35 1
0 9 23 15 0 34 12 37 48 40 0 13 38 2 27 20 39 49 0 45 14 24 30 29 0 5 4 36 41 16
 0 11 0
1164397.875000
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in5
Hasil FTNs: 0 29 4 41 36 5 0 49 14 30 16 11 24 39 0 23 38 2 45 20 27 15 0 12 48
13 40 37 0 31 7 35 34 9 10 32 0 47 3 6 42 17 28 43 0 19 26 22 18 1 0 21 33 44 8
46 0 25 0
1038446.875000
Hasil FTNx: 0 29 4 41 36 5 0 49 14 30 16 11 24 39 0 23 38 2 45 20 27 15 0 12 48
13 40 37 0 31 7 35 34 9 10 32 0 47 3 6 42 17 28 43 0 19 26 22 18 1 0 21 33 44 8
46 0 25 0
1038446.875000
Hasil FTNy: 0 39 4 41 36 5 49 35 0 15 24 29 16 11 30 27 0 19 12 13 14 20 23 31 0
 48 38 45 2 40 0 47 42 10 37 9 7 3 0 22 17 34 32 28 0 8 1 6 43 44 0 25 33 18 26
21 0 46 0
1129691.000000
```

***Study case* In6**

Hasil *running* metode *Random*:

```
Hasil Random: 0 33 28 26 8 0 15 77 58 0 57 74 14 0 82 3 56 47 61 46 0 64 86 81 6
9 0 70 75 0 37 88 50 0 76 60 36 22 1 0 87 52 34 7 0 80 24 71 0 35 21 48 51 0 23
39 38 0 62 44 12 67 0 68 27 43 72 0 53 40 66 41 0 2 45 78 0 30 32 63 0 55 84 11
0 6 79 65 0 59 9 31 16 0 29 83 73 0 49 5 0 20 17 18 0 19 25 0 10 4 0 42 54 13 0
85 0
4321.045898
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 63 62 0 66 61 60 58 0 67 59 57 56 51 0 68 55 52 0 69 54 47 0 70 5
3 46 86 0 71 64 50 0 72 49 48 81 0 73 45 87 0 74 85 26 0 75 88 82 0 76 84 83 27
0 77 80 78 0 79 25 31 0 22 28 29 36 0 23 30 32 0 21 33 34 37 0 24 35 41 0 20 38
42 0 19 39 1 6 0 18 40 2 0 17 43 44 11 0 16 3 4 7 0 15 5 0 14 8 12 0 13 9 0 10 0

2731.557861
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in6
Hasil FTNs: 0 56 47 45 46 51 0 52 48 49 61 0 57 50 53 0 66 54 55 0 42 44 43 41 0
 3 1 2 4 0 62 58 59 71 0 67 60 63 0 36 40 39 0 16 7 5 6 11 0 35 38 37 0 12 8 9 2
1 0 17 10 13 0 32 34 33 31 0 76 64 65 0 26 14 15 0 86 30 29 0 25 28 27 0 22 18 1
9 81 0 87 20 23 0 83 24 88 0 72 68 69 0 77 70 73 0 74 75 0 82 78 79 0 80 84 0 85
 0
2080.421387
Hasil FTNx: 0 56 47 45 46 51 0 52 48 49 61 0 57 50 53 0 66 54 55 0 42 44 43 41 0
 3 1 2 4 0 62 58 59 71 0 67 60 63 0 36 40 39 0 16 7 5 6 11 0 35 38 37 0 12 8 9 2
1 0 17 10 13 0 32 34 33 31 0 76 64 65 0 26 14 15 0 30 29 0 25 28 27 0 22 18 19 0
 20 23 0 24 0 72 68 69 81 0 77 70 73 0 86 74 75 0 82 78 79 0 87 80 83 0 84 85 0
88 0
2029.837036
Hasil FTNy: 0 56 47 45 46 51 0 52 48 49 61 0 57 50 53 0 66 54 55 0 42 44 43 41 0
 3 1 2 4 0 62 58 59 71 0 67 60 63 0 36 40 39 0 16 7 5 6 11 0 35 38 37 0 12 8 9 2
1 0 17 10 13 0 32 34 33 31 0 76 64 65 0 26 14 15 0 30 29 0 25 28 27 0 22 18 19 0
 20 23 0 24 0 72 68 69 81 0 77 70 73 0 86 74 75 0 82 78 79 0 87 80 83 0 84 85 0
88 0
2029.837036
```

***Study case* In7**

Hasil *running* metode *Random*:

```
Hasil Random: 0 80 121 8 131 124 140 44 78 117 67 37 5 102 115 9 111 114 110 31
106 79 14 0 71 1 92 43 75 59 126 49 48 47 29 57 13 99 63 104 34 122 88 4 56 86 1
6 100 112 123 12 39 0 107 127 96 105 19 2 135 58 116 95 18 41 28 20 52 137 38 24
 7 70 81 54 21 109 17 42 82 65 0 11 23 72 25 26 27 22 61 89 87 40 84 30 83 101 9
0 129 66 46 113 62 15 0 53 33 85 6 128 51 120 45 103 68 134 133 138 136 36 130 3
 55 60 64 35 108 0 97 93 98 119 77 32 125 69 94 50 76 132 139 10 91 0 118 73 74
0
20262.847656
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 127 126 125 124 123 122 121 120 119 118 117 116 115 114 113 112 1
11 110 109 108 107 106 105 104 103 102 101 0 130 100 99 98 97 96 95 94 93 92 91
90 89 88 87 86 85 84 83 82 81 80 79 78 12 131 13 129 0 1 132 77 2 133 3 134 4 13
5 5 136 6 137 7 128 14 0 64 140 139 138 8 9 10 11 15 16 17 18 19 20 21 22 23 24
25 26 27 34 0 66 28 29 30 31 32 33 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49
0 67 50 51 52 53 54 55 56 57 58 59 60 61 62 63 68 69 70 71 72 73 0 74 75 76 0
10289.800781
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in7
Hasil FTNs: 0 105 107 109 120 122 124 126 128 130 132 134 136 138 140 139 137 13
5 133 131 129 127 125 123 121 119 108 106 104 0 84 86 96 98 100 102 110 112 114
116 118 117 115 113 111 103 101 99 97 87 85 83 0 63 65 74 76 78 80 82 89 91 93 9
5 94 92 90 88 81 79 77 75 73 64 62 0 42 50 52 54 56 58 60 66 68 70 72 71 69 67 6
1 59 57 55 53 51 43 41 0 21 28 30 32 34 36 38 40 45 47 49 48 46 44 39 37 35 33 3
1 29 27 20 0 4 6 8 10 12 14 16 18 22 24 26 25 23 19 17 15 13 11 9 7 5 0 1 3 2 0

2845.391602
Hasil FTNx: 0 105 107 109 120 122 124 126 128 130 132 134 136 138 140 139 137 13
5 133 131 129 127 125 123 121 119 108 106 104 0 84 86 96 98 100 102 110 112 114
116 118 117 115 113 111 103 101 99 97 87 85 83 0 63 65 74 76 78 80 82 89 91 93 9
5 94 92 90 88 81 79 77 75 73 64 62 0 42 50 52 54 56 58 60 66 68 70 72 71 69 67 6
1 59 57 55 53 51 43 41 0 21 28 30 32 34 36 38 40 45 47 49 48 46 44 39 37 35 33 3
1 29 27 20 0 4 6 8 10 12 14 16 18 22 24 26 25 23 19 17 15 13 11 9 7 5 0 1 3 2 0

2845.391602
Hasil FTNy: 0 105 107 109 120 122 124 126 128 130 132 134 136 138 140 139 137 13
5 133 131 129 127 125 123 121 119 108 106 104 0 84 86 96 98 100 102 110 112 114
116 118 117 115 113 111 103 101 99 97 87 85 83 0 63 65 74 76 78 80 82 89 91 93 9
5 94 92 90 88 81 79 77 75 73 64 62 0 42 50 52 54 56 58 60 66 68 70 72 71 69 67 6
1 59 57 55 53 51 43 41 0 21 28 30 32 34 36 38 40 45 47 49 48 46 44 39 37 35 33 3
1 29 27 20 0 4 6 8 10 12 14 16 18 22 24 26 25 23 19 17 15 13 11 9 7 5 0 1 3 2 0

2845.391602
```

***Study case* In8**

Hasil *running* metode *Random*:

```
Hasil Random: 0 148 146 15 80 153 147 160 167 176 22 91 82 130 64 182 99 129 50
0 92 112 191 27 68 170 121 186 65 49 6 144 172 0 55 29 163 119 8 74 89 76 120 11
1 195 164 151 103 188 0 11 171 197 35 41 18 145 2 136 95 63 87 138 177 45 105 15
2 107 0 28 23 86 16 104 135 56 123 139 118 198 77 175 25 13 83 0 59 70 159 90 10
9 75 30 85 125 137 116 0 37 166 43 44 3 33 21 110 67 52 93 69 98 0 143 20 61 156
 101 36 199 1 114 178 131 122 32 0 40 149 181 73 201 46 12 158 150 185 180 174 1
02 0 141 106 24 72 200 194 97 53 140 133 62 96 9 14 47 127 4 54 0 183 154 117 19
3 115 79 128 51 196 58 19 126 187 0 173 10 142 157 7 189 161 124 100 60 134 66 3
8 165 113 0 184 132 5 190 155 31 71 168 48 26 78 108 88 84 57 0 169 179 192 94 8
1 42 39 162 34 17 0
4419.457031
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 182 185 174 190 187 167 188 186 183 189 171 181 179 178 0 130 162
 180 161 176 164 177 175 172 170 153 149 166 169 168 152 0 195 147 157 165 144 1
63 159 160 146 184 158 155 156 138 151 143 0 67 154 150 140 173 200 201 199 198
191 142 148 145 197 1 139 2 29 0 193 196 141 194 3 4 192 5 6 7 8 9 10 15 0 128 1
1 12 13 14 16 17 18 19 20 21 40 0 132 22 23 24 25 26 27 28 30 31 32 33 34 35 36
47 50 53 0 135 37 38 39 41 42 43 44 45 46 51 54 0 133 48 49 52 55 56 57 58 59 60
 61 62 63 64 66 69 86 107 0 137 68 70 71 72 73 74 75 76 77 78 79 127 0 136 80 81
 82 83 84 85 87 88 89 90 91 92 95 0 131 93 94 96 97 98 99 100 101 102 103 104 11
3 129 0 134 105 106 108 109 110 111 112 114 115 116 123 0 120 117 118 119 121 12
2 124 125 126 0
3711.699219
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in8
Hasil FTNs: 0 170 175 180 184 189 193 197 201 199 195 191 187 182 178 172 158 0
153 165 173 176 183 185 196 200 198 192 186 177 181 168 166 0 80 57 74 125 87 10
9 81 140 141 137 134 132 130 126 0 151 169 179 190 194 188 174 171 155 152 149 1
47 143 160 0 53 43 34 36 33 29 24 18 10 6 46 21 25 19 16 8 4 0 107 112 116 121 1
28 162 167 164 145 123 118 114 110 88 0 96 98 100 103 105 139 131 129 150 161 15
9 146 135 119 0 65 111 115 120 122 136 138 154 163 156 157 133 117 104 0 56 73 8
6 148 124 99 63 45 78 82 108 113 89 93 97 127 50 0 64 75 77 79 101 106 91 95 102
 144 142 84 76 0 55 60 66 70 83 90 94 92 85 72 68 62 58 59 0 40 42 48 54 71 67 6
1 52 51 49 47 44 38 41 0 22 26 28 30 32 37 35 31 12 39 27 23 14 0 5 9 13 17 69 2
0 15 11 7 3 1 2 0
1498.265137
Hasil FTNx: 0 170 175 180 184 189 193 197 201 199 195 191 187 182 178 172 158 0
153 165 173 176 183 185 196 200 198 192 186 177 181 168 166 0 80 57 74 125 87 10
9 81 140 141 137 134 132 130 126 0 151 169 179 190 194 188 174 171 155 152 149 1
47 143 160 0 53 43 34 36 33 29 24 18 10 6 46 21 25 19 16 8 4 0 107 112 116 121 1
28 162 167 164 145 123 118 114 110 88 0 96 98 100 103 105 139 131 129 150 161 15
9 146 135 119 0 65 111 115 120 122 136 138 154 163 156 157 133 117 104 0 56 73 8
6 148 124 99 63 45 78 82 108 113 89 93 97 127 50 0 64 75 77 79 101 106 91 95 102
 144 142 84 76 0 55 60 66 70 83 90 94 92 85 72 68 62 58 59 0 40 42 48 54 71 67 6
1 52 51 49 47 44 38 41 0 22 26 28 30 32 37 35 31 12 39 27 23 14 0 5 9 13 17 69 2
0 15 11 7 3 1 2 0
1498.265137
Hasil FTNy: 0 170 175 180 184 189 193 197 201 199 195 191 187 182 178 172 158 0
153 165 173 176 183 185 196 200 198 192 186 177 181 168 166 0 80 57 74 125 87 10
9 81 140 141 137 134 132 130 126 0 151 169 179 190 194 188 174 171 155 152 149 1
47 143 160 0 53 43 34 36 33 29 24 18 10 6 46 21 25 19 16 8 4 0 107 112 116 121 1
28 162 167 164 145 123 118 114 110 88 0 96 98 100 103 105 139 131 129 150 161 15
9 146 135 119 0 65 111 115 120 122 136 138 154 163 156 157 133 117 104 0 9 56 73
 86 148 124 99 63 45 78 82 108 11 0 64 79 101 106 102 95 91 84 97 93 127 144 142
 113 89 69 0 50 60 66 70 75 83 90 94 92 85 77 72 68 62 54 0 44 47 49 58 55 59 51
 52 67 76 71 61 48 0 22 28 30 32 35 40 42 39 38 41 37 31 23 26 0 2 15 20 12 27 1
4 17 13 0 5 1 3 7 0
1503.267090
```

***Study case* In9**

Hasil *running* metode *Random*:

```
Hasil Random: 0 61 180 56 149 93 185 107 99 240 54 43 90 162 69 3 207 0 76 33 13
5 63 74 231 230 64 117 153 118 31 55 100 157 0 201 23 139 208 132 181 233 192 70
 1 111 200 227 148 78 0 108 152 177 77 12 94 218 190 161 30 62 28 73 51 226 75 0
 113 232 57 221 213 239 234 65 134 199 124 68 11 112 0 171 109 67 211 27 82 123
174 45 119 166 154 121 84 0 183 220 182 92 25 237 189 195 96 6 26 88 142 95 173
104 72 0 38 81 89 194 178 191 238 106 163 102 219 244 176 103 0 29 204 46 86 214
 137 205 245 187 167 2 202 105 212 37 138 0 222 241 48 22 155 32 147 210 224 130
 223 164 144 151 14 0 8 18 143 128 80 159 242 58 83 175 53 170 188 114 0 160 203
 91 24 248 186 34 47 126 4 5 158 168 16 116 21 140 0 44 156 36 9 59 101 42 98 19
8 49 209 225 0 236 40 71 19 146 15 216 150 85 206 115 193 0 172 10 35 13 127 247
 217 145 87 229 110 0 131 228 246 243 52 120 179 60 196 39 133 66 125 0 20 7 136
 184 122 215 17 197 165 41 235 97 141 0 169 79 129 50 0
4759.541992
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 243 193 186 188 248 239 213 212 194 227 195 207 0 130 235 197 189
 223 215 192 191 190 224 211 240 201 185 226 0 4 199 246 198 187 247 196 245 208
 200 241 210 209 202 242 214 204 1 219 203 0 15 237 222 221 206 216 217 238 236
205 2 230 220 218 231 225 0 159 244 232 234 228 233 229 3 5 6 7 8 0 96 9 10 11 1
2 13 14 16 17 18 19 0 27 20 21 22 23 24 25 26 28 29 30 31 32 33 34 49 0 42 35 36
 37 38 39 40 41 43 50 0 52 44 45 46 47 48 51 53 54 55 56 57 58 59 60 61 0 126 62
 63 64 66 67 68 69 70 71 77 0 181 72 73 74 75 76 78 79 80 81 82 83 84 85 86 0 11
8 87 88 89 90 91 92 93 94 95 97 98 99 100 0 135 101 102 103 104 105 106 107 108
109 110 111 113 0 163 112 114 115 116 117 119 120 121 122 123 124 125 127 0 150
128 129 131 132 133 134 136 137 138 139 140 141 142 143 144 0 164 145 146 147 14
8 149 151 152 153 154 155 156 157 158 167 0 177 160 161 162 165 166 168 169 170
171 172 173 174 175 0 179 176 178 180 182 183 184 0
3802.241699
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in9
Hasil FTNs: 0 232 243 234 223 217 212 219 228 222 215 209 207 211 221 245 236 23
7 238 0 240 235 225 218 213 220 229 226 216 210 208 214 227 246 242 241 233 0 17
3 180 194 200 203 206 199 197 192 139 154 166 167 156 121 0 135 151 157 171 195
201 204 205 202 198 196 189 178 170 0 4 248 231 224 230 239 247 244 6 18 45 30 1
7 29 44 34 0 49 22 7 2 53 31 118 61 35 8 0 20 23 48 21 62 1 54 32 119 140 155 17
2 104 98 75 47 0 72 94 116 125 136 126 117 193 96 74 58 50 19 46 73 82 95 81 0 1
0 9 57 36 63 97 103 85 64 37 0 38 76 65 86 107 120 127 108 87 105 0 129 137 144
159 168 184 174 182 185 190 186 175 160 145 141 0 67 83 109 158 143 128 88 59 39
 25 11 0 124 134 138 150 165 181 179 164 149 133 142 148 153 162 169 176 0 112 1
32 163 183 191 188 177 187 161 152 146 130 122 110 89 0 41 55 51 68 78 33 66 77
5 24 12 40 26 13 3 0 42 69 90 123 147 131 111 101 99 79 56 52 70 100 0 27 91 102
 113 92 80 60 93 106 115 84 71 43 0 14 28 114 16 15 0
916.316650
Hasil FTNx: 0 232 243 234 223 217 212 219 228 222 215 209 207 211 221 245 236 23
7 238 0 240 235 225 218 213 220 229 226 216 210 208 214 227 246 242 241 233 0 17
3 180 194 200 203 206 199 197 192 139 154 166 167 156 121 0 135 151 157 171 195
201 204 205 202 198 196 189 178 170 0 248 231 224 230 239 247 244 0 49 22 7 2 53
 31 118 61 35 8 0 20 23 48 21 62 1 54 32 119 140 155 172 104 98 75 47 0 115 106
94 116 125 136 126 117 193 96 74 58 50 19 46 18 6 0 10 9 57 36 63 97 103 85 64 3
7 0 38 76 65 86 107 120 127 108 87 105 0 129 137 144 159 168 184 174 182 185 190
 186 175 160 145 141 0 67 83 109 158 143 128 88 59 39 25 11 0 124 134 138 150 16
5 181 179 164 149 133 142 148 153 162 169 176 0 112 132 163 183 191 188 177 187
161 152 146 130 122 110 89 0 41 55 51 68 78 33 66 77 5 24 12 40 26 13 3 0 43 34
44 60 72 81 82 95 73 45 30 17 29 16 28 0 42 69 90 123 147 131 111 101 99 79 56 5
2 70 100 0 4 14 27 113 91 102 80 92 114 84 93 71 15 0
908.817932
Hasil FTNy: 0 232 243 234 223 217 212 219 228 222 215 209 207 211 221 245 236 23
7 238 0 240 235 225 218 213 220 229 226 216 210 208 214 227 246 242 241 233 0 17
3 180 194 200 203 206 199 197 192 139 154 166 167 156 121 0 135 151 157 171 195
201 204 205 202 198 196 189 178 170 0 89 78 68 51 55 41 27 14 3 239 224 230 247
4 0 34 80 71 60 44 29 17 30 45 18 6 244 231 248 15 0 49 22 7 2 53 31 118 61 35 8
 0 20 23 48 21 62 1 54 32 119 140 155 172 104 98 75 47 0 72 94 116 125 136 126 1
17 193 96 74 58 50 19 46 73 82 95 81 0 10 9 57 36 63 97 103 85 64 37 0 38 76 65
86 107 120 127 108 87 105 0 129 137 144 159 168 184 174 182 185 190 186 175 160
145 141 0 67 83 109 158 143 128 88 59 39 25 11 0 124 134 138 150 165 181 179 164
 149 133 142 148 153 163 177 191 188 183 0 131 147 162 187 176 169 161 152 146 1
30 122 110 111 0 79 90 101 99 26 33 66 77 5 24 12 40 13 69 0 16 52 70 113 102 10
0 92 114 93 106 115 84 43 28 0 56 112 132 123 91 42 0
956.784607
```

***Study case* In10**

Hasil *running* metode *Random*:

```
Hasil Random: 0 168 288 109 160 243 236 217 167 261 118 35 201 1 110 2 27 212 12
3 56 286 0 170 54 125 140 205 210 57 113 188 177 163 92 175 292 66 277 52 271 24
6 16 0 103 90 46 259 75 85 42 49 151 48 273 39 274 231 149 18 257 249 115 102 0
112 237 91 89 252 88 295 187 155 166 226 165 122 101 8 180 114 296 15 70 0 11 26
 133 20 230 178 154 285 124 80 176 194 36 121 208 183 267 233 162 279 0 147 144
272 3 206 136 98 289 138 69 225 87 204 139 240 223 71 253 93 232 0 209 76 239 22
8 105 82 99 256 284 86 84 250 244 25 255 145 21 117 96 224 0 262 107 9 137 31 45
 100 173 33 254 47 134 55 290 73 6 179 215 164 266 0 200 77 221 120 264 108 218
38 83 207 30 234 126 172 67 41 10 141 189 227 0 242 247 184 278 37 53 50 148 13
116 152 72 297 97 150 94 293 182 131 238 0 260 245 7 159 222 17 22 203 294 62 19
5 79 28 61 142 51 104 68 106 280 0 59 40 60 185 192 29 24 282 111 127 193 216 28
7 281 235 161 5 196 146 119 0 129 81 191 213 186 153 74 275 43 270 229 65 258 25
1 174 158 14 283 44 219 0 78 198 291 202 157 63 132 32 64 128 265 241 276 268 4
269 34 199 135 58 0 156 190 181 263 19 197 95 169 130 171 248 220 12 211 23 214
143 0
49135.425781
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 255 254 253 252 251 250 249 248 247 246 245 244 243 242 241 240 2
39 238 237 0 130 236 235 234 297 296 295 294 293 292 291 290 289 288 287 286 285
 284 283 282 0 195 281 280 279 1 261 2 262 278 3 263 4 264 5 265 6 266 7 267 8 0
 260 9 269 10 270 11 271 12 272 13 273 14 274 15 275 16 276 17 277 18 0 64 19 26
8 20 259 21 258 22 257 23 256 24 25 26 27 28 29 30 31 32 0 66 33 34 35 36 37 38
39 40 41 42 43 44 45 46 47 48 49 50 51 0 129 52 53 54 55 56 57 58 59 60 61 62 63
 67 68 69 70 71 72 73 0 131 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 9
1 92 0 194 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111
0 196 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 132 13
3 0 193 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151
152 0 197 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 17
0 171 0 192 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188
189 190 0 198 191 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 21
4 215 216 0 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233
0
23871.892578
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in10
Hasil FTNs: 0 279 281 283 285 287 289 291 293 295 297 296 294 292 290 288 286 28
4 282 280 278 0 259 261 263 265 267 269 271 273 275 277 276 274 272 270 268 266
264 262 260 258 0 239 241 243 245 247 249 251 253 255 257 256 254 252 250 248 24
6 244 242 240 238 0 219 221 223 225 227 229 231 233 235 237 236 234 232 230 228
226 224 222 220 218 0 199 201 203 205 207 209 211 213 215 217 216 214 212 210 20
8 206 204 202 200 198 0 179 181 183 185 187 189 191 193 195 197 196 194 192 190
188 186 184 182 180 178 0 159 161 163 165 167 169 171 173 175 177 176 174 172 17
0 168 166 164 162 160 158 0 139 141 143 145 147 149 151 153 155 157 156 154 152
150 148 146 144 142 140 138 0 119 121 123 125 127 129 131 133 135 137 136 134 13
2 130 128 126 124 122 120 118 0 99 101 103 105 107 109 111 113 115 117 116 114 1
12 110 108 106 104 102 100 98 0 79 81 83 85 87 89 91 93 95 97 96 94 92 90 88 86
84 82 80 78 0 59 61 63 65 67 69 71 73 75 77 76 74 72 70 68 66 64 62 60 58 0 39 4
1 43 45 47 49 51 53 55 57 56 54 52 50 48 46 44 42 40 38 0 19 21 23 25 27 29 31 3
3 35 37 36 34 32 30 28 26 24 22 20 18 0 1 3 5 7 9 11 13 15 17 16 14 12 10 8 6 4
2 0
6660.969238
Hasil FTNx: 0 279 281 283 285 287 289 291 293 295 297 296 294 292 290 288 286 28
4 282 280 278 0 259 261 263 265 267 269 271 273 275 277 276 274 272 270 268 266
264 262 260 258 0 239 241 243 245 247 249 251 253 255 257 256 254 252 250 248 24
6 244 242 240 238 0 219 221 223 225 227 229 231 233 235 237 236 234 232 230 228
226 224 222 220 218 0 199 201 203 205 207 209 211 213 215 217 216 214 212 210 20
8 206 204 202 200 198 0 179 181 183 185 187 189 191 193 195 197 196 194 192 190
188 186 184 182 180 178 0 159 161 163 165 167 169 171 173 175 177 176 174 172 17
0 168 166 164 162 160 158 0 139 141 143 145 147 149 151 153 155 157 156 154 152
150 148 146 144 142 140 138 0 119 121 123 125 127 129 131 133 135 137 136 134 13
2 130 128 126 124 122 120 118 0 99 101 103 105 107 109 111 113 115 117 116 114 1
12 110 108 106 104 102 100 98 0 79 81 83 85 87 89 91 93 95 97 96 94 92 90 88 86
84 82 80 78 0 59 61 63 65 67 69 71 73 75 77 76 74 72 70 68 66 64 62 60 58 0 39 4
1 43 45 47 49 51 53 55 57 56 54 52 50 48 46 44 42 40 38 0 19 21 23 25 27 29 31 3
3 35 37 36 34 32 30 28 26 24 22 20 18 0 1 3 5 7 9 11 13 15 17 16 14 12 10 8 6 4
2 0
6660.969238
Hasil FTNy: 0 279 281 283 285 287 289 291 293 295 297 296 294 292 290 288 286 28
4 282 280 278 0 259 261 263 265 267 269 271 273 275 277 276 274 272 270 268 266
264 262 260 258 0 239 241 243 245 247 249 251 253 255 257 256 254 252 250 248 24
6 244 242 240 238 0 219 221 223 225 227 229 231 233 235 237 236 234 232 230 228
226 224 222 220 218 0 199 201 203 205 207 209 211 213 215 217 216 214 212 210 20
8 206 204 202 200 198 0 179 181 183 185 187 189 191 193 195 197 196 194 192 190
188 186 184 182 180 178 0 159 161 163 165 167 169 171 173 175 177 176 174 172 17
0 168 166 164 162 160 158 0 139 141 143 145 147 149 151 153 155 157 156 154 152
150 148 146 144 142 140 138 0 119 121 123 125 127 129 131 133 135 137 136 134 13
2 130 128 126 124 122 120 118 0 99 101 103 105 107 109 111 113 115 117 116 114 1
12 110 108 106 104 102 100 98 0 79 81 83 85 87 89 91 93 95 97 96 94 92 90 88 86
84 82 80 78 0 59 61 63 65 67 69 71 73 75 77 76 74 72 70 68 66 64 62 60 58 0 39 4
1 43 45 47 49 51 53 55 57 56 54 52 50 48 46 44 42 40 38 0 19 21 23 25 27 29 31 3
3 35 37 36 34 32 30 28 26 24 22 20 18 0 1 3 5 7 9 11 13 15 17 16 14 12 10 8 6 4
2 0
6660.969238
```

***Study case* In11**

Hasil *running* metode *Random*:

```
Hasil Random: 0 288 118 102 140 36 121 168 155 143 150 289 249 81 183 142 135 22
3 75 90 294 281 5 220 201 304 279 0 301 91 141 259 163 203 293 99 199 37 56 219
116 15 270 119 256 275 176 7 42 0 21 101 228 317 122 35 111 232 197 205 92 320 2
43 231 73 167 68 291 311 290 93 252 110 59 133 185 0 134 212 23 38 170 217 127 2
24 124 52 307 234 138 53 10 173 202 186 182 238 318 162 0 94 229 215 245 128 189
 237 267 129 313 248 84 319 303 277 146 51 61 261 48 193 148 250 216 144 95 0 22
 89 225 222 17 171 85 180 262 321 300 274 82 306 258 164 98 63 2 0 287 69 218 13
 254 72 16 322 263 284 152 104 253 28 120 316 278 77 19 158 200 251 0 305 40 96
78 43 41 235 295 169 310 192 242 260 47 157 33 165 175 4 214 227 0 283 29 50 149
 195 285 8 107 271 24 179 194 83 32 31 206 204 55 0 76 25 115 236 292 26 9 299 2
96 44 309 12 314 221 131 74 132 160 145 226 117 100 0 181 79 187 244 58 67 266 1
72 213 106 27 188 1 159 136 137 105 196 147 156 0 280 265 57 87 125 246 257 3 16
6 62 126 39 139 208 64 210 153 315 268 0 198 264 191 282 272 184 230 273 20 255
66 151 178 112 86 45 211 312 239 80 177 34 46 209 130 298 308 0 108 6 60 54 88 2
33 276 103 18 190 97 241 11 71 240 286 0 247 114 207 297 65 49 109 30 154 14 123
 161 302 113 174 70 269 0
4825.874512
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 313 311 310 300 317 315 314 312 301 318 316 304 319 320 308 291 3
02 309 292 303 307 299 305 298 306 290 283 297 294 293 284 289 276 295 285 282 0
 130 275 286 296 287 288 281 280 279 278 277 271 270 265 274 272 266 269 261 273
 267 268 264 263 260 262 259 322 321 5 44 0 195 1 2 3 4 0 64 6 7 8 9 10 11 27 0
67 12 13 14 15 16 17 18 19 20 77 0 68 21 22 23 24 25 26 28 29 30 31 76 0 133 32
33 34 35 36 37 38 39 40 41 42 43 45 46 189 0 134 47 48 49 50 51 52 53 54 55 56 5
7 58 59 60 61 104 0 193 62 63 66 69 70 71 72 73 74 75 78 79 80 81 82 83 84 85 0
198 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 105 106 107 108 15
2 0 199 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
127 128 129 131 132 0 257 135 136 137 138 139 140 141 142 143 144 145 146 147 14
8 149 150 151 153 154 155 156 157 158 159 160 161 209 0 192 162 163 164 165 166
167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186
187 188 190 191 194 0 196 197 200 201 202 203 204 205 206 207 208 210 211 212 21
3 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 251 0
256 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250
252 253 254 255 258 0
2446.507080
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in11
Hasil FTNs: 0 45 55 66 78 91 105 120 136 153 171 190 210 231 253 274 294 312 293
 273 252 230 209 189 170 152 135 119 104 90 77 65 54 44 137 0 75 62 74 87 101 11
6 132 149 167 186 206 227 249 271 292 311 291 270 248 226 205 185 166 148 131 11
5 100 86 73 61 0 34 43 53 76 89 103 118 134 151 169 188 208 229 251 272 250 228
207 187 168 150 133 117 102 88 64 63 52 42 269 0 123 56 67 79 92 106 121 154 172
 191 211 232 254 275 313 295 276 255 233 212 192 173 155 138 122 107 93 80 68 57
 0 97 84 98 113 129 146 164 183 203 224 246 268 289 309 310 290 247 225 204 184
165 147 130 114 99 85 72 0 82 95 109 124 140 157 175 194 214 235 257 278 297 315
 314 296 277 256 234 213 193 174 156 139 108 94 81 69 0 110 96 111 127 144 162 1
81 201 222 244 266 287 307 308 288 267 245 223 202 182 163 145 128 112 83 0 126
142 159 177 196 216 237 259 280 299 317 316 298 279 258 236 215 195 176 158 141
125 160 143 161 179 198 0 243 221 242 263 283 262 241 220 200 180 199 219 240 21
8 239 261 282 301 319 318 300 281 260 238 217 197 178 0 37 47 58 70 264 285 305
306 286 265 304 284 303 321 302 320 322 71 59 48 38 0 46 10 15 21 28 36 35 27 20
 14 9 5 0 12 18 25 33 51 41 32 24 17 11 0 23 31 40 50 60 49 39 30 22 16 0 4 13 2
6 19 8 7 6 29 0 1 2 3 0
850.803955
Hasil FTNx: 0 45 55 66 78 91 105 120 136 153 171 190 210 231 253 274 294 312 293
 273 252 230 209 189 170 152 135 119 104 90 77 65 54 44 137 0 75 62 74 87 101 11
6 132 149 167 186 206 227 249 271 292 311 291 270 248 226 205 185 166 148 131 11
5 100 86 73 61 0 34 43 53 76 89 103 118 134 151 169 188 208 229 251 272 250 228
207 187 168 150 133 117 102 88 64 63 52 42 269 0 123 56 67 79 92 106 121 154 172
 191 211 232 254 275 313 295 276 255 233 212 192 173 155 138 122 107 93 80 68 57
 0 97 84 98 113 129 146 164 183 203 224 246 268 289 309 310 290 247 225 204 184
165 147 130 114 99 85 72 0 82 95 109 124 140 157 175 194 214 235 257 278 297 315
 314 296 277 256 234 213 193 174 156 139 108 94 81 69 0 110 96 111 127 144 162 1
81 201 222 244 266 287 307 308 288 267 245 223 202 182 163 145 128 112 83 0 126
142 159 177 196 216 237 259 280 299 317 316 298 279 258 236 215 195 176 158 141
125 160 143 161 179 198 0 243 221 242 263 283 262 241 220 200 180 199 219 240 21
8 239 261 282 301 319 318 300 281 260 238 217 197 178 0 38 48 59 71 322 264 285
305 306 286 265 304 284 303 321 302 320 70 58 47 37 0 46 10 15 21 28 36 35 27 20
 14 9 5 0 12 18 25 33 51 41 32 24 17 11 0 23 31 40 50 60 49 39 30 22 16 0 4 13 2
6 19 8 7 0 2 29 6 3 0 1 0
851.023193
Hasil FTNy: 0 45 55 66 78 91 105 120 136 153 171 190 210 231 253 274 294 312 293
 273 252 230 209 189 170 152 135 119 104 90 77 65 54 44 137 0 75 62 74 87 101 11
6 132 149 167 186 206 227 249 271 292 311 291 270 248 226 205 185 166 148 131 11
5 100 86 73 61 0 34 43 53 76 89 103 118 134 151 169 188 208 229 251 272 250 228
207 187 168 150 133 117 102 88 64 63 52 42 269 0 123 56 67 79 92 106 121 154 172
 191 211 232 254 275 313 295 276 255 233 212 192 173 155 138 122 107 93 80 68 57
 0 97 84 98 113 129 146 164 183 203 224 246 268 289 309 310 290 247 225 204 184
165 147 130 114 99 85 72 0 82 95 109 124 140 157 175 194 214 235 257 278 297 315
 314 296 277 256 234 213 193 174 156 139 108 94 81 69 0 110 96 111 127 144 162 1
81 201 222 244 266 287 307 308 288 267 245 223 202 182 163 145 128 112 83 0 126
142 159 177 196 216 237 259 280 299 317 316 298 279 258 236 215 195 176 158 141
125 160 143 161 179 198 0 243 221 242 263 283 262 241 220 200 180 199 219 240 21
8 239 261 282 301 319 318 300 281 260 238 217 197 178 0 37 47 58 70 264 285 305
306 286 265 304 284 303 321 302 320 322 71 59 48 38 0 46 10 15 21 28 36 35 27 20
 14 9 5 0 12 18 25 33 51 41 32 24 17 11 0 23 31 40 50 60 49 39 30 22 16 0 4 13 2
6 19 8 7 6 29 0 1 2 3 0
850.803955
```

***<u>Study case</u>* <u>In12</u>**

Hasil *running* metode *Random*:

```
Hasil Random: 0 140 10 198 368 67 365 238 206 22 228 292 213 136 355 344 53 375
277 21 113 103 49 146 43 259 48 0 74 68 167 235 196 237 153 114 20 23 254 343 92
 255 363 275 329 229 360 77 315 6 242 13 241 115 44 0 145 372 142 340 12 250 72
280 265 152 102 117 286 374 221 300 233 132 122 90 157 281 134 339 347 313 0 239
 149 217 332 337 39 69 328 79 207 251 319 351 105 354 66 125 57 33 120 130 317 8
 41 327 81 356 15 0 230 128 86 119 263 293 248 17 166 104 147 89 172 116 283 87
370 256 282 369 246 224 84 278 187 0 50 204 244 176 192 290 291 61 97 214 367 20
1 205 158 297 37 133 264 56 100 40 220 144 274 29 27 0 154 185 216 11 267 42 118
 373 35 222 63 211 188 285 208 359 309 210 75 173 51 0 212 165 179 321 80 59 169
 203 108 312 135 7 126 311 371 258 99 18 139 353 160 318 302 199 331 0 324 288 4
5 111 64 314 177 175 174 247 83 287 62 215 121 268 24 195 295 362 330 296 225 12
3 131 109 107 0 304 155 52 326 55 106 240 357 323 226 276 227 85 336 94 219 231
298 325 30 161 334 150 168 19 342 0 129 163 180 28 95 14 4 65 305 93 253 349 171
 346 127 301 32 335 307 156 36 243 9 71 310 112 209 0 1 352 143 322 333 184 249
190 162 218 98 137 151 338 194 124 2 31 232 138 308 47 54 70 58 0 110 271 306 29
9 260 164 96 270 181 73 284 269 245 261 16 25 289 141 26 266 159 341 60 358 197
76 0 193 236 191 78 182 364 38 183 82 46 186 178 366 316 252 170 34 262 350 223
88 294 345 91 3 320 273 0 148 202 303 257 101 279 5 200 189 348 272 361 234 0
5917.855957
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 313 315 317 319 375 374 367 318 373 372 366 316 314 369 365 371 3
70 312 368 363 359 356 362 358 364 360 357 353 361 355 351 348 354 350 349 347 0
 130 344 352 346 342 339 345 341 338 343 340 337 335 332 336 333 330 327 334 331
 328 326 1 325 323 2 329 3 4 5 320 6 321 7 8 0 195 324 9 322 10 11 12 13 14 15 1
6 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 0 260
41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 66 67 68
 69 70 76 0 129 71 72 73 74 75 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 9
3 94 95 96 97 0 196 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 11
3 114 115 116 117 118 119 120 121 0 259 122 123 124 125 126 127 128 131 132 133
134 135 136 137 138 139 140 141 142 143 144 0 261 145 146 147 148 149 150 151 15
2 153 154 155 156 157 158 159 160 161 162 163 168 0 194 164 165 166 167 169 170
171 172 173 174 175 176 177 178 179 180 181 182 183 184 0 206 185 186 187 188 18
9 190 191 192 193 197 198 199 200 201 0 207 202 203 204 205 208 209 210 211 212
213 214 215 216 217 218 219 220 221 242 0 271 222 223 224 225 226 227 228 229 23
0 231 232 233 234 235 236 237 238 239 240 241 246 0 272 243 244 245 247 248 249
250 251 252 253 254 255 256 257 258 262 263 264 265 266 267 275 0 270 268 269 27
3 274 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 29
4 295 0 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 0
2847.682617
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in12
Hasil FTNs: 0 311 322 334 321 333 344 332 331 343 356 357 358 359 360 348 361 3
3 372 371 370 369 368 367 366 355 342 354 341 353 340 352 351 350 338 0 234 245
256 267 279 291 304 316 329 365 364 362 375 374 363 339 327 314 302 290 278 266
255 244 225 0 216 224 242 233 253 243 265 254 277 289 300 313 325 349 337 324 3
2 323 335 346 345 320 308 295 283 270 0 206 213 229 221 239 250 262 274 287 299
347 336 301 276 264 252 232 241 223 209 215 203 198 202 0 204 218 210 226 236 2
6 258 269 268 281 293 306 318 330 317 305 292 280 257 235 217 211 199 310 0 183
195 205 219 227 247 259 282 328 315 326 303 294 307 319 296 284 271 260 248 237
228 212 200 192 175 0 182 177 194 185 208 222 231 263 288 275 251 240 230 214 1
7 207 187 191 181 0 170 180 196 190 186 220 249 272 285 297 298 286 273 261 238
201 188 189 193 176 184 164 0 49 61 73 85 97 109 120 131 141 159 150 166 173 30
 178 172 158 140 130 119 108 96 84 72 60 48 0 32 43 31 42 30 41 29 40 28 39 51
3 75 87 99 111 122 133 142 152 160 151 167 132 121 110 98 86 74 62 50 0 25 26 1
 27 16 17 52 53 65 54 78 55 66 77 89 101 143 123 112 100 88 76 64 18 19 20 8 7
 5 4 3 0 46 58 70 82 94 106 117 128 147 138 148 157 171 165 149 139 129 118 107
95 83 71 59 47 0 92 104 115 135 126 145 154 168 162 174 179 169 155 137 146 127
136 116 105 93 81 0 57 68 80 91 103 124 134 144 161 163 156 153 125 114 113 102
90 79 67 56 0 1 12 13 14 9 38 37 36 24 35 23 34 22 33 21 44 45 69 11 10 2 0
1194.053955
```

***<u>Study case</u>* <u>In13</u>**

Hasil *running* metode *Random*:

```
Hasil Random: 0 47 88 307 371 302 419 344 87 404 267 0 67 119 112 331 116 68 338
 95 103 92 192 171 12 0 82 316 250 317 322 285 175 138 78 164 252 50 132 0 166 3
55 109 120 218 388 291 243 148 15 77 30 0 64 176 313 174 85 73 107 318 314 72 26
 259 151 0 145 220 10 121 191 140 310 224 0 81 190 412 122 333 214 142 347 304 0
 55 186 365 370 89 104 111 372 417 294 0 262 309 74 153 374 83 117 152 1 335 53
54 179 0 136 375 76 160 311 300 231 367 43 395 0 364 48 225 277 260 249 115 386
126 0 384 382 235 35 345 187 383 312 332 274 36 208 319 265 0 134 401 52 320 216
 198 315 329 221 61 199 324 59 195 0 233 373 162 261 411 196 352 197 271 62 183
137 0 20 129 350 46 376 71 184 90 21 354 0 178 323 19 236 100 394 124 155 139 7
379 0 65 125 405 27 269 282 172 38 292 272 13 94 0 34 286 135 402 130 325 287 8
254 194 0 293 240 359 228 211 299 239 57 409 105 305 385 393 0 91 128 185 102 24
 63 280 328 381 0 275 230 163 75 5 227 321 283 264 346 281 110 392 0 308 200 363
 156 337 380 415 23 39 330 79 0 410 157 146 342 181 177 33 193 295 270 296 158 2
15 0 101 32 414 150 144 348 44 49 205 98 106 37 266 133 413 29 168 290 0 96 97 8
6 289 31 169 22 369 25 159 17 173 223 14 0 396 202 398 180 298 306 210 226 66 20
1 0 336 131 326 229 327 273 246 377 301 51 0 276 257 339 389 70 189 69 141 255 0
 340 251 253 217 356 353 341 2 242 418 0 42 387 165 207 16 407 113 114 170 343 1
23 99 93 0 18 40 278 238 358 127 362 203 351 256 0 400 204 244 247 397 213 108 1
47 245 391 0 161 334 80 154 303 406 399 284 222 206 56 403 0 390 416 288 279 361
 237 268 378 258 209 11 0 368 263 143 219 60 41 234 45 149 349 232 0 241 420 366
 360 3 0 9 6 58 248 4 167 0 28 297 212 182 357 84 408 188 118 0
9710.074219
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 395 393 360 379 391 363 359 394 0 130 400 376 382 392 368 381 411
 399 371 0 195 415 373 380 416 369 378 390 387 419 357 413 377 408 0 260 410 375
 414 396 362 418 361 409 374 406 0 325 358 383 420 372 403 388 417 365 398 389 4
12 0 84 386 405 370 401 367 407 402 384 404 397 0 300 364 366 385 1 12 13 14 0 1
56 2 3 4 5 15 16 0 221 6 7 8 9 17 18 0 228 10 11 19 20 21 22 23 24 25 26 27 0 77
 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 0 293 43 44 45 46 47 48 49 50 51 5
2 53 54 55 56 57 59 0 149 58 60 61 62 63 72 0 161 64 66 67 68 73 74 76 0 89 69 7
0 71 75 78 79 80 81 0 305 82 83 85 86 87 88 90 91 92 93 94 95 0 190 96 97 98 99
100 101 102 103 104 105 106 107 108 0 158 109 110 111 112 113 114 115 116 117 11
8 119 120 132 0 125 121 122 123 124 0 230 126 127 128 129 133 134 0 233 131 135
136 137 138 139 140 141 142 143 144 145 147 0 255 146 148 150 151 152 153 154 15
5 157 159 160 162 163 0 302 164 165 166 167 168 169 170 171 172 173 174 175 176
0 306 177 178 179 180 181 182 0 231 183 184 185 186 192 193 194 0 234 187 188 18
9 191 196 197 199 0 303 198 200 201 202 203 204 205 206 207 208 209 210 212 0 30
7 211 213 214 215 216 217 218 219 220 222 223 224 0 232 225 226 227 229 235 236
237 238 239 240 252 253 254 0 304 241 242 243 244 0 301 245 246 247 248 0 292 24
9 250 251 256 257 258 259 261 262 0 311 263 264 265 266 267 268 269 270 271 272
273 274 0 335 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 291 0
298 290 294 295 296 297 299 308 309 312 313 0 334 310 314 315 316 317 318 319 32
0 321 322 323 324 0 336 326 327 328 329 330 331 332 333 337 338 339 340 341 342
343 0 356 344 345 346 347 348 349 350 351 352 353 354 355 0
6528.455566
```

Hasil *running* metode *Farthest to Nearest*:

```
Hasil FTNy: 0 315 316 375 376 371 419 311 359 251 0 250 351 310 416 370 411 410
387 386 385 326 0 342 343 402 403 369 408 309 348 249 0 339 340 399 400 368 395
366 384 306 324 246 0 196 255 256 191 299 358 418 417 412 357 298 239 179 0 223
7 346 287 228 168 0 220 279 280 188 275 334 394 393 388 333 274 215 155 0 199 25
322 263 204 144 0 236 295 354 405 414 415 356 355 296 190 291 290 231 0 212 271
7 267 266 207 0 103 163 222 281 336 341 401 396 363 389 338 329 278 219 160 0 21
243 269 0 79 139 198 257 312 317 377 372 360 413 314 353 254 195 136 0 237 292 3
 0 285 245 345 365 305 294 235 176 171 111 0 226 227 286 337 304 364 404 349 344
 302 270 211 152 147 87 0 202 203 262 313 301 361 380 325 320 265 206 205 0 164
6 0 140 200 145 146 127 67 92 7 27 32 0 209 183 276 221 162 129 69 108 167 0 133
 0 157 124 217 284 244 277 184 166 107 48 0 85 86 260 241 253 181 121 82 23 0 78
 119 178 0 99 158 218 159 100 8 95 154 153 208 182 0 12 72 60 132 120 197 138 77
7 232 185 225 234 175 174 115 114 0 62 141 122 201 210 151 150 91 90 31 30 0 20
5 14 53 52 0 101 156 123 149 98 39 40 68 94 35 34 0 36 96 41 42 102 216 161 43 9
25 26 81 2 28 0 54 55 44 49 50 110 109 104 64 4 45 0 5 105 65 165 125 172 117 58
9 63 3 29 0
2013.905884
```

***<u>Study case</u>* <u>In14</u>**

Hasil *running* metode *Random*:

```
Hasil Random: 0 152 250 394 29 195 286 337 384 240 107 364 269 164 361 50 328 42
2 218 16 116 43 274 205 144 268 81 0 66 414 183 448 321 194 70 24 11 396 142 154
 232 14 63 147 156 132 53 455 174 0 151 169 106 139 189 34 330 477 23 451 465 29
7 406 313 311 27 217 68 349 211 176 292 378 206 95 180 150 471 316 228 0 238 246
 351 358 346 173 375 383 166 127 90 120 65 433 210 143 264 432 276 392 257 412 6
7 439 254 266 357 227 6 0 109 153 333 248 170 400 380 453 134 179 306 100 401 69
 79 245 343 162 200 249 221 204 247 10 98 33 0 30 415 424 483 96 457 352 149 7 3
91 416 78 411 425 201 452 447 470 0 363 273 129 141 251 435 427 397 468 369 259
202 475 121 155 284 271 20 256 327 3 223 62 476 0 288 181 188 54 399 289 198 214
 213 417 472 367 118 117 404 140 163 408 178 293 342 124 407 320 434 413 203 454
 381 92 389 309 91 326 0 308 244 393 285 61 130 131 71 32 182 239 441 302 356 37
2 379 171 28 319 87 480 159 18 76 469 0 461 371 111 72 421 243 22 209 362 122 44
6 136 296 340 237 339 462 229 287 19 398 158 267 230 386 191 290 49 0 220 437 11
5 37 431 325 350 38 482 126 88 466 263 52 208 279 368 224 85 295 344 15 215 44 3
12 145 0 47 46 317 458 226 382 148 55 260 84 359 137 436 114 135 442 298 59 336
305 56 377 64 459 329 318 133 93 0 444 449 419 463 341 438 168 105 460 9 110 48
103 265 360 234 39 304 467 13 2 0 172 99 94 299 335 473 80 222 51 258 428 236 42
6 353 17 301 262 354 235 403 418 253 294 322 82 479 278 272 252 0 324 177 104 21
9 4 167 40 197 282 303 225 376 199 347 119 366 275 192 430 112 231 314 12 0 283
388 348 338 478 307 281 334 108 36 31 102 60 374 373 196 160 280 77 443 387 355
83 445 291 450 410 185 0 385 21 1 315 45 456 323 125 193 216 300 409 146 332 74
270 186 474 25 395 0 5 440 310 41 57 187 75 190 86 241 128 175 390 26 89 207 261
 464 35 113 138 157 42 0 73 331 212 420 345 277 429 184 255 365 423 402 161 97 1
65 405 481 242 58 101 8 370 233 123 0
8191.308594
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 443 441 439 446 444 442 440 447 445 433 438 434 430 436 432 428 4
35 431 437 424 429 425 421 427 423 420 426 422 483 450 0 130 475 1 456 477 454 4
81 448 476 451 478 462 479 453 480 452 482 449 469 2 457 470 20 0 195 474 3 458
471 4 459 472 5 460 473 6 461 0 260 467 7 468 8 463 9 464 10 465 11 466 13 455 0
 325 12 14 15 16 17 18 19 21 22 44 0 390 23 24 25 26 27 28 29 30 31 32 33 76 0 6
4 34 35 36 37 38 39 40 41 42 43 45 46 47 48 0 67 49 50 51 52 53 54 55 56 57 58 5
9 60 61 62 63 66 189 0 68 69 70 71 72 73 74 75 77 78 79 80 81 82 83 84 85 86 87
0 133 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 10
9 113 0 134 110 111 112 114 115 116 117 118 119 120 121 122 123 124 125 126 127
128 129 131 132 135 136 137 230 0 193 138 139 140 141 142 143 144 145 146 147 14
8 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 0 198 164 165 166
167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186
187 188 190 191 192 194 196 0 199 197 200 201 202 203 204 205 206 207 208 209 21
0 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 23
1 0 257 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249
250 251 252 253 254 255 256 258 259 261 262 263 264 265 266 271 0 322 267 268 26
9 270 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 29
0 291 292 293 294 295 296 297 298 299 300 301 302 303 0 329 304 305 306 307 308
309 310 311 312 313 314 315 316 317 318 319 320 321 323 324 326 327 328 330 331
332 333 334 335 336 337 338 339 340 341 342 343 345 0 386 344 346 347 348 349 35
0 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 37
0 371 372 373 374 375 376 377 378 379 380 381 382 383 391 0 387 384 385 388 389
392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411
412 413 414 415 416 417 418 419 0
3884.632080
```

Hasil *running* metode *Farthest to Nearest*:

```
Hasil FTNx: 0 190 210 231 253 274 294 313 331 348 364 379 393 406 418 429 439 44
480 477 473 468 462 455 447 438 428 417 405 392 378 363 347 330 312 293 273 252
276 296 315 333 350 366 381 395 408 420 431 441 450 458 465 471 476 479 475 470
4 380 365 349 332 314 295 275 254 232 211 191 0 214 235 257 278 298 317 335 352
2 460 467 472 466 459 451 442 432 421 409 396 382 367 351 334 316 297 277 256 23
9 280 300 319 337 354 370 385 399 412 424 435 445 454 461 453 444 434 423 411 39
258 236 215 194 174 0 177 197 218 239 261 282 302 321 339 356 372 387 401 414 42
371 355 338 320 301 281 260 238 217 196 176 0 160 179 199 219 241 263 284 304 32
415 402 388 373 357 340 322 303 283 262 240 198 178 159 158 0 127 144 162 181 20
360 376 391 404 390 375 359 342 324 305 285 264 242 220 180 161 143 126 0 98 113
67 288 308 327 345 362 377 361 344 326 307 287 266 244 222 201 163 145 128 112 9
184 204 225 247 269 290 310 329 346 328 309 289 268 246 224 203 183 147 130 114
 133 149 167 186 206 227 249 271 292 311 291 270 248 226 205 185 166 132 116 101
 134 151 169 188 208 229 251 272 250 228 207 187 168 150 118 89 64 43 42 33 25 0
20 136 135 171 152 119 104 90 77 65 54 44 35 27 20 0 38 47 57 68 80 93 107 122 1
 92 79 56 46 37 69 0 49 59 70 82 95 109 124 140 157 195 175 156 139 123 108 94 8
1 110 142 141 125 96 39 30 0 11 17 24 41 84 61 32 18 12 19 0 6 10 15 22 29 21 14
0 483 0
1193.844971
```

***Study case* In15**

Hasil *running* metode *Random*:

```
Hasil Random: 0 33 67 331 3 409 75 93 125 45 117 210 16 7 255 392 113 477 573 14
8 228 128 482 66 475 175 261 179 533 196 523 20 166 367 10 336 74 153 296 312 51
4 142 571 136 0 418 354 355 185 205 394 56 406 395 50 539 347 447 565 527 315 44
4 177 58 341 370 44 151 432 27 448 483 92 288 184 214 282 369 240 254 237 513 0
500 187 264 158 566 144 22 520 398 169 391 161 154 112 222 15 316 231 350 467 24
7 489 382 121 13 103 470 306 504 368 364 534 375 446 152 436 453 122 530 0 576 3
38 518 104 203 171 276 55 510 275 251 547 574 333 419 468 332 569 233 281 147 43
5 457 431 463 218 310 290 311 135 285 532 359 461 597 327 42 266 324 18 321 0 24
4 270 293 193 47 70 90 118 256 490 127 12 190 73 217 4 197 372 302 120 280 263 4
22 65 202 377 81 30 39 531 371 334 536 239 299 525 21 388 0 79 498 5 32 146 400
235 295 384 379 199 271 245 438 164 8 246 43 452 445 195 322 138 450 139 449 269
 351 570 268 227 352 357 551 85 508 249 124 294 421 38 588 528 173 0 87 72 292 1
16 381 583 89 108 78 19 284 131 167 480 339 549 485 561 277 402 587 37 430 455 4
11 380 415 537 64 397 420 225 401 374 494 515 36 53 301 0 496 465 594 314 191 19
4 362 143 211 97 207 516 300 69 495 178 232 519 174 346 188 80 114 410 443 126 3
37 356 474 538 163 412 57 408 216 272 580 46 497 94 344 414 0 71 209 88 596 559
546 262 259 76 325 29 595 416 172 424 40 61 522 141 317 586 383 34 600 535 297 4
05 473 226 253 68 517 487 433 552 503 313 329 241 287 28 0 186 593 133 130 590 1
65 257 318 555 230 101 156 378 62 529 460 180 579 14 484 570 24 248 51 428 95 40
4 236 198 176 330 307 220 298 54 77 544 162 157 471 41 96 459 99 35 564 0 223 27
8 215 582 349 111 260 575 23 403 585 221 550 83 25 265 168 526 592 478 286 567 5
24 393 273 319 17 462 224 274 82 0 413 387 476 511 213 363 342 492 353 584 328 3
58 481 557 434 134 159 200 98 365 601 208 479 502 192 591 440 140 558 106 562 10
9 155 501 212 491 0 229 417 348 234 464 373 390 553 305 6 182 105 389 345 442 18
1 577 304 361 458 145 563 343 572 509 512 540 107 366 63 385 160 150 599 238 119
 429 289 291 86 0 541 219 568 426 486 454 323 340 469 456 437 279 242 309 386 49
 556 110 170 149 326 545 560 11 206 335 542 48 189 441 91 506 0 204 115 303 9 20
1 283 466 543 554 250 129 581 31 123 137 360 488 2 52 423 548 598 243 439 132 58
9 84 505 0 60 425 472 407 376 451 427 399 59 1 499 320 100 252 183 267 26 521 30
8 258 493 507 102 396 0
12729.460938
```

Hasil *running* metode *Nearest Neighbor*:

```
Hasil NN: 0 65 574 572 573 566 570 571 564 568 569 575 559 561 565 562 558 563 5
67 560 554 557 555 556 552 553 549 548 550 543 544 545 542 546 538 547 540 551 5
39 541 600 585 597 590 598 578 596 594 0 130 595 588 1 586 601 592 2 587 599 580
 593 583 591 3 581 4 589 5 584 6 576 582 7 8 579 9 577 10 11 12 13 14 15 16 17 1
8 19 20 21 22 23 0 195 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42
43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 66 67 68 69 70
 71 75 76 0 260 72 73 74 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 9
5 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 11
6 117 118 119 0 325 120 121 122 123 124 125 126 127 128 129 131 132 133 134 135
136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155
156 157 158 159 160 161 162 163 164 165 166 167 170 0 390 168 169 171 172 173 17
4 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 19
4 196 197 198 199 200 201 202 203 204 205 207 224 0 455 206 208 209 210 211 212
213 214 215 216 217 218 219 220 221 222 223 225 226 227 228 229 230 231 232 233
234 235 236 237 238 239 255 0 520 240 241 242 243 244 245 246 247 248 249 250 25
1 252 253 254 256 257 258 259 0 332 261 262 263 264 265 266 267 268 269 270 271
272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 0 397 288 289 29
0 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 31
0 311 312 313 314 315 316 317 318 320 0 514 319 321 322 323 324 326 327 328 329
330 331 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350
351 352 353 354 355 356 357 358 0 457 359 360 361 362 363 364 365 366 367 368 36
9 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 38
9 391 392 394 0 516 393 395 396 398 399 400 401 402 403 404 405 406 407 408 409
410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429
430 0 527 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 44
8 449 450 451 452 453 454 456 458 459 460 461 462 463 464 465 466 467 468 469 47
1 0 524 470 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488
489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508
509 510 519 0 526 511 512 513 515 517 518 521 522 523 525 528 529 530 531 532 53
3 534 535 536 537 0
2436.826172
```

Hasil *running* metode *Farthest to Nearest*:

```
C:\Users\Guest\Desktop\KULIAH\FTN>ftnfinal<in15
Hasil FTNs: 0 523 529 537 540 552 545 541 553 559 565 569 573 577 591 585 589 5
550 561 562 548 495 510 527 526 512 506 490 480 471 449 461 470 493 505 489 479
353 361 373 401 429 413 405 402 430 453 459 587 474 387 356 268 260 222 197 259
9 351 319 290 223 201 298 409 329 229 171 125 96 139 179 151 183 161 0 335 363
 482 498 511 522 521 509 556 557 549 570 583 590 595 588 579 582 592 598 601 59
 406 396 410 424 434 446 457 455 447 435 426 415 399 384 344 333 313 296 284 29
17 302 294 257 231 219 193 187 169 143 135 131 118 114 237 163 103 72 61 0 460
 563 567 574 584 578 575 568 564 542 546 554 543 538 533 525 519 504 488 497 50
96 485 477 465 438 0 380 394 414 422 433 444 466 454 456 445 467 478 487 518 53
91 481 472 450 428 437 400 385 372 345 285 270 254 233 227 217 199 235 255 0 23
19 441 463 476 483 514 535 558 576 572 530 515 500 443 421 432 442 431 420 392
67 388 395 381 368 354 341 324 311 325 342 355 359 369 382 397 411 407 425 468
 378 403 417 376 348 318 288 203 218 191 189 177 0 198 226 232 248 269 291 279
 371 343 383 370 312 282 251 213 250 212 249 263 241 224 211 210 234 261 320 0
111 117 123 147 185 153 157 165 181 173 167 149 145 141 155 158 175 133 127 121
00 89 80 66 58 48 62 73 65 64 91 82 0 209 247 278 308 332 214 252 283 216 225 2
326 338 360 365 352 337 307 300 277 246 0 188 190 366 393 379 276 245 207 194 2
273 0 297 200 204 267 242 196 205 243 304 416 362 275 208 192 186 0 126 140 144
0 129 115 112 106 83 94 98 107 137 70 51 53 45 33 26 24 11 18 28 37 34 35 55 88
32 0 16 25 17 13 22 19 39 20 7 9 3 1 6 23 40 31 44 52 47 46 54 60 71 102 113 15
22 116 110 104 119 132 136 156 164 180 176 152 146 130 0 2 8 27 10 5 12 21 15 2
08 328 236 162 124 95 38 14 4 0
2355.433350
```

HASIL *RUNNING PROGRAM*

HASIL RUNNING PROGRAM

Bagian ini menampilkan hasil running program untuk 15 *study cases*, yang direkap dalam Tabel 1. Dari Tabel 1. terbukti bahwa metode FTN menghasilkan rute dengan total jarak yang paling baik.

Tabel 1. Rekap hasil running program

Cases	Metode	Total Jarak	Rute
in	*Random*	83.99	0-3-1-0-4-2-0
	NN	82.13	0-2-1-0-3-4-0
	FTN	74.90	0-4-1-0-3-2-0
in2	*Random*	338.46	0-2-16-13-14-1-0-5-6-9-15-0-8-3-0-10-12-0-7-4-0-11-0
	NN	320.09	0-10-2-0-16-15-14-3-1-0-11-12-13-0-9-6-5-0-7-4-0-8-0
	FTN	277.61	0-9-5-6-13-0-1-8-4-0-7-0-15-3-14-16-0-2-10-0-12-11-0
in3	*Random*	935.04	0-25-8-21-9-11-0-6-19-7-14-0-24-15-2-4-0-17-1-20-0-12-13-23-0-5-16-22-0-10-0-3-0-18-0
	NN	719.32	0-12-17-7-16-0-3-2-0-23-1-14-6-0-22-18-24-0-8-9-11-13-0-10-4-25-0-15-19-0-5-20-0-21-0
	FTN	708.42	0-13-21-20-14-0-2-25-4-10-0-23-24-15-0-1-18-22-0-6-5-7-0-19-17-0-16-12-0-9-11-8-0-3-0
in4	*Random*	147,000.67	0-18-3-35-22-16-39-29-26-21-20-8-6-47-33-23-0-5-7-40-46-30-13-43-14-11-44-41-32-24-38-28-0-10-37-36-27-42-15-9-31-2-17-25-4-34-12-19-0-45-1-0
	NN	52,006.22	0-14-23-11-12-15-32-45-43-18-7-27-35-29-6-36-0-13-21-46-20-39-9-1-8-37-30-19-26-42-17-3-0-38-31-47-5-28-2-41-10-24-44-34-25-4-33-40-0-22-16-0
	FTN	45,841.71	0-45-35-29-42-17-26-19-36-6-27-7-18-43-30-37-0-31-38-47-24-10-44-34-25-4-41-5-28-2-33-14-0-13-21-46-20-32-12-15-39-9-8-1-22-16-3-23-0-40-11-0
in5	*Random*	1,568,417.75	0-33-3-40-14-46-27-47-0-36-16-4-9-42-0-22-35-2-49-31-12-20-0-41-24-48-13-28-0-6-45-29-8-34-0-1-37-7-26-30-19-15-0-11-18-43-39-32-21-44-0-5-25-17-23-10-0-38-0
	NN	1,164,397.88	0-25-21-46-8-33-0-19-44-1-26-18-43-47-0-22-28-3-6-31-17-7-0-42-32-35-10-9-23-15-0-34-12-37-48-40-0-13-38-2-27-20-39-49-0-45-14-24-30-29-0-5-4-36-41-16-0-11-0
	FTN	1,038,446.88	0-29-4-41-36-5-0-49-14-30-16-11-24-39-0-23-38-2-45-20-27-15-0-12-48-13-40-37-0-31-7-35-34-9-10-32-0-47-3-6-42-17-28-43-0-19-26-22-18-1-0-21-33-44-8-46-0-25-0
in6	*Random*	4,321.05	0-33-28-26-8-0-15-77-58-0-57-74-14-0-82-3-56-47-61-46-0-64-86-81-69-0-70-75-0-37-88-50-0-76-60-36-22-1-0-87-52-34-7-0-80-24-71-0-35-21-48-51-0-23-39-38-0-62-44-12-67-0-68-27-43-72-0-53-40-66-41-0-2-45-78-0-30-32-63-0-55-84-11-0-6-79-65-0-59-9-31-16-0-29-83-73-0-49-5-0-20-17-18-0-19-25-0-10-4-0-42-54-13-0-85-0
	NN	2,731.56	0-65-63-62-0-66-61-60-58-0-67-59-57-56-51-0-68-55-52-0-69-54-47-0-70-53-46-86-0-71-64-50-0-72-49-48-81-0-73-45-87-0-74-85-26-0-75-88-82-0-76-84-83-27-0-77-80-78-0-79-25-31-0-22-28-29-36-0-23-30-32-0-21-33-34-37-0-24-35-41-0-20-38-42-0-19-39-1-6-0-18-40-2-0-17-43-44-11-0-16-3-4-7-0-15-5-0-14-8-12-0-13-9-0-10-0
	FTN	2,029.84	0-56-47-45-46-51-0-52-48-49-61-0-57-50-53-0-66-54-55-0-42-44-43-41-0-3-1-2-4-0-62-58-59-71-0-67-60-63-0-36-40-39-0-16-7-5-6-11-0-35-38-37-0-12-8-9-21-0-17-10-13-0-32-34-33-31-0-76-64-65-0-26-14-15-0-30-29-0-25-28-27-0-22-18-19-0-20-23-0-24-0-72-68-69-81-0-77-70-73-0-86-74-75-0-82-78-79-0-87-80-83-0-84-85-0-88-0
in7	*Random*	20,262.85	0-80-121-8-131-124-140-44-78-117-67-37-5-102-115-9-111-114-110-31-106-79-14-0-71-1-92-43-75-59-126-49-48-47-29-57-13-99-63-104-34-122-88-4-56-86-16-100-112-123-12-39-0-107-127-96-105-19-2-135-58-116-95-18-41-28-20-52-137-38-24-7-70-81-54-21-109-17-42-82-65-0-11-23-72-25-26-27-22-61-89-87-40-84-30-83-101-90-129-66-46-113-62-15-0-53-33-85-6-128-51-120-45-103-68-134-133-138-136-36-130-3-55-60-64-35-108-0-97-93-98-119-77-32-125-69-94-50-76-132-139-10-91-0-118-73-74-0
	NN	10,289.80	0-65-127-126-125-124-123-122-121-120-119-118-117-116-115-114-113-112-111-110-109-108-107-106-105-104-103-102-101-0-130-100-99-98-97-96-95-94-93-92-91-90-89-88-87-86-85-84-83-82-81-80-79-78-12-131-13-129-0-1-132-77-2-133-3-134-4-135-5-136-6-137-7-128-14-0-64-140-139-138-8-9-10-11-15-16-17-18-19-20-21-22-23-24-25-26-27-34-0-66-28-29-30-31-32-33-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-0-67-50-51-52-53-54-55-56-57-58-59-60-61-62-63-68-69-70-71-72-73-0-74-75-76-0

	FTN	2,845.39	0-105-107-109-120-122-124-126-128-130-132-134-136-138-140-139-137-135-133-131-129-127-125-123-121-119-108-106-104-0-84-86-96-98-100-102-110-112-114-116-118-117-115-113-111-103-101-99-97-87-85-83-0-63-65-74-76-78-80-82-89-91-93-95-94-92-90-88-81-79-77-75-73-64-62-0-42-50-52-54-56-58-60-66-68-70-72-71-69-67-61-59-57-55-53-51-43-41-0-21-28-30-32-34-36-38-40-45-47-49-48-46-44-39-37-35-33-31-29-27-20-0-4-6-8-10-12-14-16-18-22-24-26-25-23-19-17-15-13-11-9-7-5-0-1-3-2-0
in8	*Random*	4,419.46	0-148-146-15-80-153-147-160-167-176-22-91-82-130-64-182-99-129-50-0-92-112-191-27-68-170-121-186-65-49-6-144-172-0-55-29-163-119-8-74-89-76-120-111-195-164-151-103-188-0-11-171-197-35-41-18-145-2-136-95-63-87-138-177-45-105-152-107-0-28-23-86-16-104-135-56-123-139-118-198-77-175-25-13-83-0-59-70-159-90-109-75-30-85-125-137-116-0-37-166-43-44-3-33-21-110-67-52-93-69-98-0-143-20-61-156-101-36-199-1-114-178-131-122-32-0-40-149-181-73-201-46-12-158-150-185-180-174-102-0-141-106-24-72-200-194-97-53-140-133-62-96-9-14-47-127-4-54-0-183-154-117-193-115-79-128-51-196-58-19-126-187-0-173-10-142-157-7-189-161-124-100-60-134-66-38-165-113-0-184-132-5-190-155-31-71-168-48-26-78-108-88-84-57-0-169-179-192-94-81-42-39-162-34-17-0
	NN	3,711.70	0-65-182-185-174-190-187-167-188-186-183-189-171-181-179-178-0-130-162-180-161-176-164-177-175-172-170-153-149-166-169-168-152-0-195-147-157-165-144-163-159-160-146-184-158-155-156-138-151-143-0-67-154-150-140-173-200-201-199-198-191-142-148-145-197-1-139-2-29-0-193-196-141-194-3-4-192-5-6-7-8-9-10-15-0-128-11-12-13-14-16-17-18-19-20-21-40-0-132-22-23-24-25-26-27-28-30-31-32-33-34-35-36-47-50-53-0-135-37-38-39-41-42-43-44-45-46-51-54-0-133-48-49-52-55-56-57-58-59-60-61-62-63-64-66-69-86-107-0-137-68-70-71-72-73-74-75-76-77-78-79-127-0-136-80-81-82-83-84-85-87-88-89-90-91-92-95-0-131-93-94-96-97-98-99-100-101-102-103-104-113-129-0-134-105-106-108-109-110-111-112-114-115-116-123-0-120-117-118-119-121-122-124-125-126-0
	FTN	1,498.27	0-170-175-180-184-189-193-197-201-199-195-191-187-182-178-172-158-0-153-165-173-176-183-185-196-200-198-192-186-177-181-168-166-0-80-57-74-125-87-109-81-140-141-137-134-132-130-126-0-151-169-179-190-194-188-174-171-155-152-149-147-143-160-0-53-43-34-36-33-29-24-18-10-6-46-21-25-19-16-8-4-0-107-112-116-121-128-162-167-164-145-123-118-114-110-88-0-96-98-100-103-105-139-131-129-150-161-159-146-135-119-0-65-111-115-120-122-136-138-154-163-156-157-133-117-104-0-56-73-86-148-124-99-63-45-78-82-108-113-89-93-97-127-50-0-64-75-77-79-101-106-91-95-102-144-142-84-76-0-55-60-66-70-83-90-94-92-85-72-68-62-58-59-0-40-42-48-54-71-67-61-52-51-49-47-44-38-41-0-22-26-28-30-32-37-35-31-12-39-27-23-14-0-5-9-13-17-69-20-15-11-7-3-1-2-0
in9	*Random*	4,759.54	0-61-180-56-149-93-185-107-99-240-54-43-90-162-69-3-207-0-76-33-135-63-74-231-230-64-117-153-118-31-55-100-157-0-201-23-139-208-132-181-233-192-70-1-111-200-227-148-78-0-108-152-177-77-12-94-218-190-161-30-62-28-73-51-226-75-0-113-232-57-221-213-239-234-65-134-199-124-68-11-112-0-171-109-67-211-27-82-123-174-45-119-166-154-121-84-0-183-220-182-92-25-237-189-195-96-6-26-88-142-95-173-104-72-0-38-81-89-194-178-191-238-106-163-102-219-244-176-103-0-29-204-46-86-214-137-205-245-187-167-2-202-105-212-37-138-0-222-241-48-22-155-32-147-210-224-130-223-164-144-151-14-0-8-18-143-128-80-159-242-58-83-175-53-170-188-114-0-160-203-91-24-248-186-34-47-126-4-5-158-168-16-116-21-140-0-44-156-36-9-59-101-42-98-198-49-209-225-0-236-40-71-19-146-15-216-150-85-206-115-193-0-172-10-35-13-127-247-217-145-87-229-110-0-131-228-246-243-52-120-179-60-196-39-133-66-125-0-20-7-136-184-122-215-17-197-165-41-235-97-141-0-169-79-129-50-0
	NN	3,802.24	0-65-243-193-186-188-248-239-213-212-194-227-195-207-0-130-235-197-189-223-215-192-191-190-224-211-240-201-185-226-0-4-199-246-198-187-247-196-245-208-200-241-210-209-202-242-214-204-1-219-203-0-15-237-222-221-206-216-217-238-236-205-2-230-220-218-231-225-0-159-244-232-234-228-233-229-3-5-6-7-8-0-96-9-10-11-12-13-14-16-17-18-19-0-27-20-21-22-23-24-25-26-28-29-30-31-32-33-34-49-0-42-35-36-37-38-39-40-41-43-50-0-52-44-45-46-47-48-51-53-54-55-56-57-58-59-60-61-0-126-62-63-64-66-67-68-69-70-71-77-0-181-72-73-74-75-76-78-79-80-81-82-83-84-85-86-0-118-87-88-89-90-91-92-93-94-95-97-98-99-100-0-135-101-102-103-104-105-106-107-108-109-110-111-113-0-163-112-114-115-116-117-119-120-121-122-123-124-125-127-0-150-128-129-131-132-133-134-136-137-138-139-140-141-142-143-144-0-164-145-146-147-148-149-151-152-153-154-155-156-157-158-167-0-177-160-161-162-165-166-168-169-170-171-172-173-174-175-0-179-176-178-180-182-183-184-0
	FTN	908.82	0-232-243-234-223-217-212-219-228-222-215-209-207-211-221-245-236-237-238-0-240-235-225-218-213-220-229-226-216-210-208-214-227-246-242-241-233-0-173-180-194-200-203-206-199-197-192-139-154-166-167-156-121-0-135-151-157-171-195-201-204-205-202-198-196-189-178-170-0-248-231-224-230-239-247-244-0-49-22-7-2-53-31-118-61-35-8-0-20-23-48-21-62-1-54-32-119-140-155-172-104-98-75-47-0-115-106-94-116-125-136-126-117-193-96-74-58-50-19-46-18-6-0-10-9-57-36-63-97-103-85-64-37-0-38-76-65-86-107-120-127-108-87-105-0-129-137-144-159-168-184-174-182-185-190-186-175-160-145-141-0-67-83-109-158-143-128-88-59-39-25-11-0-124-134-138-150-165-181-179-164-149-133-142-148-153-162-169-176-0-112-132-163-183-191-188-177-187-161-152-146-130-122-110-89-0-41-55-51-68-78-33-66-77-5-24-12-40-26-13-3-0-43-34-44-60-72-81-82-95-73-45-30-17-29-16-28-0-42-69-90-123-147-131-111-101-99-79-56-52-70-100-0-4-14-27-113-91-102-80-92-114-84-93-71-15-0

in10	*Random*	49,135.43	0-168-288-109-160-243-236-217-167-261-118-35-201-1-110-2-27-212-123-56-286-0-170-54-125-140-205-210-57-113-188-177-163-92-175-292-66-277-52-271-246-16-0-103-90-46-259-75-85-42-49-151-48-273-39-274-231-149-18-257-249-115-102-0-112-237-91-89-252-88-295-187-155-166-226-165-122-101-8-180-114-296-15-70-0-11-26-133-20-230-178-154-285-124-80-176-194-36-121-208-183-267-233-162-279-0-147-144-272-3-206-136-98-289-138-69-225-87-204-139-240-223-71-253-93-232-0-209-76-239-228-105-82-99-256-284-86-84-250-244-25-255-145-21-117-96-224-0-262-107-9-137-31-45-100-173-33-254-47-134-55-290-73-6-179-215-164-266-0-200-77-221-120-264-108-218-38-83-207-30-234-126-172-67-41-10-141-189-227-0-242-247-184-278-37-53-50-148-13-116-152-72-297-97-150-94-293-182-131-238-0-260-245-7-159-222-17-22-203-294-62-195-79-28-61-142-51-104-68-106-280-0-59-40-60-185-192-29-24-282-111-127-193-216-287-281-235-161-5-196-146-119-0-129-81-191-213-186-153-74-275-43-270-229-65-258-251-174-158-14-283-44-219-0-78-198-291-202-157-63-132-32-64-128-265-241-276-268-4-269-34-199-135-58-0-156-190-181-263-19-197-95-169-130-171-248-220-12-211-23-214-143-0
	NN	23,871.89	0-65-255-254-253-252-251-250-249-248-247-246-245-244-243-242-241-240-239-238-237-0-130-236-235-234-297-296-295-294-293-292-291-290-289-288-287-286-285-284-283-282-0-195-281-280-279-1-261-2-262-278-3-263-4-264-5-265-6-266-7-267-8-0-260-9-269-10-270-11-271-12-272-13-273-14-274-15-275-16-276-17-277-18-0-64-19-268-20-259-21-258-22-257-23-256-24-25-26-27-28-29-30-31-32-0-66-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-0-129-52-53-54-55-56-57-58-59-60-61-62-63-67-68-69-70-71-72-73-0-131-74-75-76-77-78-79-80-81-82-83-84-85-86-87-88-89-90-91-92-0-194-93-94-95-96-97-98-99-100-101-102-103-104-105-106-107-108-109-110-111-0-196-112-113-114-115-116-117-118-119-120-121-122-123-124-125-126-127-128-132-133-0-193-134-135-136-137-138-139-140-141-142-143-144-145-146-147-148-149-150-151-152-0-197-153-154-155-156-157-158-159-160-161-162-163-164-165-166-167-168-169-170-171-0-192-172-173-174-175-176-177-178-179-180-181-182-183-184-185-186-187-188-189-190-0-198-191-199-200-201-202-203-204-205-206-207-208-209-210-211-212-213-214-215-216-0-217-218-219-220-221-222-223-224-225-226-227-228-229-230-231-232-233-0
	FTN	6,660.97	0-279-281-283-285-287-289-291-293-295-297-296-294-292-290-288-286-284-282-280-278-0-259-261-263-265-267-269-271-273-275-277-276-274-272-270-268-266-264-262-260-258-0-239-241-243-245-247-249-251-253-255-257-256-254-252-250-248-246-244-242-240-238-0-219-221-223-225-227-229-231-233-235-237-236-234-232-230-228-226-224-222-220-218-0-199-201-203-205-207-209-211-213-215-217-216-214-212-210-208-206-204-202-200-198-0-179-181-183-185-187-189-191-193-195-197-196-194-192-190-188-186-184-182-180-178-0-159-161-163-165-167-169-171-173-175-177-176-174-172-170-168-166-164-162-160-158-0-139-141-143-145-147-149-151-153-155-157-156-154-152-150-148-146-144-142-140-138-0-119-121-123-125-127-129-131-133-135-137-136-134-132-130-128-126-124-122-120-118-0-99-101-103-105-107-109-111-113-115-117-116-114-112-110-108-106-104-102-100-98-0-79-81-83-85-87-89-91-93-95-97-96-94-92-90-88-86-84-82-80-78-0-59-61-63-65-67-69-71-73-75-77-76-74-72-70-68-66-64-62-60-58-0-39-41-43-45-47-49-51-53-55-57-56-54-52-50-48-46-44-42-40-38-0-19-21-23-25-27-29-31-33-35-37-36-34-32-30-28-26-24-22-20-18-0-1-3-5-7-9-11-13-15-17-16-14-12-10-8-6-4-2-0
in11	*Random*	4,825.87	0-288-118-102-140-36-121-168-155-143-150-289-249-81-183-142-135-223-75-90-294-281-5-220-201-304-279-0-301-91-141-259-163-203-293-99-199-37-56-219-116-15-270-119-256-275-176-7-42-0-21-101-228-317-122-35-111-232-197-205-92-320-243-231-73-167-68-291-311-290-93-252-110-59-133-185-0-134-212-23-38-170-217-127-224-124-52-307-234-138-53-10-173-202-186-182-238-318-162-0-94-229-215-245-128-189-237-267-129-313-248-84-319-303-277-146-51-61-261-48-193-148-250-216-144-95-0-22-89-225-222-17-171-85-180-262-321-300-274-82-306-258-164-98-63-2-0-287-69-218-13-254-72-16-322-263-284-152-104-253-28-120-316-278-77-19-158-200-251-0-305-40-96-78-43-41-235-295-169-310-192-242-260-47-157-33-165-175-4-214-227-0-283-29-50-149-195-285-8-107-271-24-179-194-83-32-31-206-204-55-0-76-25-115-236-292-26-9-299-296-44-309-12-314-221-131-74-132-160-145-226-117-100-0-181-79-187-244-58-67-266-172-213-106-27-188-1-159-136-137-105-196-147-156-0-280-265-57-87-125-246-257-3-166-62-126-39-139-208-64-210-153-315-268-0-198-264-191-282-272-184-230-273-20-255-66-151-178-112-86-45-211-312-239-80-177-34-46-209-130-298-308-0-108-6-60-54-88-233-276-103-18-190-97-241-11-71-240-286-0-247-114-207-297-65-49-109-30-154-14-123-161-302-113-174-70-269-0
	NN	2,446.51	0-65-313-311-310-300-317-315-314-312-301-318-316-304-319-320-308-291-302-309-292-303-307-299-305-298-306-290-283-297-294-293-284-289-276-295-285-282-0-130-275-286-296-287-288-281-280-279-278-277-271-270-265-274-272-266-269-261-273-267-268-264-263-260-262-259-322-321-5-44-0-195-1-2-3-4-0-64-6-7-8-9-10-11-27-0-67-12-13-14-15-16-17-18-19-20-77-0-68-21-22-23-24-25-26-28-29-30-31-76-0-133-32-33-34-35-36-37-38-39-40-41-42-43-45-46-189-0-134-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-104-0-193-62-63-66-69-70-71-72-73-74-75-78-79-80-81-82-83-84-85-0-198-86-87-88-89-90-91-92-93-94-95-96-97-98-99-100-101-102-103-105-106-107-108-152-0-199-109-110-111-112-113-114-115-116-117-118-119-120-121-122-123-124-125-126-127-128-129-131-132-0-257-135-136-137-138-139-140-141-142-143-144-145-146-147-148-149-150-151-153-154-155-156-157-158-159-160-161-209-0-192-162-163-164-165-166-167-168-169-170-171-172-173-174-175-176-177-178-179-180-181-182-183-184-185-186-187-188-190-191-194-0-196-197-200-201-202-203-204-205-206-207-208-210-211-212-213-214-215-216-217-218-219-220-221-222-223-224-225-226-227-228-229-230-231-251-0-256-232-233-234-235-236-237-238-239-240-241-242-243-244-245-246-247-248-249-250-252-253-254-255-258-0

	FTN	850.80	0-45-55-66-78-91-105-120-136-153-171-190-210-231-253-274-294-312-293-273-252-230-209-189-170-152-135-119-104-90-77-65-54-44-137-0-75-62-74-87-101-116-132-149-167-186-206-227-249-271-292-311-291-270-248-226-205-185-166-148-131-115-100-86-73-61-0-34-43-53-76-89-103-118-134-151-169-188-208-229-251-272-250-228-207-187-168-150-133-117-102-88-64-63-52-42-269-0-123-56-67-79-92-106-121-154-172-191-211-232-254-275-313-295-276-255-233-212-192-173-155-138-122-107-93-80-68-57-0-97-84-98-113-129-146-164-183-203-224-246-268-289-309-310-290-247-225-204-184-165-147-130-114-99-85-72-0-82-95-109-124-140-157-175-194-214-235-257-278-297-315-314-296-277-256-234-213-193-174-156-139-108-94-81-69-0-110-96-111-127-144-162-181-201-222-244-266-287-307-308-288-267-245-223-202-182-163-145-128-112-83-0-126-142-159-177-196-216-237-259-280-299-317-316-298-279-258-236-215-195-176-158-141-125-160-143-161-179-198-0-243-221-242-263-283-262-241-220-200-180-199-219-240-218-239-261-282-301-319-318-300-281-260-238-217-197-178-0-37-47-58-70-264-285-305-306-286-265-304-284-303-321-302-320-322-71-59-48-38-0-46-10-15-21-28-36-35-27-20-14-9-5-0-12-18-25-33-51-41-32-24-17-11-0-23-31-40-50-60-49-39-30-22-16-0-4-13-26-19-8-7-6-29-0-1-2-3-0
in12	*Random*	5,917.86	0-140-10-198-368-67-365-238-206-22-228-292-213-136-355-344-53-375-277-21-113-103-49-146-43-259-48-0-74-68-167-235-196-237-153-114-20-23-254-343-92-255-363-275-329-229-360-77-315-6-242-13-241-115-44-0-145-372-142-340-12-250-72-280-265-152-102-117-286-374-221-300-233-132-122-90-157-281-134-339-347-313-0-239-149-217-332-337-39-69-328-79-207-251-319-351-105-354-66-125-57-33-120-130-317-8-41-327-81-356-15-0-230-128-86-119-263-293-248-17-166-104-147-89-172-116-283-87-370-256-282-369-246-224-84-278-187-0-50-204-244-176-192-290-291-61-97-214-367-201-205-158-297-37-133-264-56-100-40-220-144-274-29-27-0-154-185-216-11-267-42-118-373-35-222-63-211-188-285-208-359-309-210-75-173-51-0-212-165-179-321-80-59-169-203-108-312-135-7-126-311-371-258-99-18-139-353-160-318-302-199-331-0-324-288-45-111-64-314-177-175-174-247-83-287-62-215-121-268-24-195-295-362-330-296-225-123-131-109-107-0-304-155-52-326-55-106-240-357-323-226-276-227-85-336-94-219-231-298-325-30-161-334-150-168-19-342-0-129-163-180-28-95-14-4-65-305-93-253-349-171-346-127-301-32-335-307-156-36-243-9-71-310-112-209-0-1-352-143-322-333-184-249-190-162-218-98-137-151-338-194-124-2-31-232-138-308-47-54-70-58-0-110-271-306-299-260-164-96-270-181-73-284-269-245-261-16-25-289-141-26-266-159-341-60-358-197-76-0-193-236-191-78-182-364-38-183-82-46-186-178-366-316-252-170-34-262-350-223-88-294-345-91-3-320-273-0-148-202-303-257-101-279-5-200-189-348-272-361-234-0
	NN	2,847.68	0-65-313-315-317-319-375-374-367-318-373-372-366-316-314-369-365-371-370-312-368-363-359-356-362-358-364-360-357-353-361-355-351-348-354-350-349-347-0-130-344-352-346-342-339-345-341-338-343-340-337-335-332-336-333-330-327-334-331-328-326-1-325-323-2-329-3-4-5-320-6-321-7-8-0-195-324-9-322-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-0-260-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-66-67-68-69-70-76-0-129-71-72-73-74-75-77-78-79-80-81-82-83-84-85-86-87-88-89-90-91-92-93-94-95-96-97-0-196-98-99-100-101-102-103-104-105-106-107-108-109-110-111-112-113-114-115-116-117-118-119-120-121-0-259-122-123-124-125-126-127-128-131-132-133-134-135-136-137-138-139-140-141-142-143-144-0-261-145-146-147-148-149-150-151-152-153-154-155-156-157-158-159-160-161-162-163-168-0-194-164-165-166-167-169-170-171-172-173-174-175-176-177-178-179-180-181-182-183-184-0-206-185-186-187-188-189-190-191-192-193-197-198-199-200-201-0-207-202-203-204-205-208-209-210-211-212-213-214-215-216-217-218-219-220-221-242-0-271-222-223-224-225-226-227-228-229-230-231-232-233-234-235-236-237-238-239-240-241-246-0-272-243-244-245-247-248-249-250-251-252-253-254-255-256-257-258-262-263-264-265-266-267-275-0-270-268-269-273-274-276-277-278-279-280-281-282-283-284-285-286-287-288-289-290-291-292-293-294-295-0-296-297-298-299-300-301-302-303-304-305-306-307-308-309-310-311-0
	FTN	1,194.05	0-311-322-334-321-333-344-332-331-343-356-357-358-359-360-348-361-373-372-371-370-369-368-367-366-355-342-354-341-353-340-352-351-350-338-0-234-245-256-267-279-291-304-316-329-365-364-362-375-374-363-339-327-314-302-290-278-266-255-244-225-0-216-224-242-233-253-243-265-254-277-289-300-313-325-349-337-324-312-323-335-346-345-320-308-295-283-270-0-206-213-229-221-239-250-262-274-287-299-347-336-301-276-264-252-232-241-223-209-215-203-198-202-0-204-218-210-226-236-246-258-269-268-281-293-306-318-330-317-305-292-280-257-235-217-211-199-310-0-183-195-205-219-227-247-259-282-328-315-326-303-294-307-319-296-284-271-260-248-237-228-212-200-192-175-0-182-177-194-185-208-222-231-263-288-275-251-240-230-214-197-207-187-191-181-0-170-180-196-190-186-220-249-272-285-297-298-286-273-261-238-201-188-189-193-176-184-164-0-49-61-73-85-97-109-120-131-141-159-150-166-173-309-178-172-158-140-130-119-108-96-84-72-60-48-0-32-43-31-42-30-41-29-40-28-39-51-63-75-87-99-111-122-133-142-152-160-151-167-132-121-110-98-86-74-62-50-0-25-26-15-27-16-17-52-53-65-54-78-55-66-77-89-101-143-123-112-100-88-76-64-18-19-20-8-7-6-5-4-3-0-46-58-70-82-94-106-117-128-147-138-148-157-171-165-149-139-129-118-107-95-83-71-59-47-0-92-104-115-135-126-145-154-168-162-174-179-169-155-137-146-127-136-116-105-93-81-0-57-68-80-91-103-124-134-144-161-163-156-153-125-114-113-102-90-79-67-56-0-1-12-13-14-9-38-37-36-24-35-23-34-22-33-21-44-45-69-11-10-2-0

in13	*Random*	9,710.07	0-47-88-307-371-302-419-344-87-404-267-0-67-119-112-331-116-68-338-95-103-92-192-171-12-0-82-316-250-317-322-285-175-138-78-164-252-50-132-0-166-355-109-120-218-388-291-243-148-15-77-30-0-64-176-313-174-85-73-107-318-314-72-26-259-151-0-145-220-10-121-191-140-310-224-0-81-190-412-122-333-214-142-347-304-0-55-186-365-370-89-104-111-372-417-294-0-262-309-74-153-374-83-117-152-1-335-53-54-179-0-136-375-76-160-311-300-231-367-43-395-0-364-48-225-277-260-249-115-386-126-0-384-382-235-35-345-187-383-312-332-274-36-208-319-265-0-134-401-52-320-216-198-315-329-221-61-199-324-59-195-0-233-373-162-261-411-196-352-197-271-62-183-137-0-20-129-350-46-376-71-184-90-21-354-0-178-323-19-236-100-394-124-155-139-7-379-0-65-125-405-27-269-282-172-38-292-272-13-94-0-34-286-135-402-130-325-287-8-254-194-0-293-240-359-228-211-299-239-57-409-105-305-385-393-0-91-128-185-102-24-63-280-328-381-0-275-230-163-75-5-227-321-283-264-346-281-110-392-0-308-200-363-156-337-380-415-23-39-330-79-0-410-157-146-342-181-177-33-193-295-270-296-158-215-0-101-32-414-150-144-348-44-49-205-98-106-37-266-133-413-29-168-290-0-96-97-86-289-31-169-22-369-25-159-17-173-223-14-0-396-202-398-180-298-306-210-226-66-201-0-336-131-326-229-327-273-246-377-301-51-0-276-257-339-389-70-189-69-141-255-0-340-251-253-217-356-353-341-2-242-418-0-42-387-165-207-16-407-113-114-170-343-123-99-93-0-18-40-278-238-358-127-362-203-351-256-0-400-204-244-247-397-213-108-147-245-391-0-161-334-80-154-303-406-399-284-222-206-56-403-0-390-416-288-279-361-237-268-378-258-209-11-0-368-263-143-219-60-41-234-45-149-349-232-0-241-420-366-360-3-0-9-6-58-248-4-167-0-28-297-212-182-357-84-408-188-118-0
	NN	6,528.46	0-65-395-393-360-379-391-363-359-394-0-130-400-376-382-392-368-381-411-399-371-0-195-415-373-380-416-369-378-390-387-419-357-413-377-408-0-260-410-375-414-396-362-418-361-409-374-406-0-325-358-383-420-372-403-388-417-365-398-389-412-0-84-386-405-370-401-367-407-402-384-404-397-0-300-364-366-385-1-12-13-14-0-156-2-3-4-5-15-16-0-221-6-7-8-9-17-18-0-228-10-11-19-20-21-22-23-24-25-26-27-0-77-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-0-293-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-59-0-149-58-60-61-62-63-72-0-161-64-66-67-68-73-74-76-0-89-69-70-71-75-78-79-80-81-0-305-82-83-85-86-87-88-90-91-92-93-94-95-0-190-96-97-98-99-100-101-102-103-104-105-106-107-108-0-158-109-110-111-112-113-114-115-116-117-118-119-120-132-0-125-121-122-123-124-0-230-126-127-128-129-133-134-0-233-131-135-136-137-138-139-140-141-142-143-144-145-147-0-255-146-148-150-151-152-153-154-155-157-159-160-162-163-0-302-164-165-166-167-168-169-170-171-172-173-174-175-176-0-306-177-178-179-180-181-182-0-231-183-184-185-186-192-193-194-0-234-187-188-189-191-196-197-199-0-303-198-200-201-202-203-204-205-206-207-208-209-210-212-0-307-211-213-214-215-216-217-218-219-220-222-223-224-0-232-225-226-227-229-235-236-237-238-239-240-252-253-254-0-304-241-242-243-244-0-301-245-246-247-248-0-292-249-250-251-256-257-258-259-261-262-0-311-263-264-265-266-267-268-269-270-271-272-273-274-0-335-275-276-277-278-279-280-281-282-283-284-285-286-287-288-289-291-0-298-290-294-295-296-297-299-308-309-312-313-0-334-310-314-315-316-317-318-319-320-321-322-323-324-0-336-326-327-328-329-330-331-332-333-337-338-339-340-341-342-343-0-356-344-345-346-347-348-349-350-351-352-353-354-355-0
	FTN	2,013.91	0-315-316-375-376-371-419-311-359-251-0-250-351-310-416-370-411-410-409-350-0-247-327-307-392-367-387-386-385-326-0-342-343-402-403-369-408-309-348-249-0-339-340-399-400-368-395-308-335-248-0-318-319-378-379-366-384-306-324-246-0-196-255-256-191-299-358-418-417-412-357-298-239-179-0-223-282-283-189-288-347-407-406-397-346-287-228-168-0-220-279-280-188-275-334-394-393-388-333-274-215-155-0-199-258-259-186-264-323-383-382-373-322-263-204-144-0-236-295-354-405-414-415-356-355-296-190-291-290-231-0-212-271-330-381-390-391-332-331-272-187-267-266-207-0-103-163-222-281-336-341-401-396-363-389-338-329-278-219-160-0-213-268-328-273-214-128-398-303-243-269-0-79-139-198-257-312-317-377-372-360-413-314-353-254-195-136-0-237-292-352-297-238-131-374-300-240-252-0-285-245-345-365-305-294-235-176-171-111-0-226-227-286-337-304-364-404-349-344-289-230-229-0-261-242-321-362-302-270-211-152-147-87-0-202-203-262-313-301-361-380-325-320-265-206-205-0-164-224-169-170-130-70-116-10-51-56-0-140-200-145-146-127-67-92-7-27-32-0-209-183-276-221-162-129-69-108-167-0-133-193-142-143-126-66-84-6-19-24-0-157-124-217-284-244-277-184-166-107-48-0-85-86-260-241-253-181-121-82-23-0-78-137-192-180-293-194-135-76-11-119-178-0-99-158-218-159-100-8-95-154-153-208-182-0-12-72-60-132-120-197-138-77-18-17-22-73-0-75-16-59-118-177-232-185-225-234-175-174-115-114-0-62-141-122-201-210-151-150-91-90-31-30-0-20-13-420-113-173-233-71-134-74-15-14-53-52-0-101-156-123-149-98-39-40-68-94-35-34-0-36-96-41-42-102-216-161-43-9-47-106-97-46-37-0-1-83-61-80-25-26-81-2-28-0-54-55-44-49-50-110-109-104-64-4-45-0-5-105-65-165-125-172-117-58-57-112-0-21-88-148-93-33-38-89-63-3-29-0
in14	*Random*	8,191.31	0-152-250-394-29-195-286-337-384-240-107-364-269-164-361-50-328-422-218-16-116-43-274-205-144-268-81-0-66-414-183-448-321-194-70-24-11-396-142-154-232-14-63-147-156-132-53-455-174-0-151-169-106-139-189-34-330-477-23-451-465-297-406-313-311-27-217-68-349-211-176-292-378-206-95-180-150-471-316-228-0-238-246-351-358-346-173-375-383-166-127-90-120-65-433-210-143-264-432-276-392-257-412-67-439-254-266-357-227-6-0-109-153-333-248-170-400-380-453-134-179-306-100-401-69-79-245-343-162-200-249-221-204-247-10-98-33-0-30-415-424-483-96-457-352-149-7-391-416-78-411-425-201-452-447-470-0-363-273-129-141-251-435-427-397-468-369-259-202-475-121-155-284-271-20-256-327-3-223-62-476-0-288-181-188-54-399-289-198-214-213-417-472-367-118-117-404-140-163-408-178-293-342-124-407-320-434-413-203-454-381-92-389-309-91-326-0-308-244-393-285-61-130-131-71-32-182-239-441-302-356-372-379-171-28-319-87-480-159-18-76-469-0-461-371-111-72-421-243-22-209-362-122-446-136-296-340-237-339-462-229-287-19-398-158-267-230-386-191-290-49-0-220-437-115-37-431-325-350-38-482-126-88-466-263-52-208-279-368-224-85-295-344-15-215-44-312-145-0-47-46-317-458-226-382-148-55-260-84-359-137-436-114-135-442-298-59-336-305-56-377-64-459-329-318-133-93-0-444-449-419-463-341-438-168-105-460-9-110-48-103-265-360-234-39-304-467-13-2-0-172-99-94-299-335-473-80-222-51-258-428-236-426-353-17-301-262-354-235-403-418-253-294-322-82-479-278-272-252-0-324-177-104-219-4-167-40-197-282-303-225-376-199-347-119-366-275-192-430-112-231-314-12-0-283-388-348-338-478-307-281-334-108-36-31-102-60-374-373-196-160-280-77-443-387-355-83-445-291-450-410-185-0-385-21-1-315-45-456-323-125-193-216-300-409-146-332-74-270-186-474-25-395-0-5-440-310-41-57-187-75-190-86-241-128-175-390-26-89-207-261-464-35-113-138-157-42-0-73-331-212-420-345-277-429-184-255-365-423-402-161-97-165-405-481-242-58-101-8-370-233-123-0

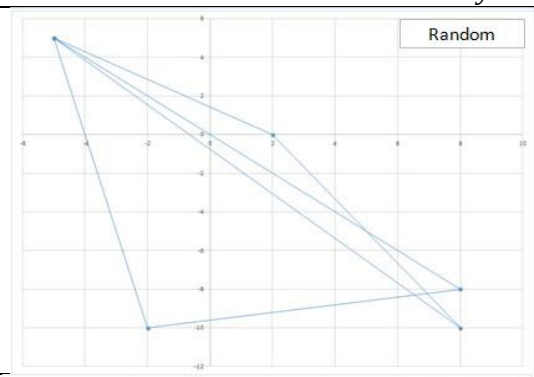
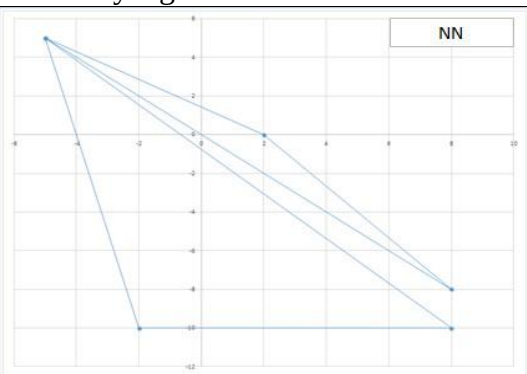
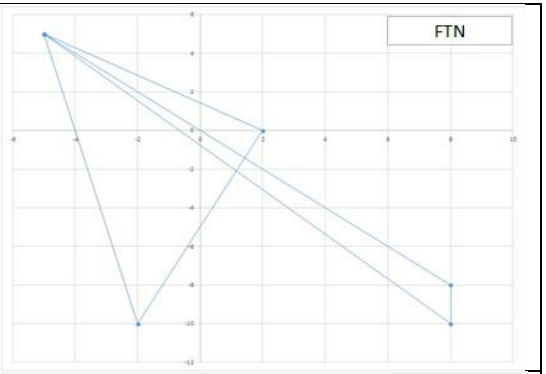
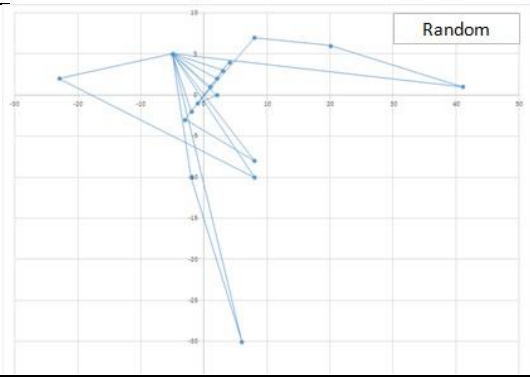
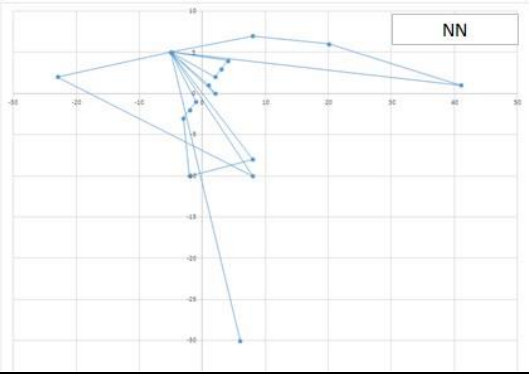
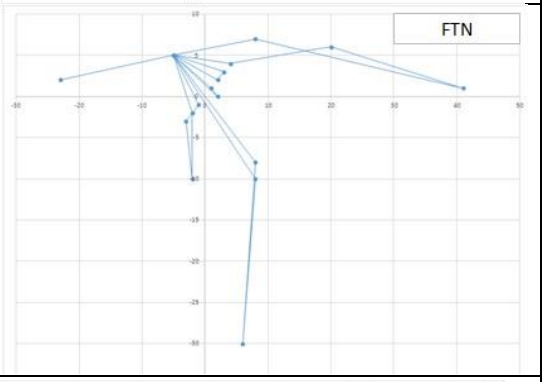
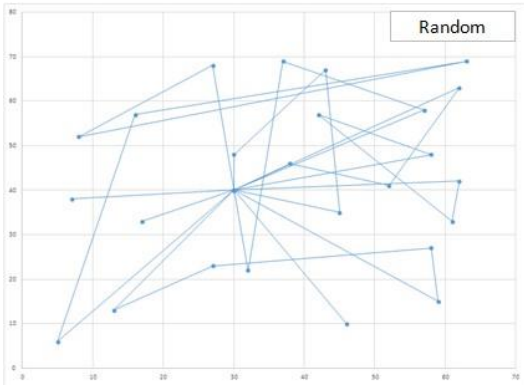
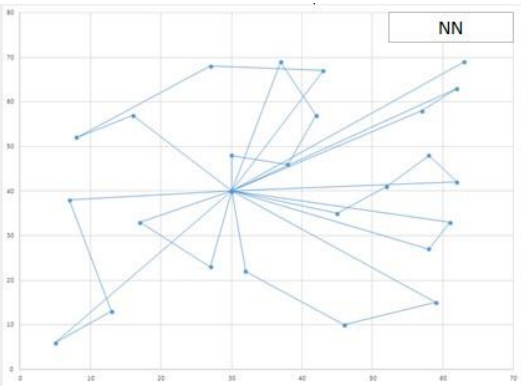
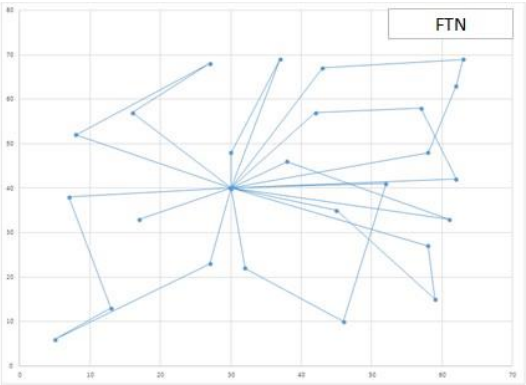
	NN	3,884.63	0-65-443-441-439-446-444-442-440-447-445-433-438-434-430-436-432-428-435-431-437-424-429-425-421-427-423-420-426-422-483-450-0-130-475-1-456-477-454-481-448-476-451-478-462-479-453-480-452-482-449-469-2-457-470-20-0-195-474-3-458-471-4-459-472-5-460-473-6-461-0-260-467-7-468-8-463-9-464-10-465-11-466-13-455-0-325-12-14-15-16-17-18-19-21-22-44-0-390-23-24-25-26-27-28-29-30-31-32-33-76-0-64-34-35-36-37-38-39-40-41-42-43-45-46-47-48-0-67-49-50-51-52-53-54-55-56-57-58-59-60-61-62-63-66-189-0-68-69-70-71-72-73-74-75-77-78-79-80-81-82-83-84-85-86-87-0-133-88-89-90-91-92-93-94-95-96-97-98-99-100-101-102-103-104-105-106-107-108-109-113-0-134-110-111-112-114-115-116-117-118-119-120-121-122-123-124-125-126-127-128-129-131-132-135-136-137-230-0-193-138-139-140-141-142-143-144-145-146-147-148-149-150-151-152-153-154-155-156-157-158-159-160-161-162-163-0-198-164-165-166-167-168-169-170-171-172-173-174-175-176-177-178-179-180-181-182-183-184-185-186-187-188-190-191-192-194-196-0-199-197-200-201-202-203-204-205-206-207-208-209-210-211-212-213-214-215-216-217-218-219-220-221-222-223-224-225-226-227-228-229-231-0-257-232-233-234-235-236-237-238-239-240-241-242-243-244-245-246-247-248-249-250-251-252-253-254-255-256-258-259-261-262-263-264-265-266-271-0-322-267-268-269-270-272-273-274-275-276-277-278-279-280-281-282-283-284-285-286-287-288-289-290-291-292-293-294-295-296-297-298-299-300-301-302-303-0-329-304-305-306-307-308-309-310-311-312-313-314-315-316-317-318-319-320-321-323-324-326-327-328-330-331-332-333-334-335-336-337-338-339-340-341-342-343-345-0-386-344-346-347-348-349-350-351-352-353-354-355-356-357-358-359-360-361-362-363-364-365-366-367-368-369-370-371-372-373-374-375-376-377-378-379-380-381-382-383-391-0-387-384-385-388-389-392-393-394-395-396-397-398-399-400-401-402-403-404-405-406-407-408-409-410-411-412-413-414-415-416-417-418-419-0
	FTN	1,193.84	0-190-210-231-253-274-294-313-331-348-364-379-393-406-418-429-439-448-456-463-469-474-478-481-482-480-477-473-468-462-455-447-438-428-417-405-392-378-363-347-330-312-293-273-252-230-209-189-170-0-212-233-255-276-296-315-333-350-366-381-395-408-420-431-441-450-458-465-471-476-479-475-470-464-457-449-440-430-419-407-394-380-365-349-332-314-295-275-254-232-211-191-0-214-235-257-278-298-317-335-352-368-383-397-410-422-433-443-452-460-467-472-466-459-451-442-432-421-409-396-382-367-351-334-316-297-277-256-234-213-192-172-0-153-216-237-259-280-300-319-337-354-370-385-399-412-424-435-445-454-461-453-444-434-423-411-398-384-369-353-336-318-299-279-258-236-215-194-174-0-177-197-218-239-261-282-302-321-339-356-372-387-401-414-426-437-446-436-425-413-400-386-371-355-338-320-301-281-260-238-217-196-176-0-160-179-199-219-241-263-284-304-323-341-358-374-389-403-416-427-415-402-388-373-357-340-322-303-283-262-240-198-178-159-158-0-127-144-162-181-200-221-243-265-286-306-325-343-360-376-391-404-390-375-359-342-324-305-285-264-242-220-180-161-143-126-0-98-113-129-146-164-182-202-223-245-267-288-308-327-345-362-377-361-344-326-307-287-266-244-222-201-163-145-128-112-97-0-73-86-100-115-131-148-165-184-204-225-247-269-290-310-329-346-328-309-289-268-246-224-203-183-147-130-114-99-85-72-0-52-63-75-88-102-117-133-149-167-186-206-227-249-271-292-311-291-270-248-226-205-185-166-132-116-101-87-74-62-51-0-26-34-53-76-103-134-151-169-188-208-229-251-272-250-228-207-187-168-150-118-89-64-43-42-33-25-0-67-28-36-45-55-66-78-91-105-120-136-135-171-152-119-104-90-77-65-54-44-35-27-20-0-38-47-57-68-80-93-107-122-138-155-193-173-154-137-121-106-92-79-56-46-37-69-0-49-59-70-82-95-109-124-140-157-195-175-156-139-123-108-94-81-58-48-0-31-40-50-60-71-83-111-110-142-141-125-96-39-30-0-11-17-24-41-84-61-32-18-12-19-0-6-10-15-22-29-21-14-9-5-0-3-7-23-16-2-0-4-13-8-1-0-483-0
in15	*Random*	12,729.46	0-33-67-331-3-409-75-93-125-45-117-210-16-7-255-392-113-477-573-148-228-128-482-66-475-175-261-179-533-196-523-20-166-367-10-336-74-153-296-312-514-142-571-136-0-418-354-355-185-205-394-56-406-395-50-539-347-447-565-527-315-444-177-58-341-370-44-151-432-27-448-483-92-288-184-214-282-369-240-254-237-513-0-500-187-264-158-566-144-22-520-398-169-391-161-154-112-222-15-316-231-350-467-247-489-382-121-13-103-470-306-504-368-364-534-375-446-152-436-453-122-530-0-576-338-518-104-203-171-276-55-510-275-251-547-574-333-419-468-332-569-233-281-147-435-457-431-463-218-310-290-311-135-285-532-359-461-597-327-42-266-324-18-321-0-244-270-293-193-47-70-90-118-256-490-127-12-190-73-217-4-197-372-302-120-280-263-422-65-202-377-81-30-39-531-371-334-536-239-299-525-21-388-0-79-498-5-32-146-400-235-295-384-379-199-271-245-438-164-8-246-43-452-445-195-322-138-450-139-449-269-351-578-268-227-352-357-551-85-508-249-124-294-421-38-588-528-173-0-87-72-292-116-381-583-89-108-78-19-284-131-167-480-339-549-485-561-277-402-587-37-430-455-411-380-415-537-64-397-420-225-401-374-494-515-36-53-301-0-496-465-594-314-191-194-362-143-211-97-207-516-300-69-495-178-232-519-174-346-188-80-114-410-443-126-337-356-474-538-163-412-57-408-216-272-580-46-497-94-344-414-0-71-209-88-596-559-546-262-259-76-325-29-595-416-172-424-40-61-522-141-317-586-383-34-600-535-297-405-473-226-253-68-517-487-433-552-503-313-329-241-287-28-0-186-593-133-130-590-165-257-318-555-230-101-156-378-62-529-460-180-579-14-484-570-24-248-51-428-95-404-236-198-176-330-307-220-298-54-77-544-162-157-471-41-96-459-99-35-564-0-223-278-215-582-349-111-260-575-23-403-585-221-550-83-25-265-168-526-592-478-286-567-524-393-273-319-17-462-224-274-82-0-413-387-476-511-213-363-342-492-353-584-328-358-481-557-434-134-159-200-98-365-601-208-479-502-192-591-440-140-558-106-562-109-155-501-212-491-0-229-417-348-234-464-373-390-553-305-6-182-105-389-345-442-181-577-304-361-458-145-563-343-572-509-512-540-107-366-63-385-160-150-599-238-119-429-289-291-86-0-541-219-568-426-486-454-323-340-469-456-437-279-242-309-386-49-556-110-170-149-326-545-560-11-206-335-542-48-189-441-91-506-0-204-115-303-9-201-283-466-543-554-250-129-581-31-123-137-360-488-2-52-423-548-598-243-439-132-589-84-505-0-60-425-472-407-376-451-427-399-59-1-499-320-100-252-183-267-26-521-308-258-493-507-102-396-0

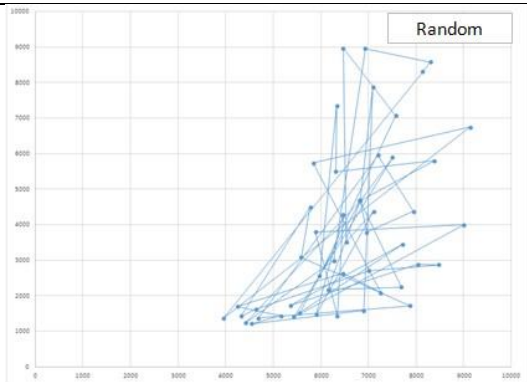
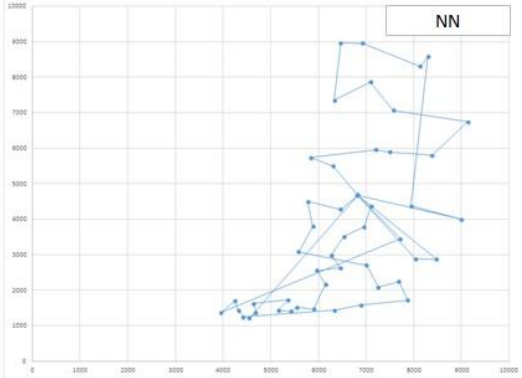
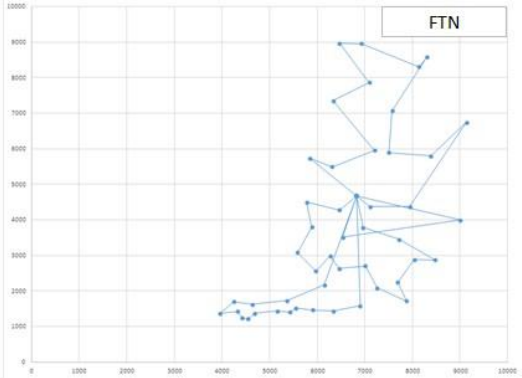
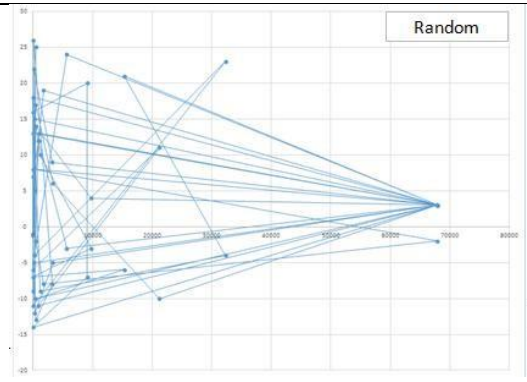
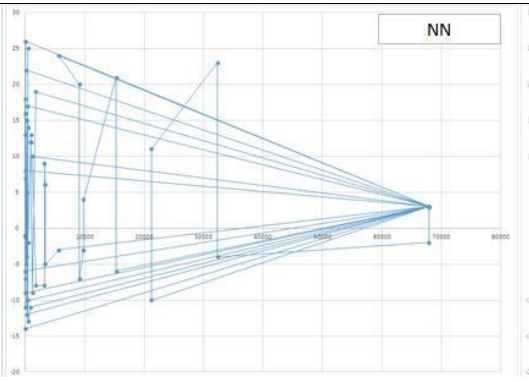
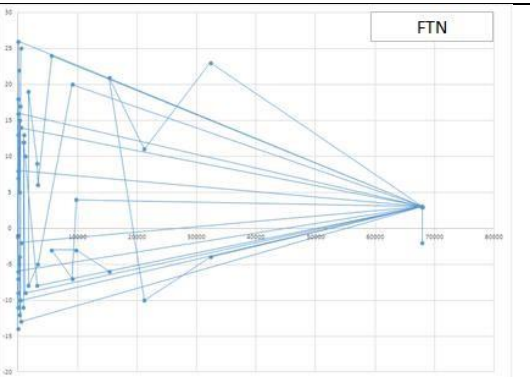
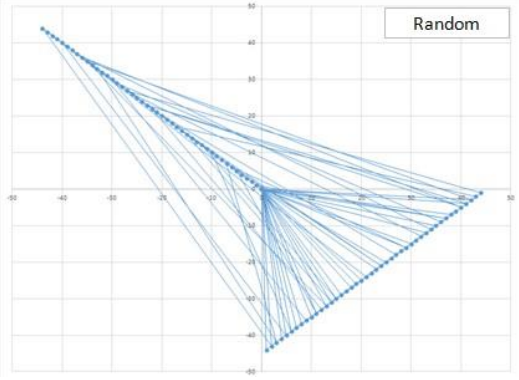
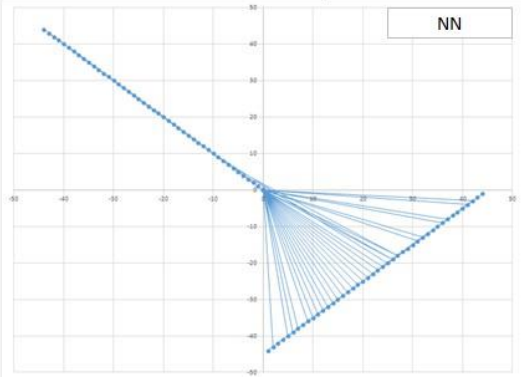
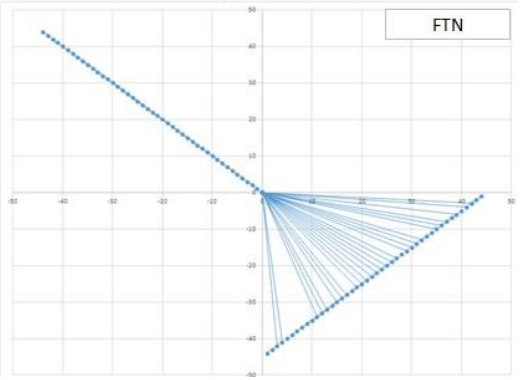
	NN	9,436.83	0-65-574-572-573-566-570-571-564-568-569-575-559-561-565-562-558-563-567-560-554-557-555-556-552-553-549-548-550-543-544-545-542-546-538-547-540-551-539-541-600-585-597-590-598-578-596-594-0-130-595-588-1-586-601-592-2-587-599-580-593-583-591-3-581-4-589-5-584-6-576-582-7-8-579-9-577-10-11-12-13-14-15-16-17-18-19-20-21-22-23-0-195-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-66-67-68-69-70-71-75-76-0-260-72-73-74-77-78-79-80-81-82-83-84-85-86-87-88-89-90-91-92-93-94-95-96-97-98-99-100-101-102-103-104-105-106-107-108-109-110-111-112-113-114-115-116-117-118-119-0-325-120-121-122-123-124-125-126-127-128-129-131-132-133-134-135-136-137-138-139-140-141-142-143-144-145-146-147-148-149-150-151-152-153-154-155-156-157-158-159-160-161-162-163-164-165-166-167-170-0-390-168-169-171-172-173-174-175-176-177-178-179-180-181-182-183-184-185-186-187-188-189-190-191-192-193-194-196-197-198-199-200-201-202-203-204-205-207-224-0-455-206-208-209-210-211-212-213-214-215-216-217-218-219-220-221-222-223-225-226-227-228-229-230-231-232-233-234-235-236-237-238-239-255-0-520-240-241-242-243-244-245-246-247-248-249-250-251-252-253-254-256-257-258-259-0-332-261-262-263-264-265-266-267-268-269-270-271-272-273-274-275-276-277-278-279-280-281-282-283-284-285-286-287-0-397-288-289-290-291-292-293-294-295-296-297-298-299-300-301-302-303-304-305-306-307-308-309-310-311-312-313-314-315-316-317-318-320-0-514-319-321-322-323-324-326-327-328-329-330-331-333-334-335-336-337-338-339-340-341-342-343-344-345-346-347-348-349-350-351-352-353-354-355-356-357-358-0-457-359-360-361-362-363-364-365-366-367-368-369-370-371-372-373-374-375-376-377-378-379-380-381-382-383-384-385-386-387-388-389-391-392-394-0-516-393-395-396-398-399-400-401-402-403-404-405-406-407-408-409-410-411-412-413-414-415-416-417-418-419-420-421-422-423-424-425-426-427-428-429-430-0-527-431-432-433-434-435-436-437-438-439-440-441-442-443-444-445-446-447-448-449-450-451-452-453-454-456-458-459-460-461-462-463-464-465-466-467-468-469-471-0-524-470-472-473-474-475-476-477-478-479-480-481-482-483-484-485-486-487-488-489-490-491-492-493-494-495-496-497-498-499-500-501-502-503-504-505-506-507-508-509-510-519-0-526-511-512-513-515-517-518-521-522-523-525-528-529-530-531-532-533-534-535-536-537-0
	FTN	2,355.43	0-523-529-537-540-552-545-541-553-559-565-569-573-577-591-585-589-596-599-600-597-593-586-580-571-550-561-562-548-495-510-527-526-512-506-490-480-471-449-461-470-493-505-489-479-439-0-256-266-272-316-331-346-353-361-373-401-429-413-405-402-430-453-459-587-474-387-356-268-260-222-197-259-239-258-220-195-221-289-323-349-351-319-290-223-201-298-409-329-229-171-125-96-139-179-151-183-161-0-335-363-377-390-418-452-458-440-462-475-482-498-511-522-521-509-556-557-549-570-583-590-595-588-579-582-592-598-601-594-581-547-494-473-386-357-0-358-406-396-410-424-434-446-457-455-447-435-426-415-399-384-344-333-313-296-284-292-314-327-321-315-299-286-287-317-302-294-257-231-219-193-187-169-143-135-131-118-114-237-163-103-72-61-0-460-464-484-492-499-513-528-539-551-563-567-574-584-578-575-568-564-542-546-554-543-538-533-525-519-504-488-497-503-508-517-524-531-516-507-501-496-485-477-465-438-0-380-394-414-422-433-444-466-454-456-445-467-478-487-518-532-566-555-560-544-534-520-536-491-481-472-450-428-437-400-385-372-345-285-270-254-233-227-217-199-235-255-0-230-293-301-305-336-391-404-412-419-441-463-476-483-514-535-558-576-572-530-515-500-443-421-432-442-431-420-392-364-330-306-271-265-202-0-339-367-388-395-381-368-354-341-324-311-325-342-355-359-369-382-397-411-407-425-468-469-448-427-436-451-374-347-375-378-403-417-376-348-318-288-203-218-191-189-177-0-198-226-232-248-269-291-279-295-309-340-423-502-486-398-389-371-343-383-370-312-282-251-213-250-212-249-263-241-224-211-210-234-261-320-0-42-56-49-57-59-68-81-90-85-108-111-117-123-147-185-153-157-165-181-173-167-149-145-141-155-158-175-133-127-121-101-84-77-75-67-76-93-97-105-100-89-80-66-58-48-62-73-65-64-91-82-0-209-247-278-308-332-214-252-283-216-225-262-240-264-253-215-281-310-280-326-338-360-365-352-337-307-300-277-246-0-188-190-366-393-379-276-245-207-194-206-238-244-274-322-334-350-303-273-0-297-200-204-267-242-196-205-243-304-416-362-275-208-192-186-0-126-140-144-148-166-172-174-159-154-128-120-129-115-112-106-83-94-98-107-137-70-51-53-45-33-26-24-11-18-28-37-34-35-55-88-79-99-92-86-69-74-63-50-43-41-32-0-16-25-17-13-22-19-39-20-7-9-3-1-6-23-40-31-44-52-47-46-54-60-71-102-113-150-182-160-184-168-142-134-109-122-116-110-104-119-132-136-156-164-180-176-152-146-130-0-2-8-27-10-5-12-21-15-29-30-36-78-87-178-138-170-228-408-328-236-162-124-95-38-14-4-0

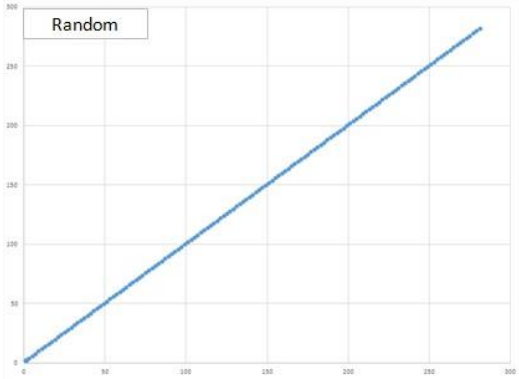
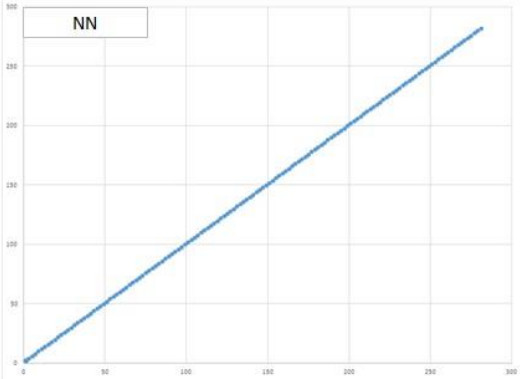
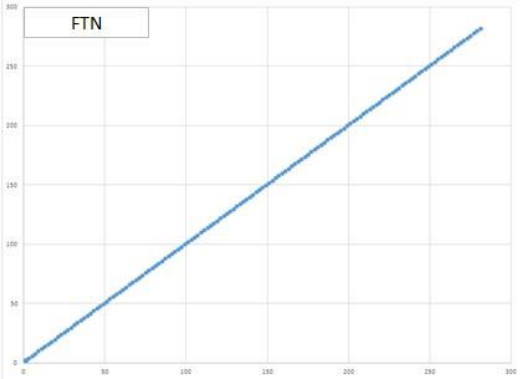
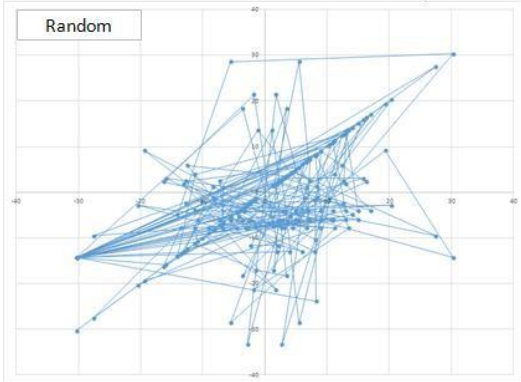
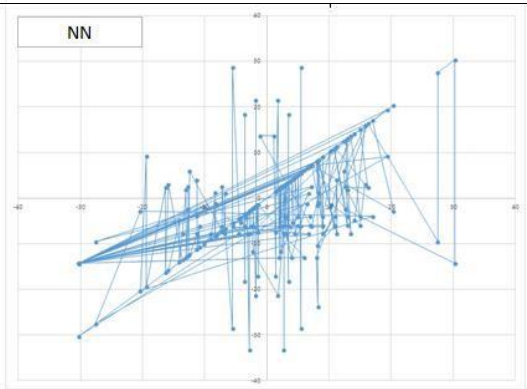
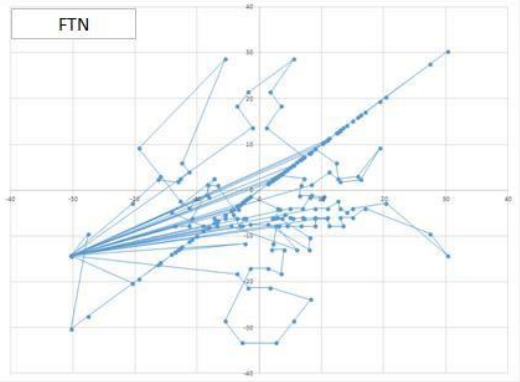
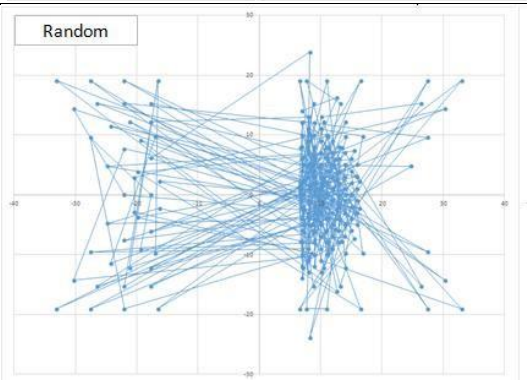
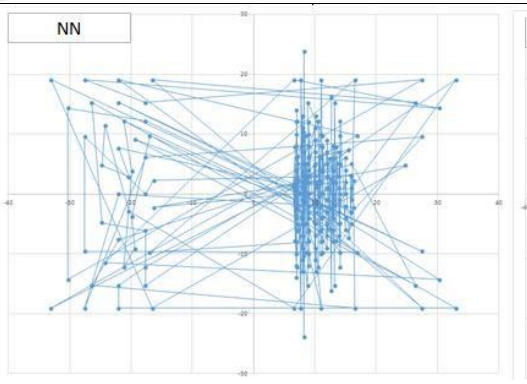
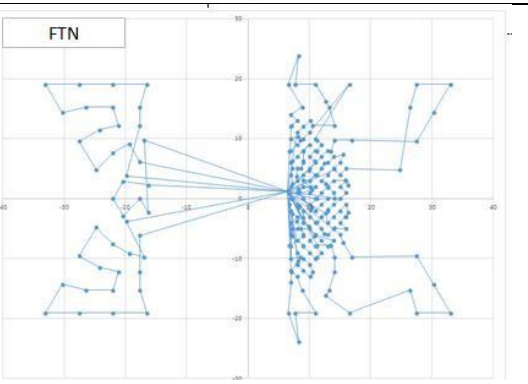
Dari segi total jarak tempuh, metode FTN memang memberikan hasil terbaik jika dibandingkan dengan metode NN ataupun metode *random*. Namun, apakah dari segi rute yang dihasilkan FTN memang benar-benar memberikan hasil yang lebih baik? Untuk memperlihatkan bahwa rute yang dibentuk oleh FTN memang lebih baik, maka penyusun memetakan hasil rute pada Tabel 1. kedalam *template excel* yang penyusun rancang sedemikian rupa hingga dapat menggambarkan grafik rute VRP. Hasil dari pemetaan rute kedalam grafik digambarkan dalam Tabel 2.

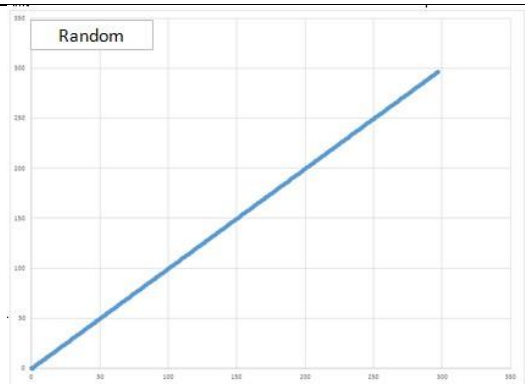
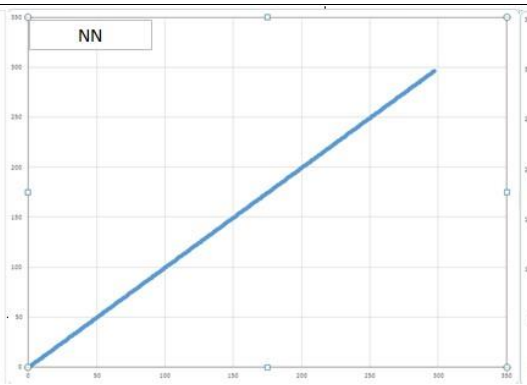
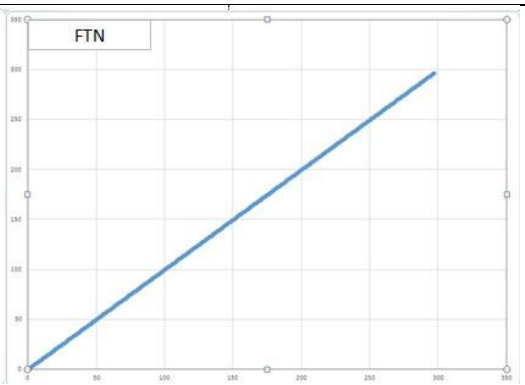
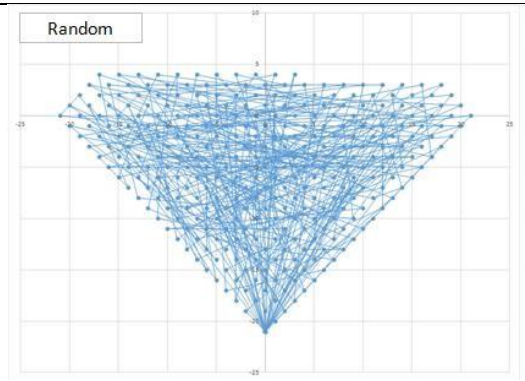
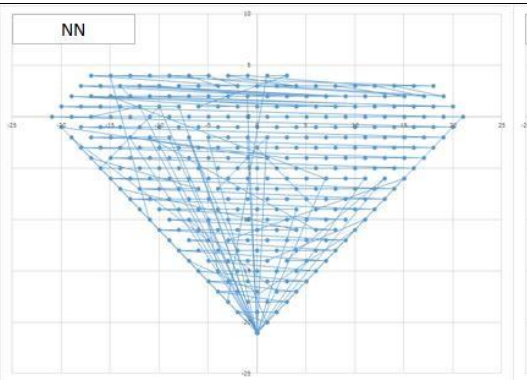
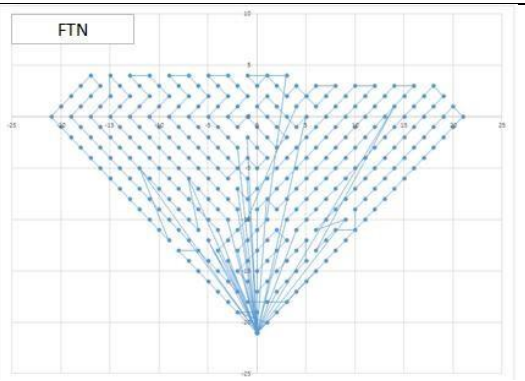
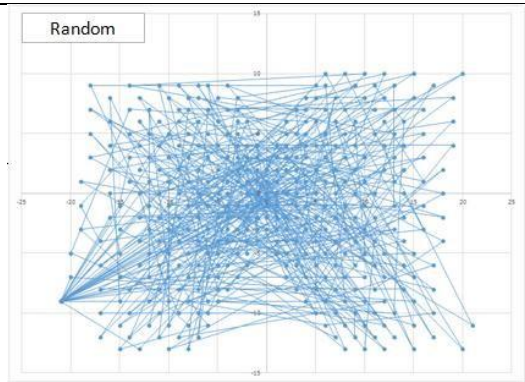
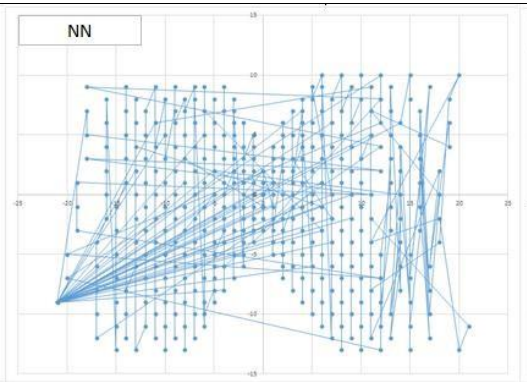
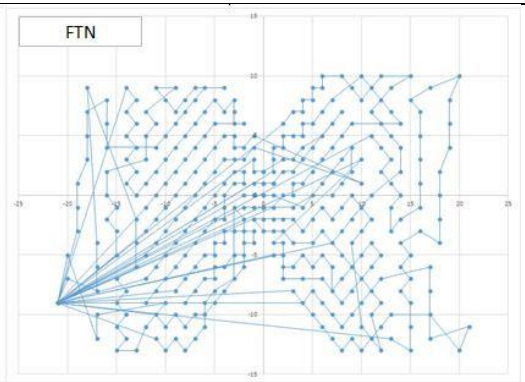
Menurut pengalaman dan pengamatan penyusun dalam hal rute VRP, semakin sebuah rute mendekati solusi optimum maka grafik yang dihasilkan semakin rapih dan berpola. Dari Tabel 2., FTN memiliki grafik yang lebih berpola dibandingkan metode *random* dan NN. Hal ini memperkuat pernyataan penyusun yang menyatakan bahwa, terlepas dari kekurangan metode FTN, FTN memang lebih baik jika dibandingkan dengan *random* dan NN.

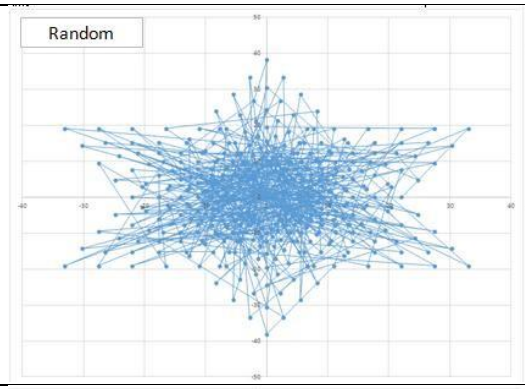
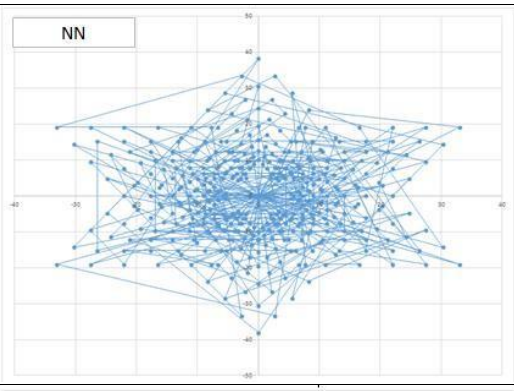
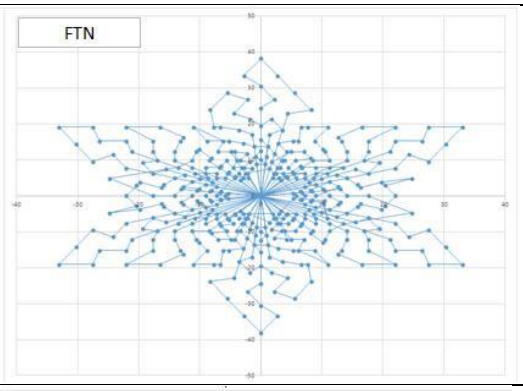
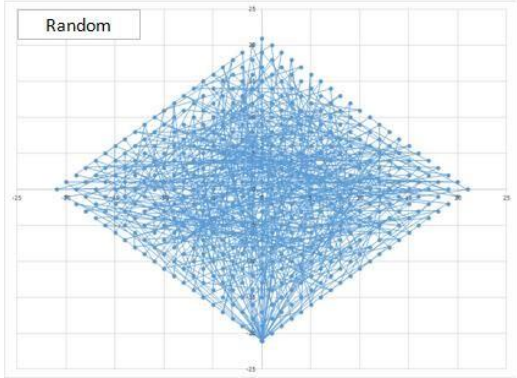
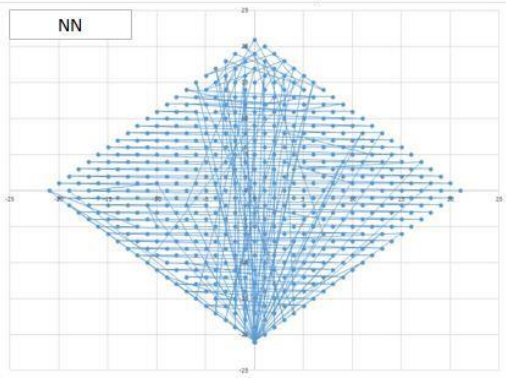
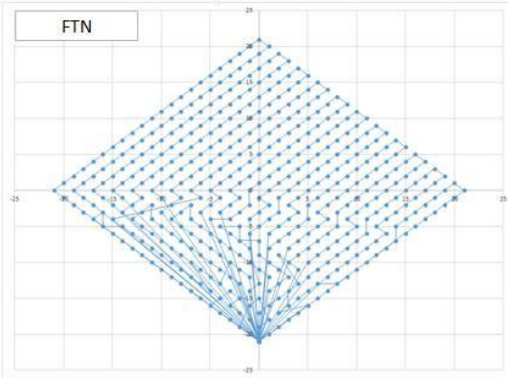
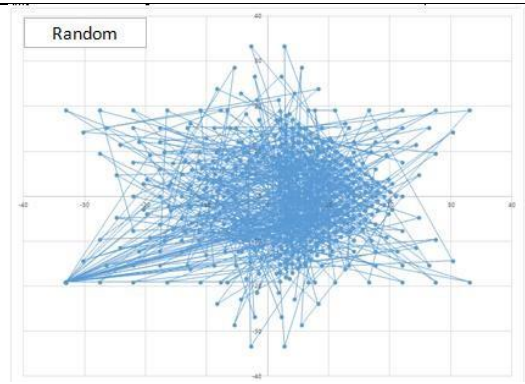
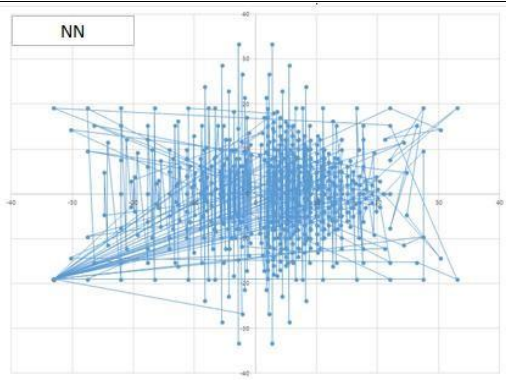
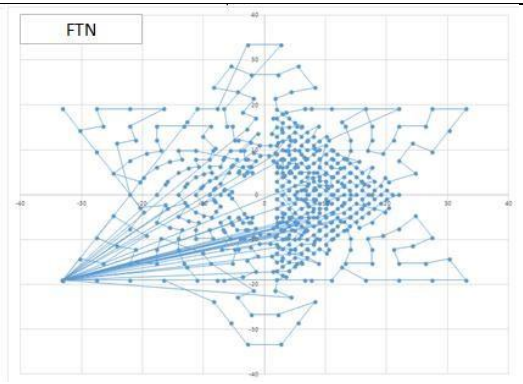
Tabel 2. Grafik rute *study cases* VRP yang dihasilkan oleh metode *random*, NN, serta FTN

In			
In2			
In3			

In4			
In5			
In6			

In7			
In8			
In9			

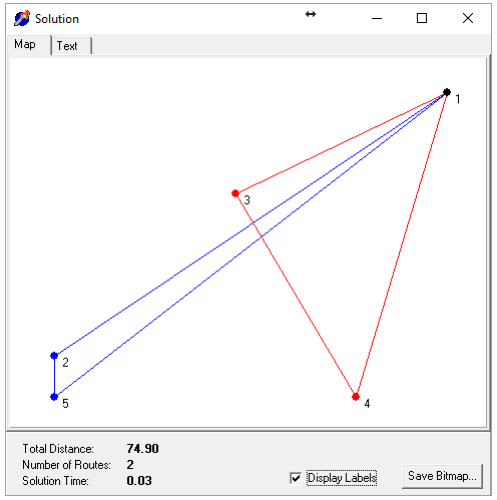
In10			
In11			
In12			

In13			
In14			
In15			

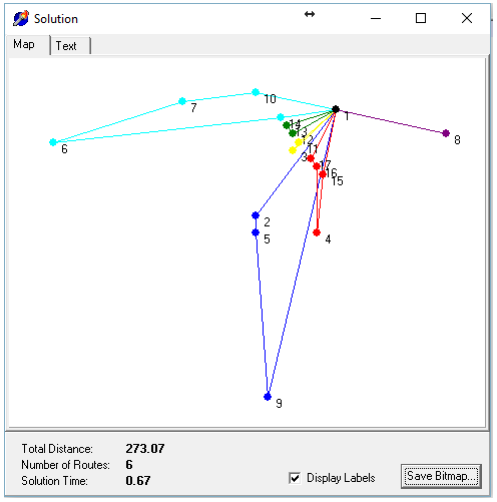
HASIL *RUNNING* VRP *SOLVER*

HASIL *RUNNING* VRP *SOLVER*

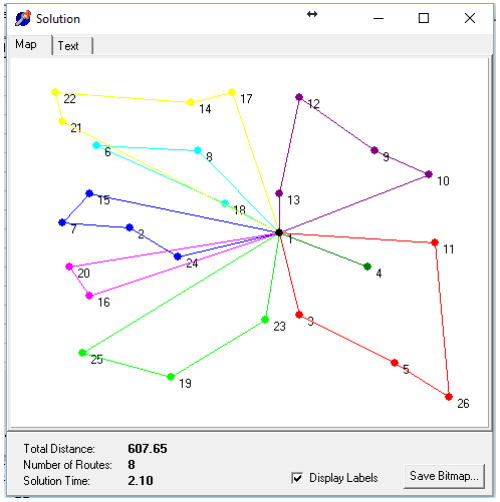
Berikut merupakan solusi penyelesaian kelima belas kasus tersebut menggunakan VRP *Solver*. Pada grafik yang dihasilkan VRP *Solver* tersebut, nilai x dianggap sebagai nilai latitude (sumbu y) dan nilai y dianggap sebagai longitude (sumbu x). Oleh karena itu, lokasi *nodes* pada grafik yang terbentuk melalui excel terlihat adanya sedikit berbeda dengan yang ditampilkan oleh VRP *Solver*. VRP *Solver* yang dikembangkan oleh Snyder (2003) dan rincian rute yang dihasilkan VRP *Solver* pada masing-masing kasus dilampirkan pada *soft copy*.



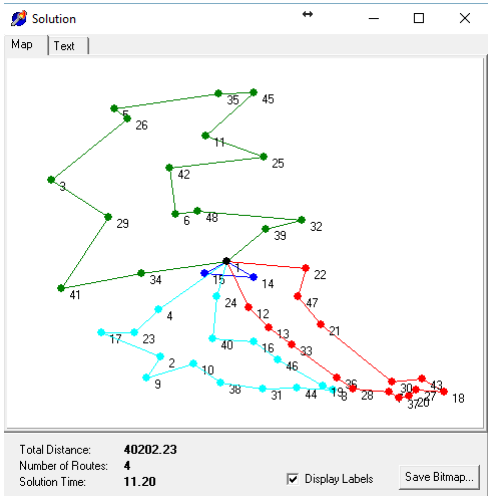
Gambar 1. Hasil *Running* In dengan VRP *Solver Snyder* (2003)



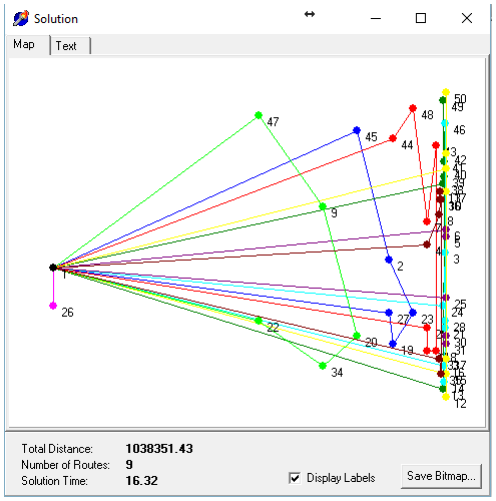
Gambar 2. Hasil *Running* In2 dengan VRP *Solver Snyder* (2003)



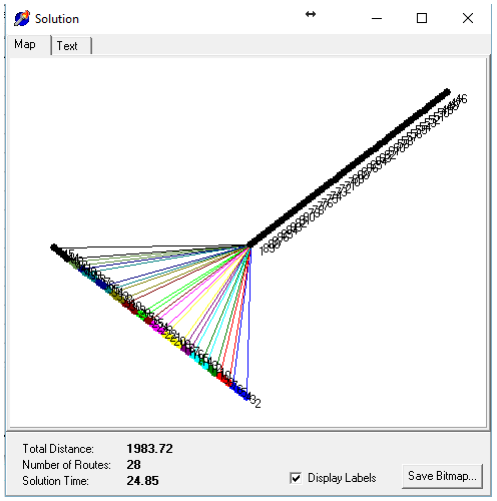
Gambar 3. Hasil *Running* In3 dengan VRP *Solver Snyder* (2003)



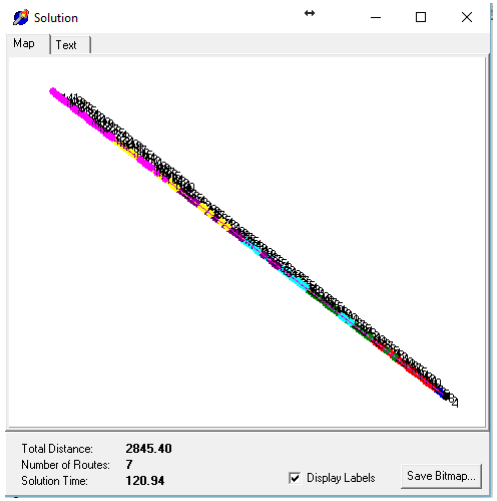
Gambar 4. Hasil *Running* In4 dengan VRP *Solver Snyder* (2003)



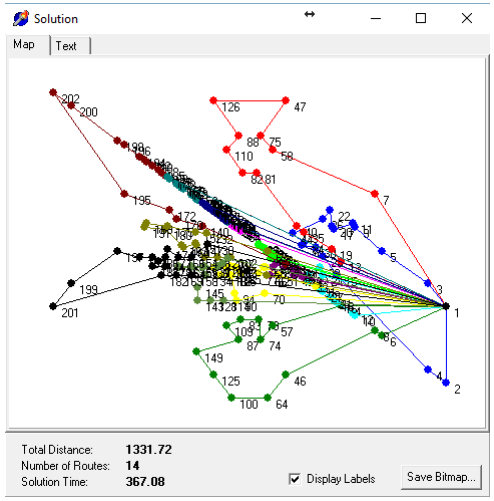
Gambar 5. Hasil *Running* In5 dengan VRP *Solver Snyder* (2003)



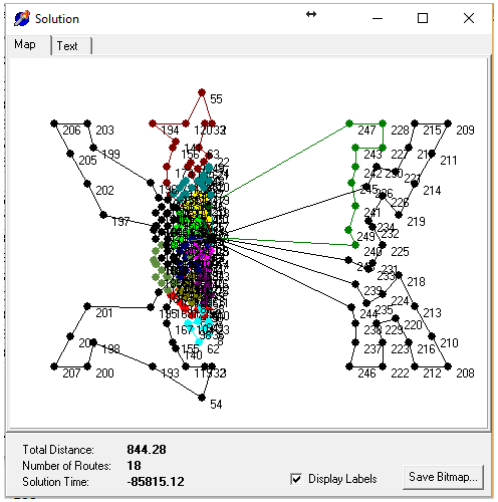
Gambar 6. Hasil *Running* In6 dengan VRP *Solver Snyder* (2003)



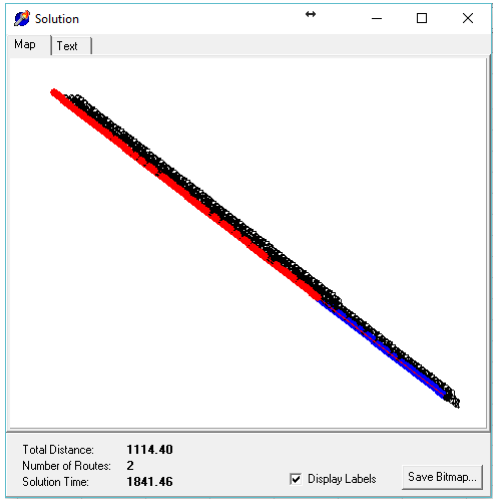
Gambar 7. Hasil *Running* In7 dengan VRP *Solver Snyder* (2003)



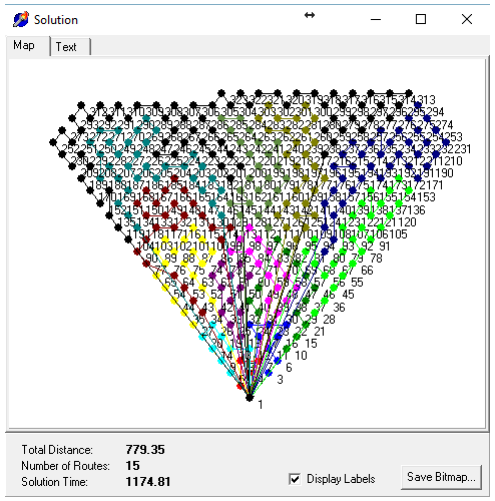
Gambar 8. Hasil *Running* In8 dengan VRP *Solver Snyder* (2003)



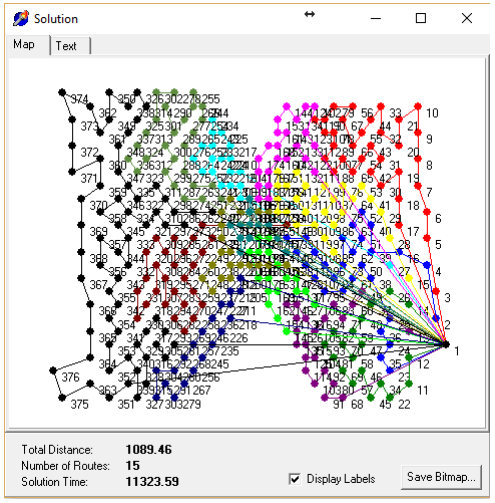
Gambar 9. Hasil *Running* In9 dengan VRP *Solver Snyder* (2003)



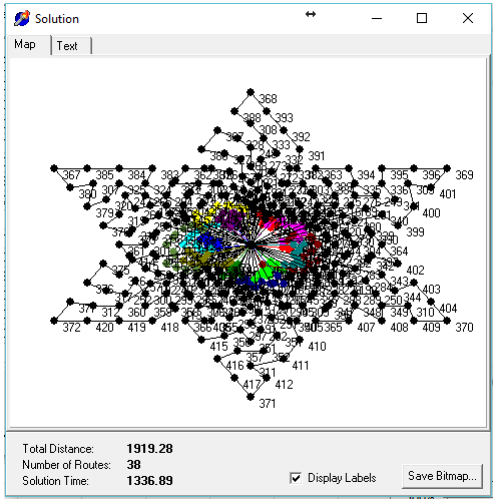
Gambar 10. Hasil *Running* In10 dengan VRP *Solver Snyder* (2003)



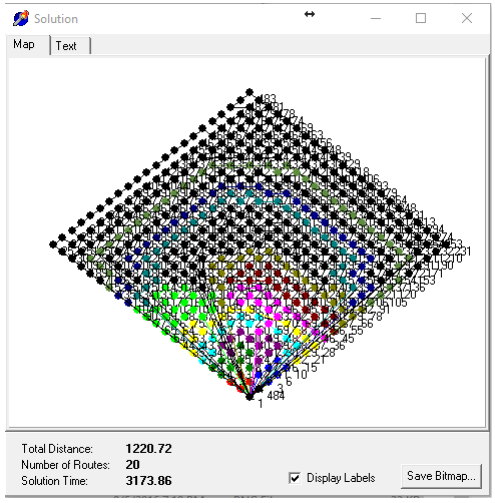
Gambar 11. Hasil *Running* In11 dengan VRP *Solver Snyder* (2003)



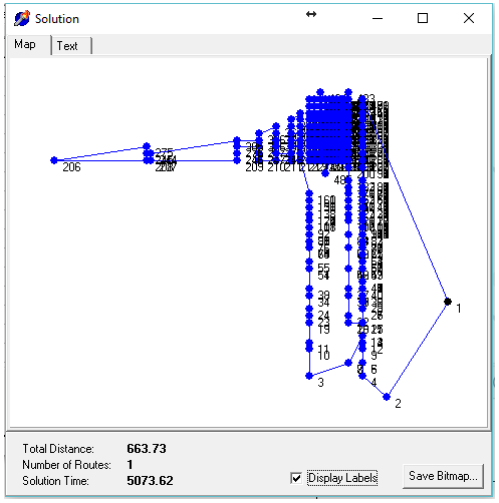
Gambar 12. Hasil *Running* In12 dengan VRP *Solver Snyder* (2003)



Gambar 13. Hasil *Running* In13 dengan VRP *Solver Snyder* (2003)



Gambar 14. Hasil *Running* In14 dengan VRP *Solver Snyder* (2003)



Gambar 15. Hasil *Running* In15 dengan VRP *Solver Snyder* (2003)

SOURCE CODE PROGRAM

SOURCE CODE PROGRAM

Bagian ini menampilkan source code program untuk metode *random*, NN, dan FTN menggunakan bahasa C++.

1. Source code metode *Random*

```
#include  <cstdio>
#include  <cmath>
#include  <time.h>
#include <cstdlib>

float x[1000], y[1000], jarak[1000][1000], temp[1000][1000];
int urut[1000][1000], rute[1000][1000], permintaan[1000],
kapasitas;
int jawaban[10000], pivot = 0;
float jaraktotal = 0;
int N, i, j, k = 1;
int l = 50, r = 50;
bool selesai = 1, flag[1000];

float hitung_jarak(int a, int b);
void urutkan();
int terjauh(int x);
int terdekat(int x);
bool cekflag();
int random();

int main() {
  FILE *fp;
  fp = fopen("hasilRandom.txt", "w+");
  srand(time(NULL));
  scanf("%d", &N);
  for (i = 0; i <= N; i++) {
    scanf("%f %f %d", &x[i], &y[i], &permintaan[i]);
    flag[i] = 0;
  }
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      jarak[i][j] = hitung_jarak(i, j);
      temp[i][j] = jarak[i][j];
      urut[i][j] = j;
      rute[i][j] = 0;
      printf("%f ", jarak[i][j]);
    }
    printf("\n");
  }
  urutkan();
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
    printf("%d ", urut[i][j]);
    }
    printf("\n");
  }
  do {
    l = 0;
    r = 0;
    rute[k][r] = 0;
    kapasitas = permintaan[0];
    while (1) {
      float tmp = random();
      if (tmp != -1) {
      rute[k][r+1] = tmp;
      flag[rute[k][r+1]] = 1;
        kapasitas -= permintaan[rute[k][r+1]];
        r++;
      } else break;
      for (i = l; i <= r; i++) {
        printf("%d ", rute[k][i]);
      }
      printf("\n");
    }
    for (i = l; i <= r; i++) {
      printf("%d ", rute[k][i]);
      jawaban[pivot] = rute[k][i];
      pivot++;
    }
    printf("\n");
    k++;
  } while (!cekflag());
  jawaban[pivot] = 0;
  printf("Hasil Random: ");
  for (i = 0; i <= pivot; i++) {
  printf("%d ", jawaban[i]);
    jaraktotal += jarak[jawaban[i]][jawaban[i+1]];
  }
  printf("\n%f\n", jaraktotal);
  fprintf(fp, ",Random\n");
  fprintf(fp, "Total Jarak,%f\n", jaraktotal);
  for (i = 0; i <= pivot; i++) {
    if (i == 0) fprintf(fp, "Rute");
    fprintf(fp, ",%d\n", jawaban[i]);
  }
  return 0;
}

bool cekflag() {
  bool selesai = 1;
  for (i = 1; i <= N; i++) {
    if (!flag[i]) selesai = 0;
  }
  return selesai;
}

int random() {
  bool flagrandom[1000];
  if (cekflag()) return -1;
  else {
    for (int i = 1; i <= N; i++) {
      flagrandom[i] = 0;
    }
    while (1) {
      int x = (rand() % N) + 1;
      flagrandom[x] = 1;
      if (!flag[x] && kapasitas-permintaan[x] >= 0)
        return x;
      bool selesai = -1;
      for (int i = 1; i <= N; i++) {
        if (!flagrandom[i]) selesai = 0;
      }
      if (selesai) return -1;
    }
  }
}

int terdekat(int x) {
  for (int i = 1; i <= N; i++) {
    if (!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >=
0 && urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

int terjauh(int x) {
  for (int i = N; i >= 1; i--) {
```

```
    if(!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >= 0
&& urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

float hitung_jarak(int a, int b) {
  return sqrt(pow((x[a]-x[b]), 2) + pow((y[a]-y[b]), 2));
}

void urutkan() {
  float tmp;
  int tmp2;
  for (int k = 0; k <= N; k++) {
    bool swapped = true;
    int j = 0;
    while (swapped) {
      swapped = false;
      j++;
      for (int i = 1; i <= N - j; i++) {
        if (temp[k][i] > temp[k][i+1]) {
          tmp = temp[k][i];
          temp[k][i] = temp[k][i+1];
          temp[k][i+1] = tmp;
          tmp2 = urut[k][i];
          urut[k][i] = urut[k][i+1];
          urut[k][i+1] = tmp2;
          swapped = true;
        }
      }
    }
  }
  return;
}
```

2. Source code metode *Nearest Neighbor* (NN)

```
#include <cstdio>
#include <cmath>

float x[1000], y[1000], jarak[1000][1000], temp[1000][0100];
int urut[1000][1000], rute[1000][1000], permintaan[1000],
kapasitas;
int jawaban[1000], pivot = 0;
float jaraktotal = 0;
int N, i, j, k = 1;
int l = 50, r = 50;
bool selesai = 1, flag[1000];

float hitung_jarak(int a, int b);
void urutkan();
int terjauh(int x);
int terdekat(int x);

int main() {
  FILE *fp;
  fp = fopen("hasilNN.txt", "w+");
  scanf("%d", &N);
  for (i = 0; i <= N; i++) {
    scanf("%f %f %d", &x[i], &y[i], &permintaan[i]);
    flag[i] = 0;
  }
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      jarak[i][j] = hitung_jarak(i, j);
      temp[i][j] = jarak[i][j];
      urut[i][j] = j;
      rute[i][j] = 0;
      printf("%f ", jarak[i][j]);
    }
    printf("\n");
  }
  urutkan();
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
    printf("%d ", urut[i][j]);
    }
    printf("\n");
  }
  do {
    l = 0;
    r = 0;
    rute[k][r] = 0;
    kapasitas = permintaan[0];
    while (1) {
      if (terdekat(rute[k][r]) != -1) {
        rute[k][r+1] = terdekat(rute[k][r]);
        flag[rute[k][r+1]] = 1;
        kapasitas -= permintaan[rute[k][r+1]];
        r++;
      } else break;
    for (i = l; i <= r; i++) {
      printf("%d ", rute[k][i]);
    }
    printf("\n");
  }
  for (i = l; i <= r; i++) {
    printf("%d ", rute[k][i]);
    jawaban[pivot] = rute[k][i];
    pivot++;
  }
  printf("\n");
  selesai = 1;
  for (i = 1; i <= N; i++) {
    if (!flag[i]) selesai = 0;
  }
  k++;
 } while (!selesai);
 jawaban[pivot] = 0;
 printf("Hasil NN: ");
 for (i = 0; i <= pivot; i++) {
   printf("%d ", jawaban[i]);
   jaraktotal += jarak[jawaban[i]][jawaban[i+1]];
 }
 printf("\n%f\n", jaraktotal);
 fprintf(fp, ",NN\n");
 fprintf(fp, "Total Jarak,%f\n", jaraktotal);
 for (i = 0; i <= pivot; i++) {
   if (i == 0) fprintf(fp, "Rute");
   fprintf(fp, ",%d\n", jawaban[i]);
 }
 return 0;
}

int terdekat(int x) {
  for (int i = 1; i <= N; i++) {
    if (!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >= 0
&& urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

int terjauh(int x) {
  for (int i = N; i >= 1; i--) {
    if(!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >= 0
&& urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

float hitung_jarak(int a, int b) {
  return sqrt(pow((x[a]-x[b]), 2) + pow((y[a]-y[b]), 2));
}

void urutkan() {
```

```
  float tmp;
  int tmp2;
  for (int k = 0; k <= N; k++) {
    bool swapped = true;
    int j = 0;
    while (swapped) {
      swapped = false;
      j++;
      for (int i = 1; i <= N - j; i++) {
        if (temp[k][i] > temp[k][i+1]) {
          tmp = temp[k][i];
          temp[k][i] = temp[k][i+1];
          temp[k][i+1] = tmp;
          tmp2 = urut[k][i];
          urut[k][i] = urut[k][i+1];
          urut[k][i+1] = tmp2;
          swapped = true;
        }
      }
    }
  }
  return;
}
```

3. Source code metode *Farthest to Nearest* (FTN)

Procedur ftn {prosedur tentang FTN tanpa memperhitungkan pesebaran *nodes*}:

```
#include <cstdio>
#include <cmath>

float x[1000], y[1000], jarak[1000][1000], temp[1000][1000];
int urut[1000][1000], rute[1000][1000], permintaan[1000],
kapasitas;
int jawaban[1000], pivot = 0;
float jaraktotal = 0;
int N, i, j, k = 1;
int l = 50, r = 50;
bool selesai = 1, flag[1000];

float hitung_jarak(int a, int b);
void urutkan();
int terjauh(int x);
int terdekat(int x);

int main() {
  scanf("%d", &N);
  for (i = 0; i <= N; i++) {
    scanf("%f %f %d", &x[i], &y[i], &permintaan[i]);
    flag[i] = 0;
  }
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      jarak[i][j] = hitung_jarak(i, j);
      temp[i][j] = jarak[i][j];
      urut[i][j] = j;
      rute[i][j] = 0;
      printf("%f ", jarak[i][j]);
    }
    printf("\n");
  }
  urutkan();
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
    printf("%d ", urut[i][j]);
    }
    printf("\n");
  }
  do {
    l = 500;
    r = 500;
    kapasitas = permintaan[0];
    jawaban[pivot] = 0;
    pivot++;
    if (terjauh(0) != -1) {
      rute[k][l] = terjauh(0);
      flag[rute[k][l]] = 1;
      kapasitas -= permintaan[rute[k][l]];
      printf("%d\n", kapasitas);
      if (terdekat(rute[k][l]) != -1) {
        r++;
        rute[k][r] = terdekat(rute[k][l]);
        flag[rute[k][r]] = 1;
        kapasitas -= permintaan[rute[k][r]];
      }
    }
    while (1) {
      if (jarak[rute[k][r]][0] > jarak[rute[k][l]][0]) {
        if (terdekat(rute[k][r]) != -1) {
          rute[k][r+1] = terdekat(rute[k][r]);
          flag[rute[k][r+1]] = 1;
          kapasitas -= permintaan[rute[k][r+1]];
          r++;
        } else break;
      } else {
        if (terdekat(rute[k][l]) != -1) {
          rute[k][l-1] = terdekat(rute[k][l]);
          flag[rute[k][l-1]] = 1;
          kapasitas -= permintaan[rute[k][l-1]];
          l--;
        } else break;
      }
      for (i = l; i <= r; i++) {
        printf("%d ", rute[k][i]);
      }
      printf("\n");
    }
    for (i = l; i <= r; i++) {
      printf("%d ", rute[k][i]);
      jawaban[pivot] = rute[k][i];
      pivot++;
    }
    printf("\n");
    selesai = 1;
    for (i = 1; i <= N; i++) {
      if (!flag[i]) selesai = 0;
    }
    k++;
  } while (!selesai);
  jawaban[pivot] = 0;
  for (i = 0; i <= pivot; i++) {
    printf("%d ", jawaban[i]);
    jaraktotal += jarak[jawaban[i]][jawaban[i+1]];
  }
```

```
  printf("\n%f\n", jaraktotal);
  return 0;
}

int terdekat(int x) {
  for (int i = 1; i <= N; i++) {
    if (!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >= 0
&& urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

int terjauh(int x) {
  for (int i = N; i >= 1; i--) {
    if(!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >= 0
&& urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

float hitung_jarak(int a, int b) {
  return sqrt(pow((x[a]-x[b]), 2) + pow((y[a]-y[b]), 2));
}

void urutkan() {
  float tmp;
  int tmp2;
  for (int k = 0; k <= N; k++) {
    bool swapped = true;
    int j = 0;
    while (swapped) {
      swapped = false;
      j++;
      for (int i = 1; i <= N - j; i++) {
        if (temp[k][i] > temp[k][i+1]) {
          tmp = temp[k][i];
          temp[k][i] = temp[k][i+1];
          temp[k][i+1] = tmp;
          tmp2 = urut[k][i];
          urut[k][i] = urut[k][i+1];
          urut[k][i+1] = tmp2;
          swapped = true;
        }
      }
    }
  }
  return;
}
```

Procedur ftn5 {prosedur tentang FTN yang memperhitungkan pesebaran *nodes* berdasarkan sumbu X}:

```
#include <cstdio>
#include <cmath>

float x[1000], y[1000], jarak[1000][1000], temp[1000][1000];
int urut[1000][1000], rute[1000][1000], permintaan[1000],
kapasitas;
int jawaban[1000], pivot = 0;
float jaraktotal = 0;
int N, i, j, k = 1;
int l = 50, r = 50;
bool selesai = 1, flag[1000];

float hitung_jarak(int a, int b);
void urutkan();
int terjauh(int x0);
int terdekat(int x0);

int main() {
  scanf("%d", &N);
  for (i = 0; i <= N; i++) {
    scanf("%f %f %d", &x[i], &y[i], &permintaan[i]);
    flag[i] = 0;
  }
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      jarak[i][j] = hitung_jarak(i, j);
      temp[i][j] = jarak[i][j];
      urut[i][j] = j;
      rute[i][j] = 0;
      printf("%f ", jarak[i][j]);
    }
    printf("\n");
  }
  urutkan();
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      printf("%d ", urut[i][j]);
    }
    printf("\n");
  }
  do {
    l = 500;
    r = 500;
    kapasitas = permintaan[0];
    jawaban[pivot] = 0;
    pivot++;
    if (terjauh(0) != -1) {
      rute[k][l] = terjauh(0);
      flag[rute[k][l]] = 1;
      kapasitas -= permintaan[rute[k][l]];
      printf("%d\n", kapasitas);
      if (terdekat(rute[k][l]) != -1) {
        r++;
        rute[k][r] = terdekat(rute[k][l]);
        flag[rute[k][r]] = 1;
        kapasitas -= permintaan[rute[k][r]];
      }
    }
    while (1) {
      if (jarak[rute[k][r]][0] > jarak[rute[k][l]][0]) {
        if (terdekat(rute[k][r]) != -1) {
          rute[k][r+1] = terdekat(rute[k][r]);
          flag[rute[k][r+1]] = 1;
          kapasitas -= permintaan[rute[k][r+1]];
          r++;
        } else break;
      } else {
        if (terdekat(rute[k][l]) != -1) {
          rute[k][l-1] = terdekat(rute[k][l]);
          flag[rute[k][l-1]] = 1;
```

```
        kapasitas -= permintaan[rute[k][l-1]];
        l--;
      } else break;
    }
    for (i = l; i <= r; i++) {
      printf("%d ", rute[k][i]);
    }
    printf("\n");
  }
  for (i = l; i <= r; i++) {
    printf("%d ", rute[k][i]);
    jawaban[pivot] = rute[k][i];
    pivot++;
  }
  printf("\n");
  selesai = 1;
  for (i = 1; i <= N; i++) {
    if (!flag[i]) selesai = 0;
  }
  k++;
 } while (!selesai);
 jawaban[pivot] = 0;
 for (i = 0; i <= pivot; i++) {
  printf("%d ", jawaban[i]);
  jaraktotal += jarak[jawaban[i]][jawaban[i+1]];
 }
 printf("\n%f\n", jaraktotal);
 return 0;
}

int terdekat(int x0) {
 for (int i = 1; i <= N; i++) {
  if (!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (x[urut[x0][i]]-x[0])*(x[x0]-x[0])
>= 0) return urut[x0][i];
 }
 return -1;
}

int terjauh(int x0) {
 for (int i = N; i >= 1; i--) {
  if(!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (x[urut[x0][i]]-x[0])*(x[x0]-x[0])
>= 0) return urut[x0][i];
 }
 return -1;
}

float hitung_jarak(int a, int b) {
 return sqrt(pow((x[a]-x[b]), 2) + pow((y[a]-y[b]), 2));
}

void urutkan() {
 float tmp;
 int tmp2;
 for (int k = 0; k <= N; k++) {
  bool swapped = true;
  int j = 0;
  while (swapped) {
   swapped = false;
   j++;
   for (int i = 1; i <= N - j; i++) {
    if (temp[k][i] > temp[k][i+1]) {
     tmp = temp[k][i];
     temp[k][i] = temp[k][i+1];
     temp[k][i+1] = tmp;
     tmp2 = urut[k][i];
     urut[k][i] = urut[k][i+1];
     urut[k][i+1] = tmp2;
     swapped = true;
    }
   }
  }
 }
 return;
}
```

Procedur ftn6 {prosedur tentang FTN yang memperhitungkan pesebaran *nodes* berdasarkan sumbu Y}:

```
#include <cstdio>
#include <cmath>

float x[1000], y[1000], jarak[1000][1000], temp[1000][1000];
int urut[1000][1000], rute[1000][1000], permintaan[1000],
kapasitas;
int jawaban[1000], pivot = 0;
float jaraktotal = 0;
int N, i, j, k = 1;
int l = 50, r = 50;
bool selesai = 1, flag[1000];

float hitung_jarak(int a, int b);
void urutkan();
int terjauh(int x0);
int terdekat(int x0);

int main() {
 scanf("%d", &N);
 for (i = 0; i <= N; i++) {
  scanf("%f %f %d", &x[i], &y[i], &permintaan[i]);
  flag[i] = 0;
 }
 for (i = 0; i <= N; i++) {
  for (j = 0; j <= N; j++) {
   jarak[i][j] = hitung_jarak(i, j);
   temp[i][j] = jarak[i][j];
   urut[i][j] = j;
   rute[i][j] = 0;
   printf("%f ", jarak[i][j]);
  }
  printf("\n");
 }
 urutkan();
 for (i = 0; i <= N; i++) {
  for (j = 0; j <= N; j++) {
  printf("%d ", urut[i][j]);
  }
  printf("\n");
 }
 do {
  l = 500;
  r = 500;
  kapasitas = permintaan[0];
```

```
  jawaban[pivot] = 0;
  pivot++;
  if (terjauh(0) != -1) {
    rute[k][l] = terjauh(0);
    flag[rute[k][l]] = 1;
    kapasitas -= permintaan[rute[k][l]];
    printf("%d\n", kapasitas);
    if (terdekat(rute[k][l]) != -1) {
      r++;
      rute[k][r] = terdekat(rute[k][l]);
      flag[rute[k][r]] = 1;
      kapasitas -= permintaan[rute[k][r]];
    }
  }
  while (1) {
    if (jarak[rute[k][r]][0] > jarak[rute[k][l]][0]) {
      if (terdekat(rute[k][r]) != -1) {
        rute[k][r+1] = terdekat(rute[k][r]);
        flag[rute[k][r+1]] = 1;
        kapasitas -= permintaan[rute[k][r+1]];
        r++;
      } else break;
    } else {
      if (terdekat(rute[k][l]) != -1) {
        rute[k][l-1] = terdekat(rute[k][l]);
        flag[rute[k][l-1]] = 1;
        kapasitas -= permintaan[rute[k][l-1]];
        l--;
      } else break;
    }
    for (i = l; i <= r; i++) {
      printf("%d ", rute[k][i]);
    }
    printf("\n");
  }
  for (i = l; i <= r; i++) {
    printf("%d ", rute[k][i]);
    jawaban[pivot] = rute[k][i];
    pivot++;
  }
  printf("\n");
  selesai = 1;
  for (i = 1; i <= N; i++) {
    if (!flag[i]) selesai = 0;
  }
  k++;
 } while (!selesai);
 jawaban[pivot] = 0;
 for (i = 0; i <= pivot; i++) {
   printf("%d ", jawaban[i]);
   jaraktotal += jarak[jawaban[i]][jawaban[i+1]];
  }
  printf("\n%f\n", jaraktotal);
  return 0;
}

int terdekat(int x0) {
  for (int i = 1; i <= N; i++) {
    if (!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (y[urut[x0][i]]-y[0])*(y[x0]-y[0])
>= 0) return urut[x0][i];
  }
  return -1;
}

int terjauh(int x0) {
  for (int i = N; i >= 1; i--) {
    if(!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (y[urut[x0][i]]-y[0])*(y[x0]-y[0])
>= 0) return urut[x0][i];
  }
  return -1;
}

float hitung_jarak(int a, int b) {
  return sqrt(pow((x[a]-x[b]), 2) + pow((y[a]-y[b]), 2));
}

void urutkan() {
  float tmp;
  int tmp2;
  for (int k = 0; k <= N; k++) {
    bool swapped = true;
    int j = 0;
    while (swapped) {
      swapped = false;
      j++;
      for (int i = 1; i <= N - j; i++) {
        if (temp[k][i] > temp[k][i+1]) {
          tmp = temp[k][i];
          temp[k][i] = temp[k][i+1];
          temp[k][i+1] = tmp;
          tmp2 = urut[k][i];
          urut[k][i] = urut[k][i+1];
          urut[k][i+1] = tmp2;
          swapped = true;
        }
      }
    }
  }
  return;
}
```

Program ftnfinal {program pemanggil procedure FTN, ftn5, dan ftn6 serta membandingkan ketiganya untuk mendapatkan hasil terbaik}:

```
#include <cstdio>
#include <cmath>

float x[1000], y[1000], jarak[1000][1000], temp[1000][1000];
int urut[1000][1000], rute[1000][1000], permintaan[1000],
kapasitas;
int jawaban[2000], jawaban2[1000], jawaban3[1000], pivot = 0,
pivot2 = 0, pivot3 = 0;
float jaraktotal = 0, jaraktotal3 = 0, jaraktotal2 = 0;
int N, i, j, k = 1;
int l = 50, r = 50;
bool selesai = 1, flag[1000];

float hitung_jarak(int a, int b);
void urutkan();
int terjauh(int x);
int terdekat(int x);
void asma();
```

```
void asma5();
void asma6();
int terjauhx(int x0);
int terdekatx(int x0);
int terjauhy(int x0);
int terdekaty(int x0);
float min(float a, float b) {
  if (a <= b) return a;
  else return b;
}

int main() {
  FILE *fp;
  fp = fopen("hasil.txt", "w+");
  scanf("%d", &N);
  for (i = 0; i <= N; i++) {
    scanf("%f %f %d", &x[i], &y[i], &permintaan[i]);
    flag[i] = 0;
  }
  asma();
  asma5();
  asma6();
  if (min(jaraktotal,min(jaraktotal2,jaraktotal3)) == jaraktotal)
    fprintf(fp, ",Solusi Inisial,,\n");
  else     if     (min(jaraktotal,min(jaraktotal2,jaraktotal3))     ==
jaraktotal2)
    fprintf(fp, ",,Solusi Inisial,\n");
  else     if     (min(jaraktotal,min(jaraktotal2,jaraktotal3))     ==
jaraktotal3)
    fprintf(fp, ",,,Solusi Inisial\n");
  fprintf(fp, ",FTNs,FTNx,FTNy\n");
  fprintf(fp, "Total Jarak,%f,%f,%f\n", jaraktotal, jaraktotal2,
jaraktotal3);
  for (i = 0; i <= pivot || i <= pivot2 || i <= pivot3; i++) {
    if (i == 0) fprintf(fp, "Rute");
    fprintf(fp, ",");
    if (i <= pivot) fprintf(fp, "%d", jawaban[i]);
    fprintf(fp, ",");
    if (i <= pivot2) fprintf(fp, "%d", jawaban2[i]);
    fprintf(fp, ",");
    if (i <= pivot3) fprintf(fp, "%d\n", jawaban3[i]);
  }
  return 0;
}

void asma6() {
  pivot3 = 0;
  jaraktotal3 = 0;
  for (i = 0; i <= 999; i++) {
    jawaban3[i] = 0;
    flag[i] = 0;
  }
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      jarak[i][j] = hitung_jarak(i, j);
      temp[i][j] = jarak[i][j];
      urut[i][j] = j;
      rute[i][j] = 0;
      // printf("%f ", jarak[i][j]);
    }
    // printf("\n");
  }
  urutkan();
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      // printf("%d ", urut[i][j]);
    }
    // printf("\n");
  }
  do {
    l = 500;
    r = 500;
    kapasitas = permintaan[0];
    jawaban3[pivot3] = 0;
    pivot3++;
    if (terjauhy(0) != -1) {
      rute[k][l] = terjauhy(0);
      flag[rute[k][l]] = 1;
      kapasitas -= permintaan[rute[k][l]];
      // printf("%d\n", kapasitas);
      if (terdekaty(rute[k][l]) != -1) {
        r++;
        rute[k][r] = terdekaty(rute[k][l]);
        flag[rute[k][r]] = 1;
        kapasitas -= permintaan[rute[k][r]];
      }
    }
    while (1) {
      if (jarak[rute[k][r]][0] > jarak[rute[k][l]][0]) {
        if (terdekaty(rute[k][r]) != -1) {
          rute[k][r+1] = terdekaty(rute[k][r]);
          flag[rute[k][r+1]] = 1;
          kapasitas -= permintaan[rute[k][r+1]];
          r++;
        } else break;
      } else {
        if (terdekaty(rute[k][l]) != -1) {
          rute[k][l-1] = terdekaty(rute[k][l]);
          flag[rute[k][l-1]] = 1;
          kapasitas -= permintaan[rute[k][l-1]];
          l--;
        } else break;
      }
      for (i = l; i <= r; i++) {
        // printf("%d ", rute[k][i]);
      }
      // printf("\n");
    }
    for (i = l; i <= r; i++) {
      // printf("%d ", rute[k][i]);
      jawaban3[pivot3] = rute[k][i];
      pivot3++;
    }
    // printf("\n");
    selesai = 1;
    for (i = 1; i <= N; i++) {
      if (!flag[i]) selesai = 0;
    }
    k++;
  } while (!selesai);
  jawaban3[pivot3] = 0;
  printf("Hasil FTNy: ");
  for (i = 0; i <= pivot3; i++) {
    printf("%d ", jawaban3[i]);
    jaraktotal3 += jarak[jawaban3[i]][jawaban3[i+1]];
  }
  printf("\n%f\n", jaraktotal3);
}

void asma5() {
  jaraktotal2 = 0;
  pivot2 = 0;
```

```
for (i = 0; i <= 999; i++) {
  jawaban2[i] = 0;
  flag[i] = 0;
}
for (i = 0; i <= N; i++) {
  for (j = 0; j <= N; j++) {
    jarak[i][j] = hitung_jarak(i, j);
    temp[i][j] = jarak[i][j];
    urut[i][j] = j;
    rute[i][j] = 0;
    // printf("%f ", jarak[i][j]);
  }
  // printf("\n");
}
urutkan();
for (i = 0; i <= N; i++) {
  for (j = 0; j <= N; j++) {
    // printf("%d ", urut[i][j]);
  }
  // printf("\n");
}
do {
  l = 500;
  r = 500;
  kapasitas = permintaan[0];
  jawaban2[pivot2] = 0;
  pivot2++;
  if (terjauhx(0) != -1) {
    rute[k][l] = terjauhx(0);
    flag[rute[k][l]] = 1;
    kapasitas -= permintaan[rute[k][l]];
    // printf("%d\n", kapasitas);
    if (terdekatx(rute[k][l]) != -1) {
      r++;
      rute[k][r] = terdekatx(rute[k][l]);
      flag[rute[k][r]] = 1;
      kapasitas -= permintaan[rute[k][r]];
    }
  }
  while (1) {
    if (jarak[rute[k][r]][0] > jarak[rute[k][l]][0]) {
      if (terdekatx(rute[k][r]) != -1) {
        rute[k][r+1] = terdekatx(rute[k][r]);
        flag[rute[k][r+1]] = 1;
        kapasitas -= permintaan[rute[k][r+1]];
        r++;
      } else break;
    } else {
      if (terdekatx(rute[k][l]) != -1) {
        rute[k][l-1] = terdekatx(rute[k][l]);
        flag[rute[k][l-1]] = 1;
        kapasitas -= permintaan[rute[k][l-1]];
        l--;
      } else break;
    }
    for (i = l; i <= r; i++) {
      // printf("%d ", rute[k][i]);
    }
    // printf("\n");
  }
  for (i = l; i <= r; i++) {
    // printf("%d ", rute[k][i]);
    jawaban2[pivot2] = rute[k][i];
    pivot2++;
    // printf("%d ", pivot2);
  }
  // printf("\n");
  selesai = 1;
  for (i = 1; i <= N; i++) {
    if (!flag[i]) selesai = 0;
  }
  k++;
} while (!selesai);
jawaban2[pivot2] = 0;
printf("Hasil FTNx: ");
for (i = 0; i <= pivot2; i++) {
  printf("%d ", jawaban2[i]);
  jaraktotal2 += jarak[jawaban2[i]][jawaban2[i+1]];
}
printf("\n%f\n", jaraktotal2);
}

void asma() {
  pivot = 0;
  jaraktotal = 0;
  for (i = 0; i <= 1000; i++) {
    jawaban[i] = 0;
    flag[i] = 0;
  }
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      jarak[i][j] = hitung_jarak(i, j);
      temp[i][j] = jarak[i][j];
      urut[i][j] = j;
      rute[i][j] = 0;
      // printf("%f ", jarak[i][j]);
    }
    // printf("\n");
  }
  urutkan();
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      // printf("%d ", urut[i][j]);
    }
    // printf("\n");
  }
  do {
    l = 500;
    r = 500;
    kapasitas = permintaan[0];
    jawaban[pivot] = 0;
    pivot++;
    if (terjauh(0) != -1) {
      rute[k][l] = terjauh(0);
      flag[rute[k][l]] = 1;
      kapasitas -= permintaan[rute[k][l]];
      // printf("%d\n", kapasitas);
      if (terdekat(rute[k][l]) != -1) {
        r++;
        rute[k][r] = terdekat(rute[k][l]);
        flag[rute[k][r]] = 1;
        kapasitas -= permintaan[rute[k][r]];
      }
    }
    while (1) {
      if (jarak[rute[k][r]][0] > jarak[rute[k][l]][0]) {
        if (terdekat(rute[k][r]) != -1) {
          rute[k][r+1] = terdekat(rute[k][r]);
          flag[rute[k][r+1]] = 1;
          kapasitas -= permintaan[rute[k][r+1]];
          r++;
        } else break;
```

```
      } else {
        if (terdekat(rute[k][l]) != -1) {
          rute[k][l-1] = terdekat(rute[k][l]);
          flag[rute[k][l-1]] = 1;
          kapasitas -= permintaan[rute[k][l-1]];
          l--;
        } else break;
      }
      for (i = l; i <= r; i++) {
        // printf("%d ", rute[k][i]);
      }
      // printf("\n");
    }
    for (i = l; i <= r; i++) {
      // printf("%d ", rute[k][i]);
      jawaban[pivot] = rute[k][i];
      pivot++;
    }
    // printf("\n");
    selesai = 1;
    for (i = 1; i <= N; i++) {
      if (!flag[i]) selesai = 0;
    }
    k++;
  } while (!selesai);
  jawaban[pivot] = 0;
  printf("Hasil FTNs: ");
  for (i = 0; i <= pivot; i++) {
    printf("%d ", jawaban[i]);
    jaraktotal += jarak[jawaban[i]][jawaban[i+1]];
  }
  printf("\n%f\n", jaraktotal);
}

int terdekat(int x) {
  for (int i = 1; i <= N; i++) {
    if (!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >= 0
&& urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

int terjauh(int x) {
  for (int i = N; i >= 1; i--) {
    if(!flag[urut[x][i]] && kapasitas-permintaan[urut[x][i]] >= 0
&& urut[x][i] != x) return urut[x][i];
  }
  return -1;
}

int terdekatx(int x0) {
  for (int i = 1; i <= N; i++) {
    if (!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (x[urut[x0][i]]-x[0])*(x[x0]-x[0])
>= 0) return urut[x0][i];
  }
  return -1;
}

int terjauhx(int x0) {
  for (int i = N; i >= 1; i--) {
    if(!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (x[urut[x0][i]]-x[0])*(x[x0]-x[0])
>= 0) return urut[x0][i];
  }
  return -1;
}
int terdekaty(int x0) {
  for (int i = 1; i <= N; i++) {
    if (!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (y[urut[x0][i]]-y[0])*(y[x0]-y[0])
>= 0) return urut[x0][i];
  }
  return -1;
}

int terjauhy(int x0) {
  for (int i = N; i >= 1; i--) {
    if(!flag[urut[x0][i]] && kapasitas-permintaan[urut[x0][i]] >=
0 && urut[x0][i] != x0 && (y[urut[x0][i]]-y[0])*(y[x0]-y[0])
>= 0) return urut[x0][i];
  }
  return -1;
}

float hitung_jarak(int a, int b) {
  return sqrt(pow((x[a]-x[b]), 2) + pow((y[a]-y[b]), 2));
}

void urutkan() {
  float tmp;
  int tmp2;
  for (int k = 0; k <= N; k++) {
    bool swapped = true;
    int j = 0;
    while (swapped) {
      swapped = false;
      j++;
      for (int i = 1; i <= N - j; i++) {
        if (temp[k][i] > temp[k][i+1]) {
          tmp = temp[k][i];
          temp[k][i] = temp[k][i+1];
          temp[k][i+1] = tmp;
          tmp2 = urut[k][i];
          urut[k][i] = urut[k][i+1];
          urut[k][i+1] = tmp2;
          swapped = true;
        }
      }
    }
  }
  return;
}
```