TUGAS KECIL 2 STRATEGI ALGORITMA

IMPLEMENTASI ALGORITMA *DIVIDE AND CONQUER*

Disusun untuk memenuhi laporan tugas kecil mata kuliah Strategi Algoritma semester 2 Institut Teknologi Bandung

Disusun oleh:

Nigel Sahl	13521043
Athif Nirwasito	13521053

TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

Jl. Ganesa No. 10, Lb. Siliwangi, Kecamatan Coblong,

Kota Bandung, Jawa Barat, 40132

2023

Daftar Isi

I. Deskripsi Persoalan

Mencari Pasangan Titik Terdekat 3D dengan Algoritma Divide and Conquer

Mencari sepasang titik terdekat dengan Algoritma Divide and Conquer sudah dijelaskan di dalam kuliah Stima. Persoalan tersebut dirumuskan untuk titik pada bidang datar (2D). Pada Tucil 2 kali ini kita mengembangkan algoritma mencari sepasang titik terdekat pada bidang 3D. Misalkan terdapat n buah titik pada ruang 3D. Setiap titik P di dalam ruang dinyatakan dengan koordinat P = (x, y, z). Kita mencari sepasang titik yang mempunyai jarak terdekat satu sama lain. Jarak dua buah titk P1 = (x1, y1, z1) dan P2 = (x2, y2, z2) dihitung dengan rumus Euclidean berikut:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Program yang kami buat menggunakan Bahasa Python untuk mencari sepasang titik yang jaraknya terdekat datu sama lain dengan menerapkan algoritma divide and conquer untuk penyelesaiannya, dan perbandingannya dengan Algoritma Brute Force.

Masukan program:

- Banyaknya dimensi
- Banyaknya titik N
- titik-titik (dibangkitkan secara acak) dalam koordinat (x, y, z) atau untuk dimensi lain maka (x1, x2, x3, … xn)

Luaran program:

- Sepasang titik yang jaraknya terdekat dan nilai jaraknya
- Banyaknya operasi perhitungan rumus Euclidian
- Waktu riil dalam detik (spesifikasikan komputer yang digunakan)
- Penggambaran semua titik dalam bidang 3D, sepasang titik yang jaraknya terdekat ditunjukkan dengan warna yang berbeda dari titik lainnya.

II. Landasan Teori

Definisi

- Divide: membagi persoalan menjadi beberapa upa-persoalan yang memiliki kemiripan dengan persoalan semula namun berukuran lebih kecil (idealnya berukuran hampir sama),
- Conquer (solve): menyelesaikan masing-masing upa-persoalan (secara langsung jika sudah berukuran kecil atau secara rekursif jika masih berukuran besar).
- Combine: mengabungkan solusi masing-masing upa-persoalan sehingga membentuk solusi persoalan semula.

Skema Umum Algoritma *Divide and Conquer*

```
procedure DIVIDEandCONQUER(input P : problem, n : integer)
{ Menyelesaikan persoalan P dengan algoritma divide and conquer
  Masukan: masukan persoalan P berukuran n
  Luaran: solusi dari persoalan semula   }
Deklarasi
   r : integer

Algoritma
  if n ≤ n0 then {ukuran persoalan P sudah cukup kecil }
     SOLVE persoalan P yang berukuran n ini
  else
     DIVIDE menjadi r upa-persoalan, P1, P2, …, Pr , yang masing-masing berukuran n1, n2, …, nr
     for masing-masing P1, P2, …, Pr , do
        DIVIDEandCONQUER(Pi, ni)
     endfor
     COMBINE solusi dari P1, P2, …, Pr menjadi solusi persoalan semula
  endif
```

Kompleksitas algoritma *divide and conquer*: $T(n) = \begin{cases} g(n) & ,n \leq n_0 \\ T(n_1) + T(n_2) \ldots + T(n_r) + f(n) & ,n > n_0 \end{cases}$

Penjelasan:

$$T(n) = \begin{cases} g(n) & ,n \leq n_0 \\ T(n_1) + T(n_2) \ldots + T(n_r) + f(n) & ,n > n_0 \end{cases}$$

- $T(n)$: kompleksitas waktu penyelesaian persoalan *P* yang berukuran *n*
- $g(n)$: kompleksitas waktu untuk SOLVE jika *n* sudah berukuran kecil
- $T(n_1) + T(n_2) \ldots + T(n_r)$: kompleksitas waktu untuk memproses setiap upa-persoalan
- $f(n)$: kompleksitas waktu untuk COMBINE solusi dari masing-masing upa-persoalan

III. Algoritma *Divide and Conquer*

Algoritma pencarian dua titik terdekat untuk dimensi 2, 3 dan seterusnya dapat dirincikan sebagai berikut:

1. Urutkan array of points berdasarkan besar koordinat x.
2. Jika jumlah titik < =3, selesaikan dengan bruteforce.
3. Jika jumlah titik > 3, cari L yang merupakan median dari semua koordinat x, kemudian bagi ruang menjadi 2 ruang dengan jumlah point yang sama menggunakan median L.
4. Lakukan rekursi pada setiap ruang dan simpan nilainya dalam d1 dan d2.
5. Ambil nilai terkecil dari d1 dan d2, simpan di d3.
6. Kumpulkan semua titik yang berada di antara L + d3 dan L-d3 pada sumbu x, simpan dalam array SL.
7. Urutkan semua titik pada SL berdasarkan sumbu Y.
8. Lakukan traversal pada semua titik pada SL. Untuk setiap titik, lakukan perbandingan pada titik-titik selanjutnya pada sumbu Y menggunakan jarak euclidean. Jika jarak euclideannya lebih kecil dari nilai d3, ubah nilai d3 dengan nilai jarak euclidean tersebut. Ulangi hingga selisih titik dengan titik yang dibandingkannya pada sumbu Y melebihi nilai d3. Jika melebihi d3, lanjutkan traversal pada titik selanjutnya.
9. Kembalikan nilai d3.

IV. *Source Code* Program

1. Program Fungsi Divide and Conquer

```
def DivNCon(S, D, count):
    # S: Set of points
    # D: Dimension
    # count: Frekuensi pemanggilan fungsi EuclideanDistanceBF
    #Brute Force untuk len(S)<=3#
    if (len(S)==3):
        d2, count = DivNCon(S[1:3], D, count)
        dm1 = EuclideanDistanceBF(S[0], S[1])
        dm2 = EuclideanDistanceBF(S[0], S[2])
        count +=2
        if d2[0] <=  dm1[0] and d2[0] <= dm2[0]:
            return d2, count
        elif dm1[0] <= dm2[0] and dm1[0] <= d2[0]:
            return dm1,count
        else:
            return dm2, count

    elif(len(S) == 2):
        count+=1
        return EuclideanDistanceBF(S[0],S[1]), count
```

```
    else:
        d1, count = DivNCon(S[:floor(len(S)/2)], D,count)
        d2,count = DivNCon(S[floor(len(S)/2):], D,count)
        if (d1[0] <= d2[0]):
            d3 = d1
        else:
            d3 = d2
        SL = np.empty((0,D), int)
        if(len(S) %2 == 0):
            L= (S[floor(len(S)/2)][0] + S[floor(len(S)/2)-1][0])/2
        else:
            L  = S[floor(len(S)/2)][0]

        rDistLeft = L - d3[0]
        rDistRight = L + d3[0]
        for i in range(len(S)):
            if rDistRight>= S[i][0] >= rDistLeft and S[i][0] <= rDistRight:
                SL = np.append(SL,[S[i]], axis = 0)
```

```
if (D != 1):
    sortListY(SL)
    for i in range(len(SL)):
        j= i+1
        while(j<len(SL) and SL[j][1] - SL[i][1]<d3[0]):
            count +=1
            temp = EuclideanDistanceBF(SL[i], SL[j])
            if temp[0] <= d3[0]:
                d3 = temp
            j+=1
else:
    for i in range(len(SL)):
        j = i+1
        while j<len(S) and SL[j][0] - SL[i][0]<d3[0]:
            temp = EuclideanDistanceBF(SL[i], SL[j])
            count+=1
            if (temp[0] <= d3[0]):
                d3 = temp
return d3, count
```

2. Program Sort

Tampilan secara umum

```
# Python program for implementation of Quicksort Sort

import numpy as np
S2 = np.array([[1,3,0,1],[12,30,0,1],[40,50,0,1],[5,1,0,0],[12,10,0,0],[3,4,0,1]])

def partition(S, low, high, axes):···

# function to perform quicksort
def quickSort(S, low, high, axes):···

def swap(S, i, j):···

def sortList(S):···

def sortListY(S):···
```

Partisi

```
def partition(S, low, high, axes):

    # choose the rightmost element as pivot
    pivot = S[high][axes]

    # pointer for greater element
    i = low - 1

    # traverse through all elements
    # compare each element with pivot
    for j in range(low, high):
        if S[j][axes] <= pivot:

            # If element smaller than pivot is found
            # swap it with the greater element pointed by i
            i = i + 1

            # Swapping element at i with element at j
            swap(S, i, j)

    # Swap the pivot element with the greater element specified by i
    swap(S, i+1, high)

    # Return the position from where partition is done
    return i + 1
```

Quick Sort

```
# function to perform quicksort
def quickSort(S, low, high, axes):
    if low < high:

        # Find pivot element such that
        # element smaller than pivot are on the left
        # element greater than pivot are on the right
        pi = partition(S, low, high, axes)

        # Recursive call on the left of pivot
        quickSort(S, low, pi - 1, axes)

        # Recursive call on the right of pivot
        quickSort(S, pi + 1, high, axes)
```

Swap

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

Sort List

```
def sortList(S):
    return (quickSort(S, 0, len(S) - 1, 0))

def sortListY(S):
    return (quickSort(S, 0, len(S) - 1, 1))
```

3. Program Main

Import function, time, dan platform

```
from functions import *
from time import time
import platform
```

Inisiasi awal dan beberapa *test case*

```
lower_bound = 0
upper_bound = 1000
dimension = 3

# Test Case
test = np.array([[1,1,0],[3,0,1],[7,8,9]])
S = np.array([[2,3],[12,30],[40,50],[5,1],[12,10],[3,4]])
S2 = np.array([[1,3,0,1],[12,30,0,1],[40,50,0,1],[5,1,0,0],[12,10,0,0],[3,4,0,1]])
Stest = np.array([[4,3,1],[3,1,0],[3,3,4],[4,3,0]])
Stest2 = np.array([[78, 75],[80, 85],[78, 20],[57, 47], [4, 80]])
Stest3 = np.array([[79,40],···
Stest4 = np.array([[51, 32],[51, 32],[78, 20],[57, 47], [4, 80]])
Stest5 = np.array([[51, 32],[51, 32],[78, 20], [4, 80]])
Stest6 = np.array([[42, 91],···

# Menerima input dari user
```

Input dari user

```
# Menerima input dari user
print("Masukan banyak dimensi")
dimension = int(input("dimensi = "))
print("Masukan banyak titik")
n = int(input("n = "))
List_Points = np.empty((0,dimension), int)
print()
```

Random titik

```
# Random titik
for i in range(n):
    # print("Masukan titik ke-", i+1, ": ")
    # x1 = int(input())
    # x2 = int(input())
    # x3 = int(input())
    # print()
    # List_Points = np.append(List_Points, np.array([[x1,x2,x3]]), axis=0)
    List_Points = np.append(List_Points, np.random.randint(lower_bound, upper_bound, size=(1, dimension)), axis=0)
```

Eksekusi Program *Divide and Conquer*

```
# Eksekusi program
List_PointsBruteForce = (List_Points.copy())
sortList(List_PointsBruteForce)
sortList(List_Points)
# print(List_Points)
# print(len(sortedListX))
# print(len(List_Points))
t1 = time()
hasil, count = DivNCon(List_Points, dimension, 0)
t2 = time()
hasilB, brute = BruteForce(List_PointsBruteForce)
t3 = time()
```

***Output* Program**

```
# Menampilkan hasil
print("titik 1 = ", hasil[1])
print("titik 2 = ", hasil[2])
print("Distance = ", hasil[0])
print()
print("titik 1 Brute = ", hasilB[1])
print("titik 2 Brute = ", hasilB[2])
print("Distance = ", hasilB[0])

print()
if (hasil[0] == hasilB[0]):
    print("Hasil sama")
else:
    print("Hasil beda")

print("time divide and conquer = ", t2-t1)
print("Euclidean distance function call count = ", count)
print()
print("time brute force = ", t3-t2)
print("Euclidean distance function call count = ", brute)
print()
print("dieksekusi pada ", platform.machine())
```

4. Program Visualisasi

Import hal-hal yang dibutuhkan

```
# importing main, numpy and matplotlib
import numpy as np
import matplotlib.pyplot as plt
from main import *
```

Inisiasi awal

```
fig = plt.figure()
pointHasilColor = 'brown'
pointHasilColorB = 'yellow'
pointsColor = 'green'
lineHasilColor = 'red'
lineHasilColorB = 'yellow'
lineWidthConts = 2
```

Pembagian kondisi untuk 3 dimensi yang dapat divisualisasi yaitu 1D, 2D, dan 3D

```
15  if (dimension == 3):···
46
47  elif(dimension == 2 ):···
71
72  elif (dimension == 1):···
97
98    else :
99        print('Error plotting, dimension not supported')
100
```

Defining all axis

```
if (dimension == 3):
    ax = plt.axes(projection =str(dimension)+'d')
    ax.invert_yaxis()
    # defining all 3 axis
    x = List_Points[:,0]
    y = List_Points[:,1]
    z = List_Points[:,2]
    Phasil = np.concatenate(([hasil[1]], [hasil[2]]), axis=0)
    # PhasilBrute = np.concatenate(([hasilB[1]], [hasilB[2]]), axis=0)

    xhasil = Phasil[:,0]
    yhasil = Phasil[:,1]
    zhasil = Phasil[:,2]
    # xhasilB = PhasilBrute[:,0]
    # yhasilB = PhasilBrute[:,1]
    # zhasilB = PhasilBrute[:,2]
```

Plotting, making points, and making line

```
# make points
ax.scatter3D( x, y, z, color = pointsColor)
ax.scatter3D( xhasil, yhasil, zhasil, color = pointHasilColor)
# ax.scatter3D( xhasil, yhasil, zhasil, color = pointHasilColorB)
ax.set_title('3D line plot')
ax.set_xlabel('x-axis')
ax.set_ylabel('y-axis')
ax.set_zlabel('z-axis')
plt.show()
```

Visualisasi pada dimensi 2

```
elif(dimension == 2 ):
    # defining all 3 axis
    x = List_Points[:,0]
    y = List_Points[:,1]
    Phasil = np.concatenate(([hasil[1]], [hasil[2]]), axis=0)
    # PhasilBrute = np.concatenate(([hasilB[1]], [hasilB[2]]), axis=0)

    xhasil = Phasil[:,0]
    yhasil = Phasil[:,1]
    # xhasilB = PhasilBrute[:,0]
    # yhasilB = PhasilBrute[:,1]

    # plotting
    plt.plot(xhasil, yhasil, lineHasilColor, linewidth = lineWidthConts)
    # plt.plot(xhasilB, yhasilB, lineHasilColorB, linewidth = lineWidthConts)

    # make points
    plt.scatter( x, y, color = pointsColor)
    plt.scatter( xhasil, yhasil, color = pointHasilColor)
    # plt.scatter( xhasilB, yhasilB, color = pointHasilColorB)
    plt.title('2D line plot')
    plt.xlabel('x-axis')
    plt.ylabel('y-axis')
    plt.show()
```

Visualisasi pada dimensi 1

```
elif (dimension == 1):
    # defining all 3 axis
    x = List_Points[:,0]
    y = np.array([0]*len(List_Points))
    Phasil = np.concatenate(([hasil[1]], [hasil[2]]), axis=0)
    # PhasilBrute = np.concatenate(([hasilB[1]], [hasilB[2]]), axis=0)

    xhasil = Phasil[:,0]
    yhasil = np.array([0]*len(Phasil))
    # xhasilB = PhasilBrute[:,0]
    # yhasilB = np.array([0]*len(PhasilBrute))

    # plotting
    plt.plot(xhasil, yhasil,  lineHasilColor, linewidth = lineWidthConts)
    # plt.plot(xhasilB, lineHasilColorB, linewidth = lineWidthConts)

    # make points
    plt.ylim([-1, 1])
    plt.scatter( x, y, color = pointsColor)
    plt.scatter( xhasil, yhasil, color = pointHasilColor)
    # plt.scatter( xhasilB, yhasilB, color = pointHasilColorB)
    plt.title('1D line plot')
    plt.xlabel('x-axis')
    plt.ylabel('y-axis')
    plt.show()
```

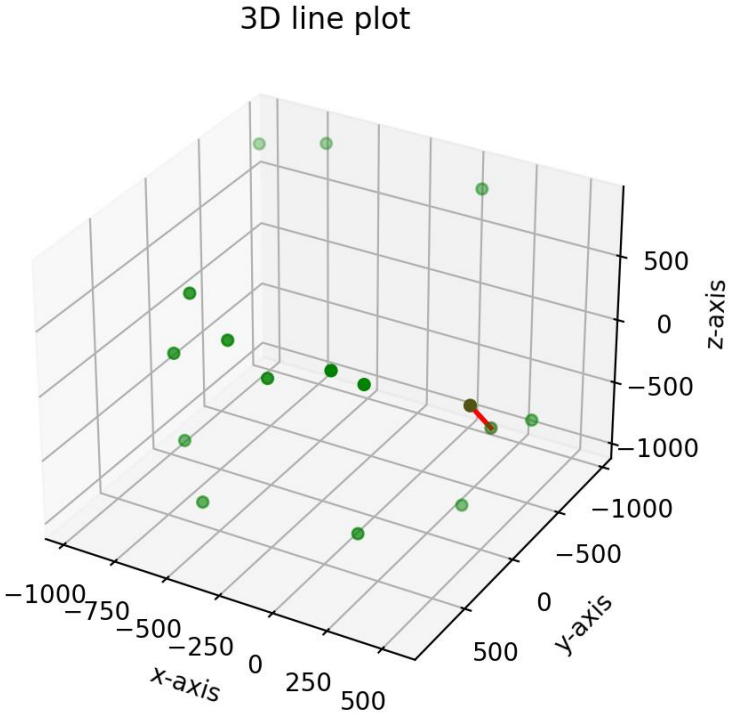
V. Ketercapaian Program

No	Poin	Ya	Tidak
1	Program berhasil dikompilasi tanpa ada kesalahan.	✅	
2	Program berhasil running	✅	
3	Program dapat menerima masukan dan dan menuliskan luaran.	✅	
4	Luaran program sudah benar (solusi closest pair benar)	✅	
5	Bonus 1 dikerjakan	✅	
6	Bonus 2 dikerjakan	✅	

VI. *Screenshot* Luaran Program

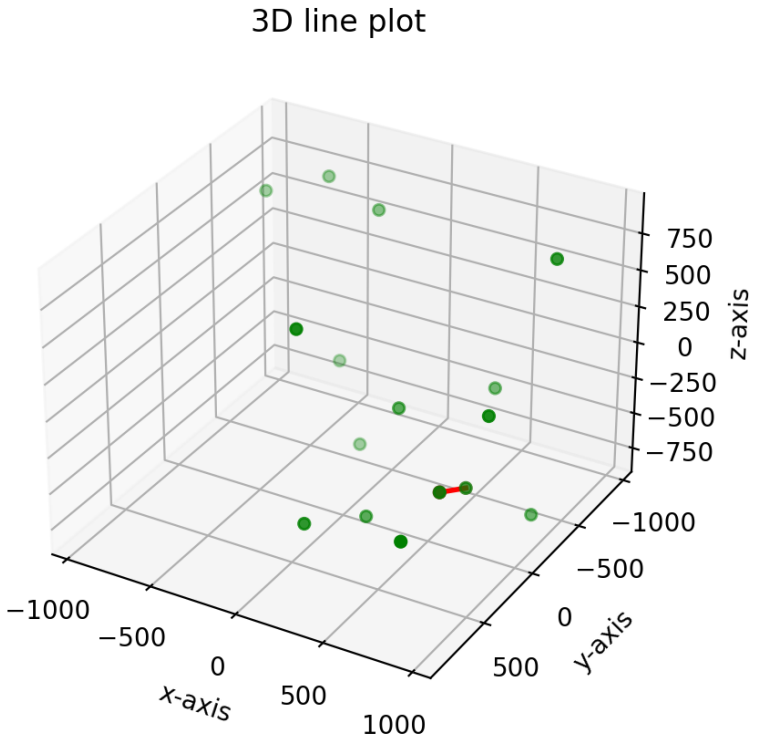
1. Test case dimensi 3
    a. n =16
        i. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 16

titik 1 =  [ 315.87872192 -515.53042021 -619.15449802]
titik 2 =  [ 223.0433898  -499.17165623 -472.59395384]
Distance =  174.25843210538832

titik 1 Brute =  [ 315.87872192 -515.53042021 -619.15449802]
titik 2 Brute =  [ 223.0433898  -499.17165623 -472.59395384]
Distance =  174.25843210538832

Hasil sama
time divide and conquer =  0.0003044605255126953
Euclidean distance function call count =  35

time brute force =  0.0011165142059326172
Euclidean distance function call count =  121

dieksekusi pada  AMD64
```



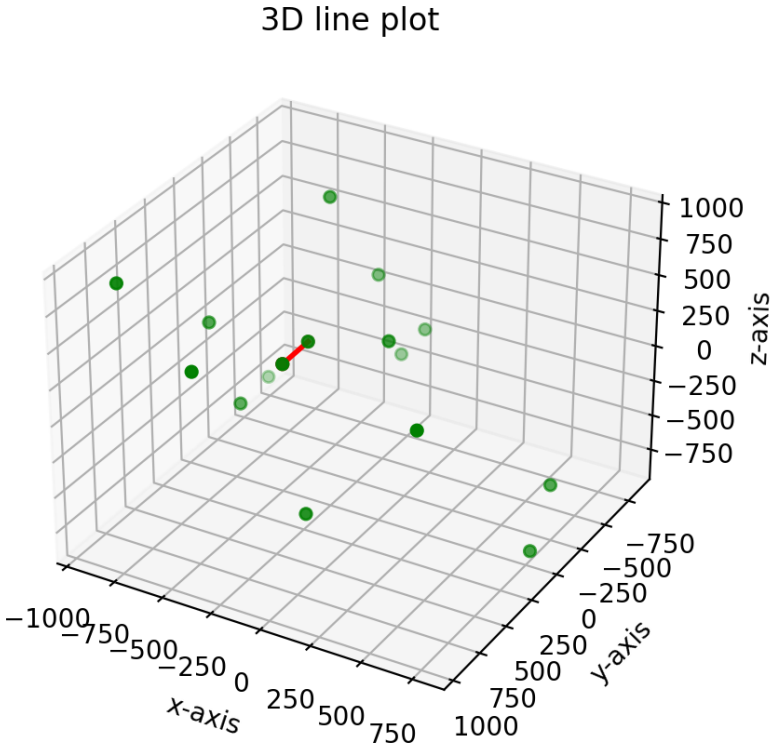
ii. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 16

titik 1 =  [ 738.34913638   47.11940676 -426.73702852]
titik 2 =  [ 746.76647324  330.42588096 -267.9862528 ]
Distance =  324.86184552475555

titik 1 Brute =  [ 746.76647324  330.42588096 -267.9862528 ]
titik 2 Brute =  [ 738.34913638   47.11940676 -426.73702852]
Distance =  324.86184552475555

Hasil sama
time divide and conquer =  0.0010159015655517578
Euclidean distance function call count =  30

time brute force =  0.002243518829345703
Euclidean distance function call count =  121

dieksekusi pada  AMD64
```



iii. _

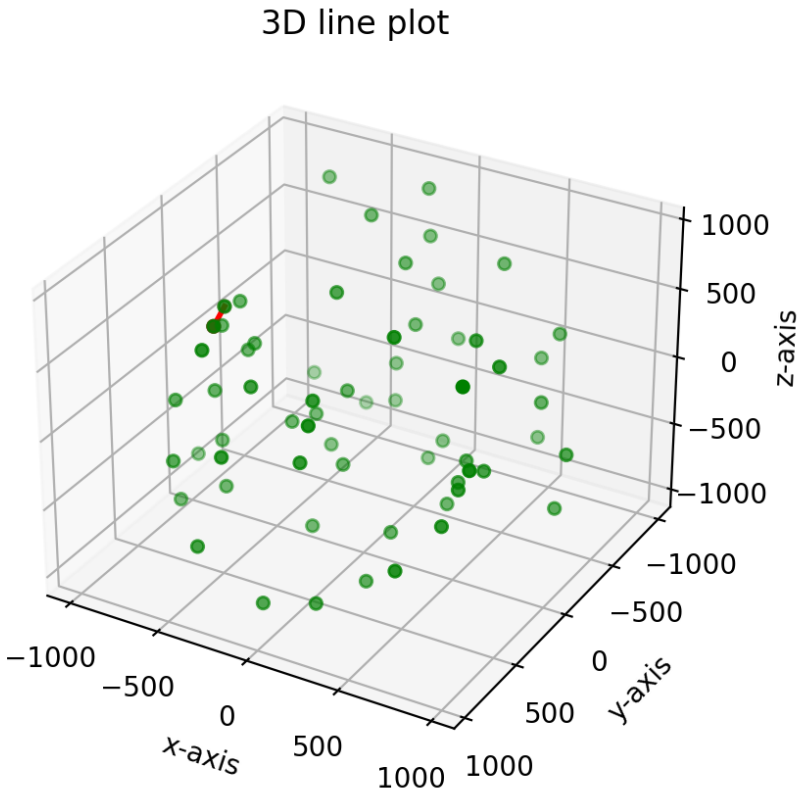
```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 16

titik 1 =  [-136.59321627  397.95785637  505.34007149]
titik 2 =  [-143.67650754  611.78467821  487.069565  ]
Distance =  214.7228310158507

titik 1 Brute =  [-136.59321627  397.95785637  505.34007149]
titik 2 Brute =  [-143.67650754  611.78467821  487.069565  ]
Distance =  214.7228310158507

Hasil sama
time divide and conquer =  0.0009067058563232422
Euclidean distance function call count =  29

time brute force =  0.0011556148529052734
Euclidean distance function call count =  121

dieksekusi pada  AMD64
```



b. n=64

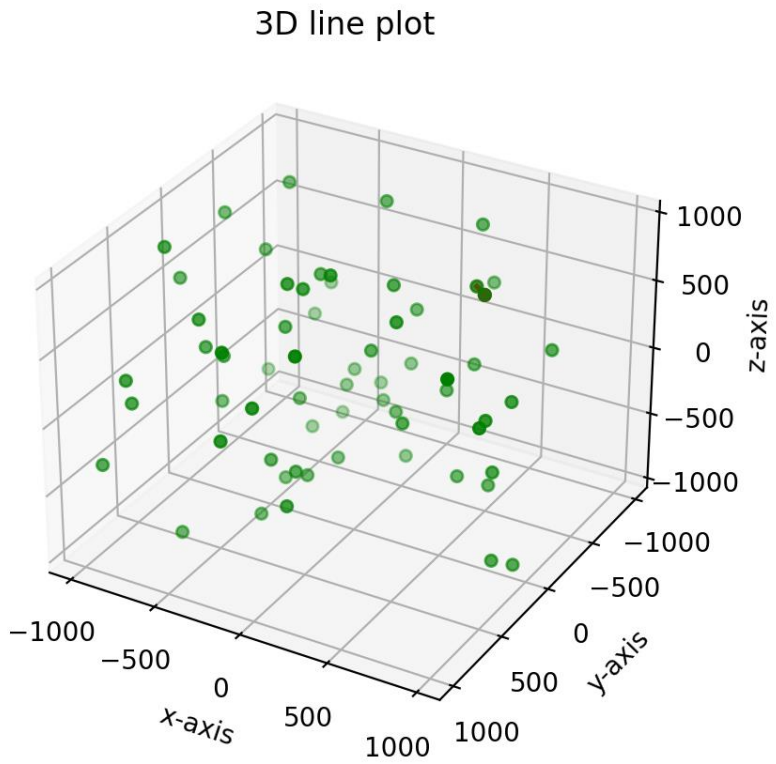
i. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 64

titik 1 =  [-428.5408501   557.53789592  876.82735193]
titik 2 =  [-447.62437187  625.47612017  772.94887045]
Distance =  125.58073909560973

titik 1 Brute =  [-428.5408501   557.53789592  876.82735193]
titik 2 Brute =  [-447.62437187  625.47612017  772.94887045]
Distance =  125.58073909560973

Hasil sama
time divide and conquer =  0.006091594696044922
Euclidean distance function call count =  217

time brute force =  0.011623144149780273
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```



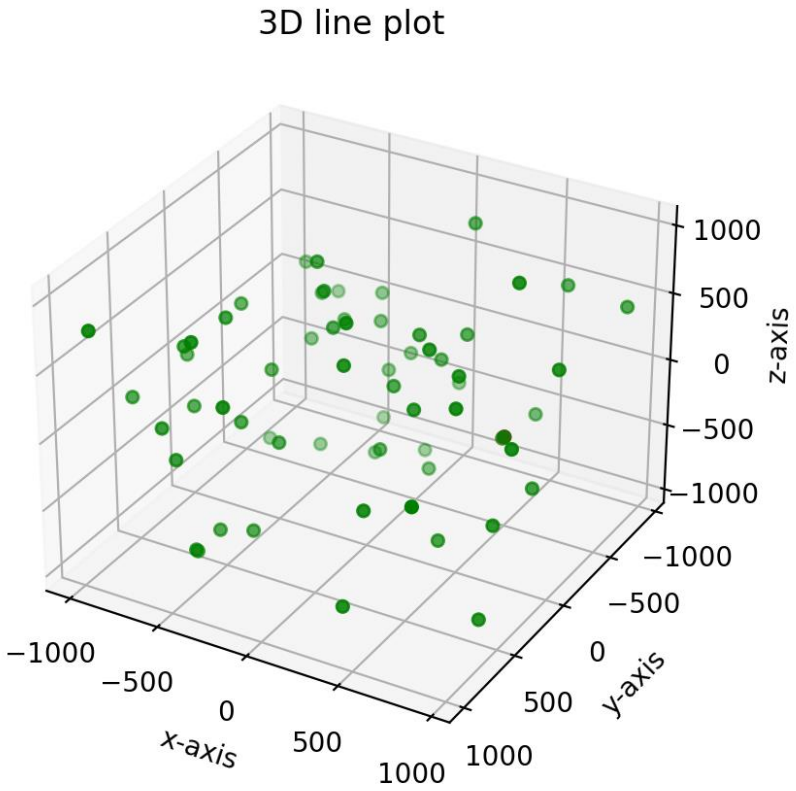
ii. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 64

titik 1 =  [ 536.25810834 -294.35831718  752.97534058]
titik 2 =  [ 583.42541566 -291.44745502  709.13862115]
Distance =  64.45840495251629

titik 1 Brute =  [ 583.42541566 -291.44745502  709.13862115]
titik 2 Brute =  [ 536.25810834 -294.35831718  752.97534058]
Distance =  64.45840495251629

Hasil sama
time divide and conquer =  0.003590106964111328
Euclidean distance function call count =  127

time brute force =  0.006772518157958984
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```



iii. _

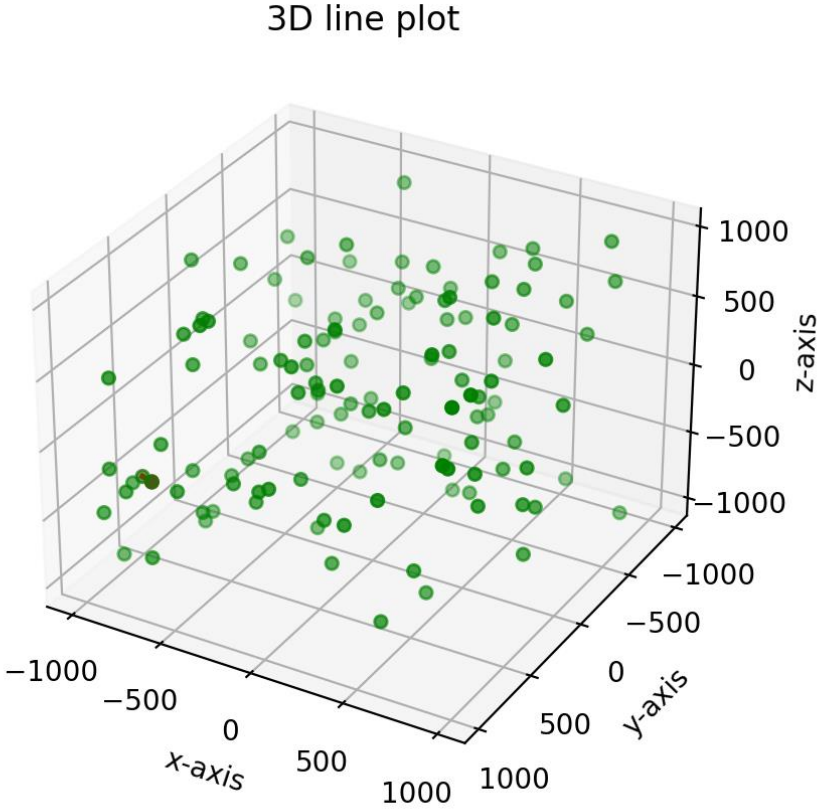
```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 64

titik 1 =  [ 501.93074022 -485.80997182 -409.06726727]
titik 2 =  [ 492.83517636 -482.06622532 -420.32007267]
Distance =  14.945586281509385

titik 1 Brute =  [ 501.93074022 -485.80997182 -409.06726727]
titik 2 Brute =  [ 492.83517636 -482.06622532 -420.32007267]
Distance =  14.945586281509385

Hasil sama
time divide and conquer =  0.003840923309326172
Euclidean distance function call count =  187

time brute force =  0.008748531341552734
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```



c. n=128

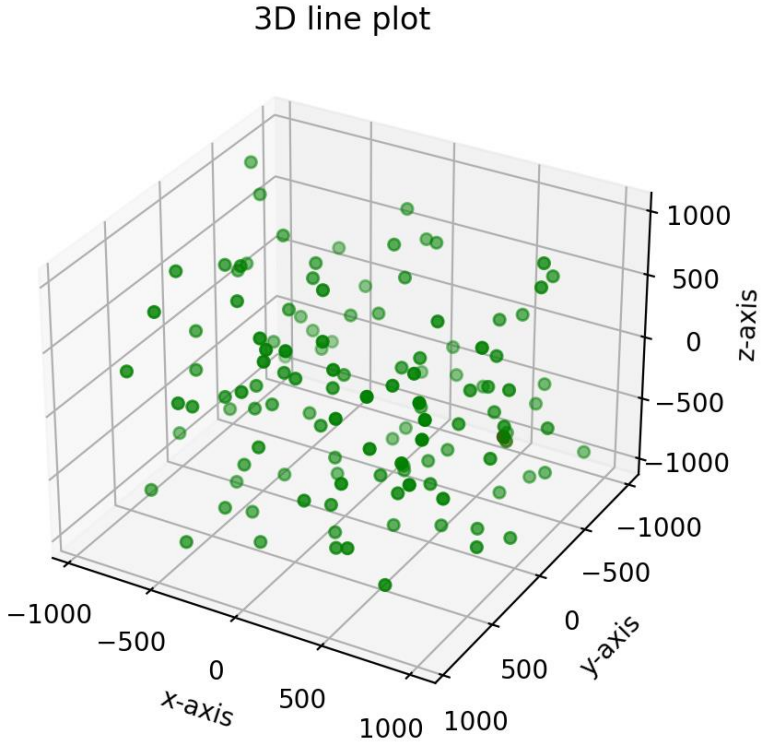
i. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 128

titik 1 =  [-910.40595748  625.37606253 -429.48954223]
titik 2 =  [-855.27413094  629.1863543  -445.18874447]
Distance =  57.44999192276686

titik 1 Brute =  [-910.40595748  625.37606253 -429.48954223]
titik 2 Brute =  [-855.27413094  629.1863543  -445.18874447]
Distance =  57.44999192276686

Hasil sama
time divide and conquer =  0.004327535629272461
Euclidean distance function call count =  456

time brute force =  0.05975461006164551
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```



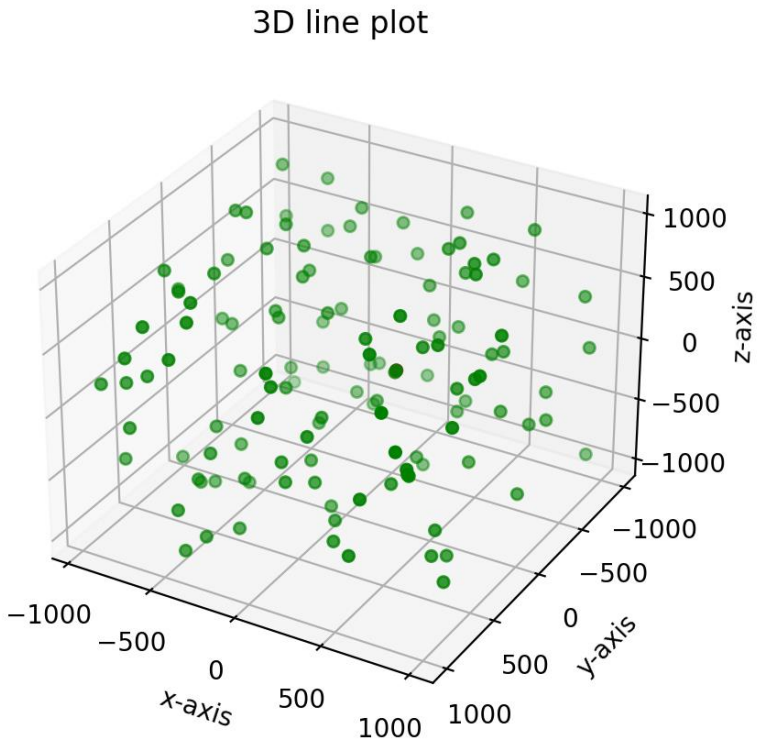
ii. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 128

titik 1 =  [ 680.6683211  -481.73949249 -584.91794958]
titik 2 =  [ 686.78583612 -438.65852161 -515.07421184]
Distance =  82.28937807243051

titik 1 Brute =  [ 680.6683211  -481.73949249 -584.91794958]
titik 2 Brute =  [ 686.78583612 -438.65852161 -515.07421184]
Distance =  82.28937807243051

Hasil sama
time divide and conquer =  0.003449678421020508
Euclidean distance function call count =  416

time brute force =  0.057320594787597656
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```



iii. _

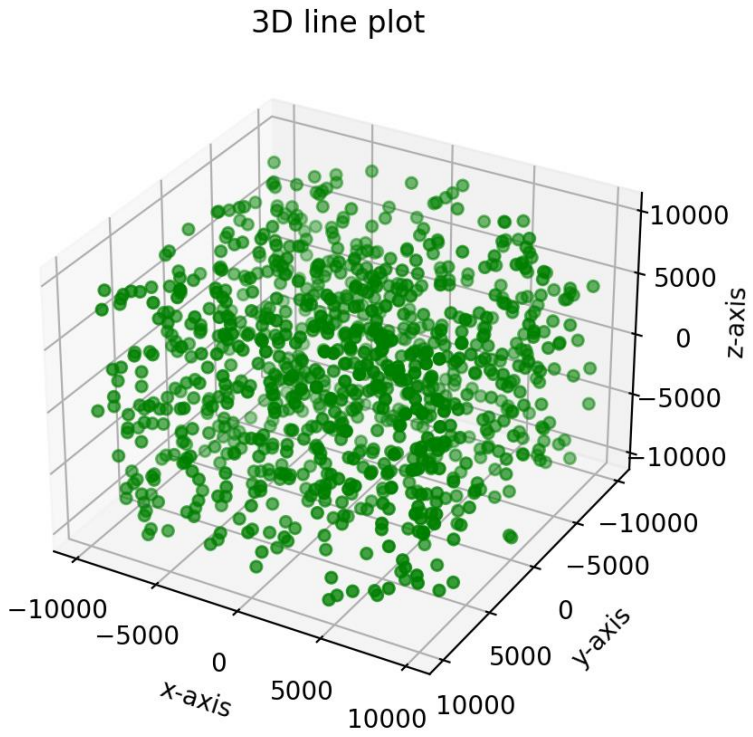
```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 128

titik 1 =  [277.20175352 -31.68500758 137.24892936]
titik 2 =  [273.04644235 -24.76734761 125.6286402 ]
Distance =  14.14749979723888

titik 1 Brute =  [273.04644235 -24.76734761 125.6286402 ]
titik 2 Brute =  [277.20175352 -31.68500758 137.24892936]
Distance =  14.14749979723888

Hasil sama
time divide and conquer =  0.0034456253051757812
Euclidean distance function call count =  326

time brute force =  0.056554317474365234
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```



d. n=1000

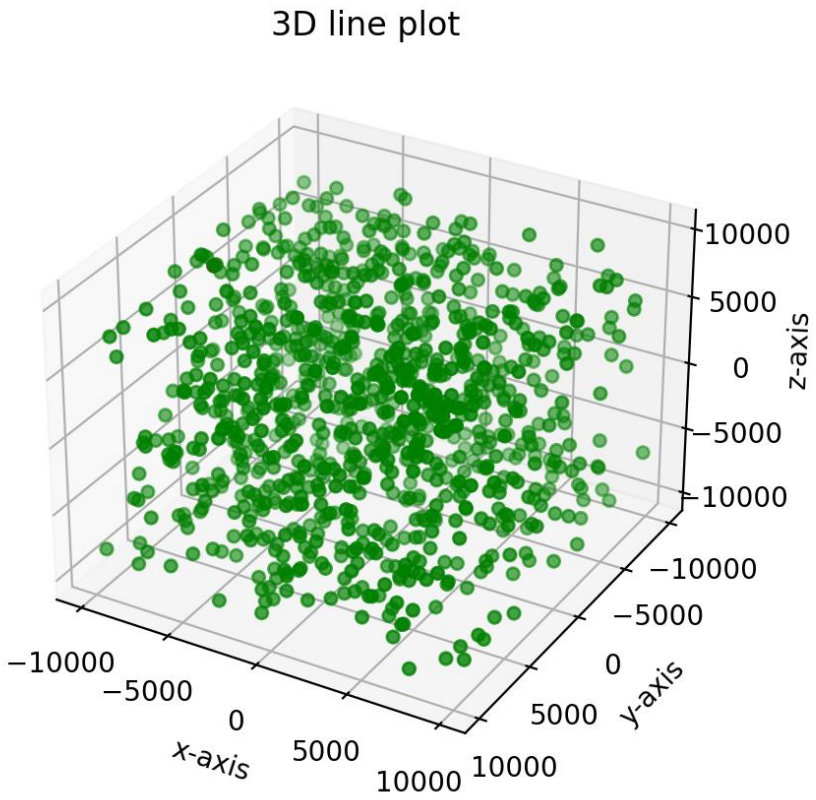
i. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 1000

titik 1 =  [-1800.80447224 -6938.56446367 -2144.87617965]
titik 2 =  [-1868.14666253 -6801.02850968 -2082.18278696]
Distance =  165.47377653459353

titik 1 Brute =  [-1800.80447224 -6938.56446367 -2144.87617965]
titik 2 Brute =  [-1868.14666253 -6801.02850968 -2082.18278696]
Distance =  165.47377653459353

Hasil sama
time divide and conquer =  0.07438230514526367
Euclidean distance function call count =  4092

time brute force =  3.554462432861328
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```



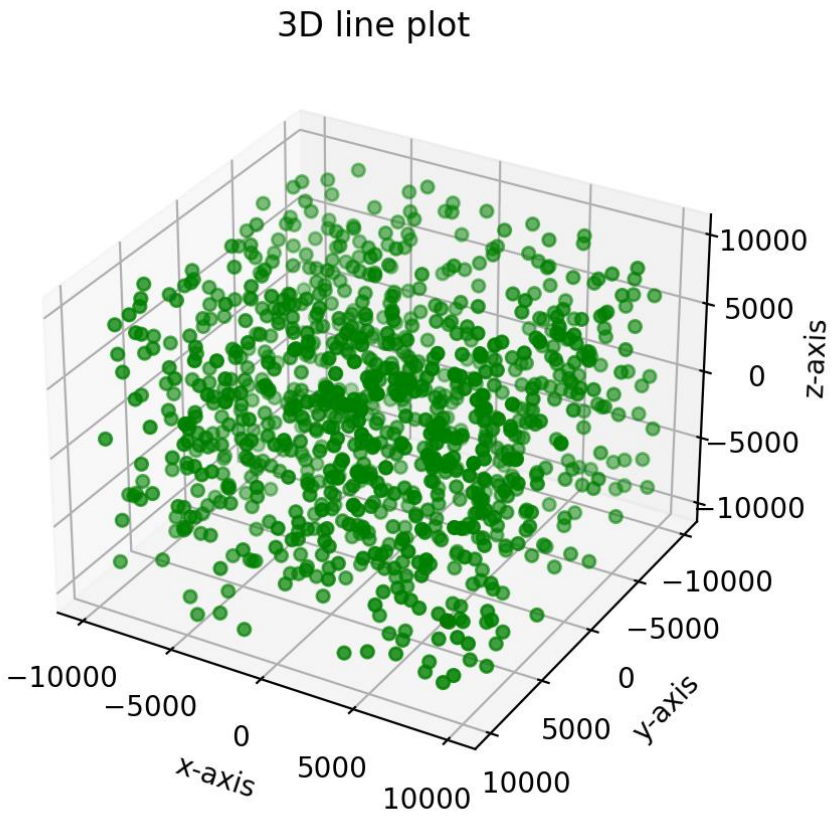
ii. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 1000

titik 1 =  [-7425.82662963  2496.30471999  9291.43872506]
titik 2 =  [-7463.63211076  2654.90359537  9295.8970542 ]
Distance =  163.10344684864492

titik 1 Brute =  [-7425.82662963  2496.30471999  9291.43872506]
titik 2 Brute =  [-7463.63211076  2654.90359537  9295.8970542 ]
Distance =  163.10344684864492

Hasil sama
time divide and conquer =  0.08730030059814453
Euclidean distance function call count =  4296

time brute force =  3.5984110832214355
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```



iii. _

```
Masukan banyak dimensi
dimensi = 3
Masukan banyak titik
n = 1000

titik 1 =  [ 6677.77726325  5378.74810052 -8853.38788086]
titik 2 =  [ 6598.54115223  5433.05722337 -8765.93190172]
Distance =  129.90916210902986

titik 1 Brute =  [ 6598.54115223  5433.05722337 -8765.93190172]
titik 2 Brute =  [ 6677.77726325  5378.74810052 -8853.38788086]
Distance =  129.90916210902986

Hasil sama
time divide and conquer =  0.07455301284790039
Euclidean distance function call count =  4396

time brute force =  3.540508270263672
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```



2. Test case dimensi 6
    a. n =16
        i. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 16

titik 1 =  [ 3897.12260886  1398.60275391 -1072.57674682 -8717.22659782
  1871.45254374  5394.19752578]
titik 2 =  [-1043.3317187   8374.09838658  4750.81261999 -9758.32736947
   804.0369663   6703.873202  ]
Distance =  10531.666972057159

titik 1 Brute =  [-1043.3317187   8374.09838658  4750.81261999 -9758.32736947
   804.0369663   6703.873202  ]
titik 2 Brute =  [ 3897.12260886  1398.60275391 -1072.57674682 -8717.22659782
  1871.45254374  5394.19752578]
Distance =  10531.666972057159

Hasil sama
time divide and conquer =  0.0019888877868652344
Euclidean distance function call count =  159

time brute force =  0.0009989738464355469
Euclidean distance function call count =  121

dieksekusi pada  AMD64
Error plotting, dimension not supported
```

        ii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 16

titik 1 =  [ 9239.203909   -9400.24339782 -2082.3838745  -1167.1930417
   311.565477   -3611.83185545]
titik 2 =  [ 8064.60691984 -7525.34049367 -6039.64398866  2276.5365475
  6938.14762367 -2608.11218574]
Distance =  8793.927689240963

titik 1 Brute =  [ 8064.60691984 -7525.34049367 -6039.64398866  2276.5365475
  6938.14762367 -2608.11218574]
titik 2 Brute =  [ 9239.203909   -9400.24339782 -2082.3838745  -1167.1930417
   311.565477   -3611.83185545]
Distance =  8793.927689240963

Hasil sama
time divide and conquer =  0.002444744110107422
Euclidean distance function call count =  139

time brute force =  0.0009946823120117188
Euclidean distance function call count =  121

dieksekusi pada  AMD64
```

iii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 16

ython
 1 =  [ -650.30621683  2420.30911833  4132.98653788 -9825.46277451
  8809.32138786 -2160.31353545]
titik 2 =  [ -762.643892    4108.30821784  4894.24539102 -9802.92443698
  1262.26889876 -1004.39647421]
Distance =  7857.23419894006

titik 1 Brute =  [ -762.643892    4108.30821784  4894.24539102 -9802.92443698
  1262.26889876 -1004.39647421]
titik 2 Brute =  [ -650.30621683  2420.30911833  4132.98653788 -9825.46277451
  8809.32138786 -2160.31353545]
Distance =  7857.23419894006

Hasil sama
time divide and conquer =  0.003256082534790039
Euclidean distance function call count =  120

time brute force =  0.0015189647674560547
Euclidean distance function call count =  121

dieksekusi pada  AMD64
Error plotting, dimension not supported
```

b. n=64

i. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 64

titik 1 =  [ 3620.73458593 -6825.95350441  5813.20369915 -8872.09415657
 -4382.97164196 -6883.12194683]
titik 2 =  [ 2760.57022043 -4343.26392053  6583.71992136 -8328.03757757
 -2024.76965028 -7135.52430471]
Distance =  3663.078855178164

titik 1 Brute =  [ 2760.57022043 -4343.26392053  6583.71992136 -8328.03757757
 -2024.76965028 -7135.52430471]
titik 2 Brute =  [ 3620.73458593 -6825.95350441  5813.20369915 -8872.09415657
 -4382.97164196 -6883.12194683]
Distance =  3663.078855178164

Hasil sama
time divide and conquer =  0.014343976974487305
Euclidean distance function call count =  1084

time brute force =  0.014577388763427734
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
Error plotting, dimension not supported
```

ii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 64

titik 1 =  [-4218.20047073 -2267.59795154  -432.87812037  4079.0130174
 -2382.98639741 -4333.77937195]
titik 2 =  [-6141.75401623 -1543.25575589  2950.21720745  4899.64937202
   -67.98634776 -5625.15957173]
Distance =  4834.293736706321

titik 1 Brute =  [-6141.75401623 -1543.25575589  2950.21720745  4899.64937202
   -67.98634776 -5625.15957173]
titik 2 Brute =  [-4218.20047073 -2267.59795154  -432.87812037  4079.0130174
 -2382.98639741 -4333.77937195]
Distance =  4834.293736706321

Hasil sama
time divide and conquer =  0.014760732650756836
Euclidean distance function call count =  1096

time brute force =  0.013300895690917969
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```

iii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 64

titik 1 =  [-576.86104156 -981.53034059 -322.19074599  249.59973248  831.35756003
 -126.28971183]
titik 2 =  [-531.9653424  -763.27209431 -187.33078905  478.58959605  680.02256863
 -202.70937889]
Distance =  386.02862149458576

titik 1 Brute =  [-576.86104156 -981.53034059 -322.19074599  249.59973248  831.35756003
 -126.28971183]
titik 2 Brute =  [-531.9653424  -763.27209431 -187.33078905  478.58959605  680.02256863
 -202.70937889]
Distance =  386.02862149458576

Hasil sama
time divide and conquer =  0.013072013854980469
Euclidean distance function call count =  958

time brute force =  0.01371455192565918
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```

c. n=128

i. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 128

titik 1 =  [4395.64933016 7328.07766458 7995.83901126 6121.70240311 4768.51824
811
 1865.87333035]
titik 2 =  [5611.85648984 9231.18893705 6797.86183901 9052.00200758 4868.27287
513
  170.71625062]
Distance =  4243.383786360762

titik 1 Brute =  [5611.85648984 9231.18893705 6797.86183901 9052.00200758 4868
.27287513
  170.71625062]
titik 2 Brute =  [4395.64933016 7328.07766458 7995.83901126 6121.70240311 4768
.51824811
 1865.87333035]
Distance =  4243.383786360762

Hasil sama
time divide and conquer =  0.03733062744140625
Euclidean distance function call count =  3707

time brute force =  0.06062746047973633
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
Error plotting, dimension not supported
```

ii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 128

titik 1 =  [ 2007.14742919 -9347.41683563  4229.86782251 -2889.66378873
 -6411.83410604  9158.7057249 ]
titik 2 =  [-1720.08321887 -8040.45189889  5264.71874891 -1212.43297443
 -4942.81610228  9148.82178594]
Distance =  4652.154004502052

titik 1 Brute =  [ 2007.14742919 -9347.41683563  4229.86782251 -2889.66378873
 -6411.83410604  9158.7057249 ]
titik 2 Brute =  [-1720.08321887 -8040.45189889  5264.71874891 -1212.43297443
 -4942.81610228  9148.82178594]
Distance =  4652.154004502052

Hasil sama
time divide and conquer =  0.04046058654785156
Euclidean distance function call count =  3962

time brute force =  0.05399441719055176
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```

iii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 128

titik 1 =  [-7828.92196781 -2343.19570533 -1392.36514767  3944.22336681
  4264.91314922  4258.1606674 ]
titik 2 =  [-6950.19430143 -2091.31014742  -797.90605298  5861.44747825
  5451.61898131  5952.00393046]
Distance =  3023.5929432411504

titik 1 Brute =  [-7828.92196781 -2343.19570533 -1392.36514767  3944.22336681
  4264.91314922  4258.1606674 ]
titik 2 Brute =  [-6950.19430143 -2091.31014742  -797.90605298  5861.44747825
  5451.61898131  5952.00393046]
Distance =  3023.5929432411504

Hasil sama
time divide and conquer =  0.02899456024169922
Euclidean distance function call count =  2677

time brute force =  0.05946159362792969
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```

d. n=1000

i. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 1000

titik 1 =  [ 7921.62364482  3918.50677413 -3941.97919844  2964.77722626
 -4587.89025063  5677.06565848]
titik 2 =  [ 7266.81916898  4005.64907951 -4202.95167091  2223.23025586
 -5489.60100244  5785.14521506]
Distance =  1370.8116917165523

titik 1 Brute =  [ 7921.62364482  3918.50677413 -3941.97919844  2964.77722626
 -4587.89025063  5677.06565848]
titik 2 Brute =  [ 7266.81916898  4005.64907951 -4202.95167091  2223.23025586
 -5489.60100244  5785.14521506]
Distance =  1370.8116917165523

Hasil sama
time divide and conquer =  0.5255045890808105
Euclidean distance function call count =  58775

time brute force =  3.5322000980377197
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```

ii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 1000

titik 1 =  [ 5831.8519006  -8861.96439446 -5924.46715179  1251.25457896
   770.68067244  9034.67497745]
titik 2 =  [ 5713.14458733 -8461.93341055 -5929.571766     117.17384488
   408.39412132  8444.03132981]
Distance =  1392.9798442296167

titik 1 Brute =  [ 5831.8519006  -8861.96439446 -5924.46715179  1251.25457896
   770.68067244  9034.67497745]
titik 2 Brute =  [ 5713.14458733 -8461.93341055 -5929.571766     117.17384488
   408.39412132  8444.03132981]
Distance =  1392.9798442296167

Hasil sama
time divide and conquer =  0.583726167678833
Euclidean distance function call count =  66392

time brute force =  3.5226852893829346
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```

iii. _

```
Masukan banyak dimensi
dimensi = 6
Masukan banyak titik
n = 1000

titik 1 =  [ 532.62066779 -133.12394253  259.23710882 -753.75045227  125.98960912
   50.3641062 ]
titik 2 =  [ 489.82789304 -111.63529665  407.37971532 -861.1172943    70.59675061
   32.44672037]
Distance =  197.87939658840753

titik 1 Brute =  [ 489.82789304 -111.63529665  407.37971532 -861.1172943    70.59675061
   32.44672037]
titik 2 Brute =  [ 532.62066779 -133.12394253  259.23710882 -753.75045227  125.98960912
   50.3641062 ]
Distance =  197.87939658840753

Hasil sama
time divide and conquer =  0.8453354835510254
Euclidean distance function call count =  80451

time brute force =  3.703979969024658
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
Error plotting, dimension not supported
```

3. Test case dimensi 7
    a. n =16
        i. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 16

titik 1 =  [-7624.68624696  1817.83636142 -7144.58985059  3300.95339055
 -8652.01444773 -4123.1473229   8214.10952666]
titik 2 =  [-7153.19489399  2473.13460946 -3586.6901904   9369.5625954
 -4575.79312209 -4683.95892033  9724.10281171]
Distance =  8327.578112945941

titik 1 Brute =  [-7624.68624696  1817.83636142 -7144.58985059  3300.95339055
 -8652.01444773 -4123.1473229   8214.10952666]
titik 2 Brute =  [-7153.19489399  2473.13460946 -3586.6901904   9369.5625954
 -4575.79312209 -4683.95892033  9724.10281171]
Distance =  8327.578112945941

Hasil sama
time divide and conquer =  0.002378225326538086
Euclidean distance function call count =  133

time brute force =  0.0012485980987548828
Euclidean distance function call count =  121

dieksekusi pada  AMD64
```

        ii. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 16

titik 1 =  [ 7915.59210112 -1124.97675378   617.5715767  -4118.5668375
  6306.97514416   911.41309114   838.15481292]
titik 2 =  [ 6976.20866213  -655.4273317   4497.06455735 -5499.22100859
  6661.75541792 -1487.42915658  1596.08366572]
Distance =  4951.197728856285

titik 1 Brute =  [ 7915.59210112 -1124.97675378   617.5715767  -4118.5668375
  6306.97514416   911.41309114   838.15481292]
titik 2 Brute =  [ 6976.20866213  -655.4273317   4497.06455735 -5499.22100859
  6661.75541792 -1487.42915658  1596.08366572]
Distance =  4951.197728856285

Hasil sama
time divide and conquer =  0.0
Euclidean distance function call count =  100

time brute force =  0.0
Euclidean distance function call count =  121

dieksekusi pada  AMD64
```

iii. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 16

titik 1 =  [ 9131.38045715  5483.76104435 -8335.15498251  2541.31754247
 -1246.55001685 -1324.85765229  7097.76141562]
titik 2 =  [ 3814.24101699  5996.55198555 -4589.19584414 -2825.78746434
   -87.15404919   616.02860287  4527.28443632]
Distance =  9115.456256396305

titik 1 Brute =  [ 3814.24101699  5996.55198555 -4589.19584414 -2825.78746434
   -87.15404919   616.02860287  4527.28443632]
titik 2 Brute =  [ 9131.38045715  5483.76104435 -8335.15498251  2541.31754247
 -1246.55001685 -1324.85765229  7097.76141562]
Distance =  9115.456256396305

Hasil sama
time divide and conquer =  0.0
Euclidean distance function call count =  142

time brute force =  0.0030732154846191406
Euclidean distance function call count =  121

dieksekusi pada  AMD64
```

b. n=64

i. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 64

titik 1 =  [ 5876.15577109 -7492.00384578 -5137.55929989 -7781.16202489
  6035.80308215  5734.4397826  -4123.57230346]
titik 2 =  [ 8564.27487003 -6941.27263658 -5297.65781528 -9538.5369969
  4038.37153913  5486.15629582 -6463.60599377]
Distance =  4491.149364293209

titik 1 Brute =  [ 8564.27487003 -6941.27263658 -5297.65781528 -9538.5369969
  4038.37153913  5486.15629582 -6463.60599377]
titik 2 Brute =  [ 5876.15577109 -7492.00384578 -5137.55929989 -7781.16202489
  6035.80308215  5734.4397826  -4123.57230346]
Distance =  4491.149364293209

Hasil sama
time divide and conquer =  0.011312723159790039
Euclidean distance function call count =  1151

time brute force =  0.008622407913208008
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```

ii. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 64

titik 1 =  [ 5639.52719637 -6512.79222195 -5989.45605807  1616.55439394
 -8179.77026362 -5718.76184021  3456.88273182]
titik 2 =  [ 7654.43827872 -5784.99769171 -9861.50290377   855.94093486
 -5778.02969796 -6552.51140639  4768.70616443]
Distance =  5324.021784295576

titik 1 Brute =  [ 5639.52719637 -6512.79222195 -5989.45605807  1616.55439394
 -8179.77026362 -5718.76184021  3456.88273182]
titik 2 Brute =  [ 7654.43827872 -5784.99769171 -9861.50290377   855.94093486
 -5778.02969796 -6552.51140639  4768.70616443]
Distance =  5324.021784295576

Hasil sama
time divide and conquer =  0.024074077606201172
Euclidean distance function call count =  1384

time brute force =  0.017099618911743164
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```

iii. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 64

titik 1 =  [ 3219.90506803   597.84468698 -4506.4425129  -6039.60001175
  9661.1869223  -8001.49787065 -9747.60244618]
titik 2 =  [ 2121.65620884  4043.30247448 -6444.40596921 -8792.73060237
  6667.26757039 -5815.32527425 -9589.23392931]
Distance =  6179.056936402675

titik 1 Brute =  [ 2121.65620884  4043.30247448 -6444.40596921 -8792.73060237
  6667.26757039 -5815.32527425 -9589.23392931]
titik 2 Brute =  [ 3219.90506803   597.84468698 -4506.4425129  -6039.60001175
  9661.1869223  -8001.49787065 -9747.60244618]
Distance =  6179.056936402675

Hasil sama
time divide and conquer =  0.010502815246582031
Euclidean distance function call count =  1516

time brute force =  0.016690492630004883
Euclidean distance function call count =  2017

dieksekusi pada  AMD64
```

c. n=128

i. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 128

titik 1 =  [-5138.12989063  3408.82253169  2272.37635695 -3131.57137185
 -6822.86936843  5128.53820545  7712.56185139]
titik 2 =  [-1823.88168312  5312.08099204  2150.21421442 -3186.82613536
 -4643.03298858  5646.72433984  7461.2971439 ]
Distance =  4439.36347822584

titik 1 Brute =  [-1823.88168312  5312.08099204  2150.21421442 -3186.82613536
 -4643.03298858  5646.72433984  7461.2971439 ]
titik 2 Brute =  [-5138.12989063  3408.82253169  2272.37635695 -3131.57137185
 -6822.86936843  5128.53820545  7712.56185139]
Distance =  4439.36347822584

Hasil sama
time divide and conquer =  0.042154550552368164
Euclidean distance function call count =  4604

time brute force =  0.05674934387207031
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```

ii. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 128

titik 1 =  [-7551.15125882  2738.22790574  5030.57273363 -9092.49947575
 -3030.70708774  3582.56026388 -1497.90644139]
titik 2 =  [-8539.84085473  2872.37266629  3045.87631211 -7430.24239743
 -3750.16100216  6286.94724341 -1799.99259027]
Distance =  3952.2397858944573

titik 1 Brute =  [-7551.15125882  2738.22790574  5030.57273363 -9092.49947575
 -3030.70708774  3582.56026388 -1497.90644139]
titik 2 Brute =  [-8539.84085473  2872.37266629  3045.87631211 -7430.24239743
 -3750.16100216  6286.94724341 -1799.99259027]
Distance =  3952.2397858944573

Hasil sama
time divide and conquer =  0.039957523345947266
Euclidean distance function call count =  3669

time brute force =  0.05583477020263672
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```

iii. _


```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 128

titik 1 =  [-7551.15125882  2738.22790574  5030.57273363 -9092.49947575
 -3030.70708774  3582.56026388 -1497.90644139]
titik 2 =  [-8539.84085473  2872.37266629  3045.87631211 -7430.24239743
 -3750.16100216  6286.94724341 -1799.99259027]
Distance =  3952.2397858944573

titik 1 Brute =  [-7551.15125882  2738.22790574  5030.57273363 -9092.49947575
 -3030.70708774  3582.56026388 -1497.90644139]
titik 2 Brute =  [-8539.84085473  2872.37266629  3045.87631211 -7430.24239743
 -3750.16100216  6286.94724341 -1799.99259027]
Distance =  3952.2397858944573

Hasil sama
time divide and conquer =  0.039957523345947266
Euclidean distance function call count =  3669

time brute force =  0.05583477020263672
Euclidean distance function call count =  8129

dieksekusi pada  AMD64
```


d. n=1000

i. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 1000

titik 1 =  [ 8803.84896935 -1532.86439437  2454.93915647   788.63636327
 -8168.36312565  2075.83701837  3130.04856345]
titik 2 =  [ 9191.80897209    26.76716065  3278.68257293  1128.48852632
 -8508.20635499  2758.85807777  2348.22683299]
Distance =  2137.8196056662186

titik 1 Brute =  [ 9191.80897209    26.76716065  3278.68257293  1128.48852632
 -8508.20635499  2758.85807777  2348.22683299]
titik 2 Brute =  [ 8803.84896935 -1532.86439437  2454.93915647   788.63636327
 -8168.36312565  2075.83701837  3130.04856345]
Distance =  2137.8196056662186

Hasil sama
time divide and conquer =  0.9051914215087891
Euclidean distance function call count =  105021

time brute force =  3.6308434009552
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```

ii. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 1000

titik 1 =  [-6258.59402209  4369.5885266  -4787.27789586  7362.20805395
 -3140.86683976  5044.99641822 -3299.10941916]
titik 2 =  [-7891.60926485  5004.00154286 -4201.67745935  7500.76545686
 -3520.84114955  6288.49151857 -2338.24312798]
Distance =  2458.7129262391068

titik 1 Brute =  [-7891.60926485  5004.00154286 -4201.67745935  7500.76545686
 -3520.84114955  6288.49151857 -2338.24312798]
titik 2 Brute =  [-6258.59402209  4369.5885266  -4787.27789586  7362.20805395
 -3140.86683976  5044.99641822 -3299.10941916]
Distance =  2458.7129262391068

Hasil sama
time divide and conquer =  0.893986701965332
Euclidean distance function call count =  106295

time brute force =  3.4935665130615234
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```

iii. _

```
Masukan banyak dimensi
dimensi = 7
Masukan banyak titik
n = 1000

titik 1 =  [-5857.99159408  8014.98995012 -5194.81610749 -5969.70572873
  2934.69134875   358.58352667  4988.5581424 ]
titik 2 =  [-5145.62456494  9886.1631408  -4718.53856848 -6882.00828915
  3745.56729188  1548.48718365  4104.5074671 ]
Distance =  2814.751899225931

titik 1 Brute =  [-5145.62456494  9886.1631408  -4718.53856848 -6882.00828915
  3745.56729188  1548.48718365  4104.5074671 ]
titik 2 Brute =  [-5857.99159408  8014.98995012 -5194.81610749 -5969.70572873
  2934.69134875   358.58352667  4988.5581424 ]
Distance =  2814.751899225931

Hasil sama
time divide and conquer =  1.0197842121124268
Euclidean distance function call count =  121111

time brute force =  3.651024103164673
Euclidean distance function call count =  499501

dieksekusi pada  AMD64
```

VII. Link Repositori

Link: Onyxcodeotto/TUCIL2_13521053_13521043 (github.com)

VIII. Referensi

Daftar Referensi

1. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2006-2007/Algoritma%20Divide%20and%20Conquer%20(Bagian%202).pdf
2. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Divide-and-Conquer-(2021)-Bagian1.pdf
3. https://numpy.org
4. https://sites.cs.ucsb.edu/~suri/cs235/ClosestPair.pdf
5. https://www.geeksforgeeks.org/