LAPORAN TUGAS KECIL 2 IF2211 STRATEGI ALGORITMA

MENCARI PASANGAN TITIK TERDEKAT 3D DENGAN ALGORITMA DIVIDE AND CONQUER

Oleh:

Muhammad Raihan Iqbal 13518134

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2023

Bab 1
Algoritma Divide and Conquer

Berikut adalah langkah-langkah algoritma *Divide and Conquer* yang digunakan untuk mencari pasangan titik terdekat ini:

1. Melakukan pengurutan list titik-titik yang telah di-*generate*.
2. Melakukan pengecekan jumlah titik secara rekursif:
    a. Apabila jumlah titik kurang dari atau sama dengan tiga, maka akan dilakukan perhitungan secara *brute force*.
    b. Apabila jumlah titik lebih dari tiga, maka akan dilakukan pembagian list titik-titik menjadi dua bagian, lalu dilakukan pengecekan jumlah titik lagi.
3. Menghitung jarak dari titik-titik yang berada pada daerah abu-abu.
4. Melakukan perbandingan hasil antara titik yang berada di dalam dan di luar daerah abu-abu, lalu mengambil nilai terkecil dari perbandingan tersebut.

Bab 2

Kode Program

Berikut adalah kode program yang dibuat:

```
import random
import math

from datetime import datetime
from operator import itemgetter

# sort list of points
def sortListOfPoints(points):
  sortedPoints = sorted(points, key=itemgetter(0,1))
  return sortedPoints

# calculate euclidean distance
def euclidean_distance(p1, p2):
  return math.sqrt(pow((p1[0] - p2[0]), 2) + pow((p1[1] - p2[1]), 2) +
pow((p1[2] - p2[2]), 2))

# brute force
def closestPairBruteForce(thePoints):
  distance = float('inf')
  p = []
  count = 0
  for i in range(len(thePoints)-1):
    for j in range(i+1, len(thePoints)):
      if (thePoints[i] != thePoints[j]):
        temp_distance = euclidean_distance(thePoints[i], thePoints[j])
        count += 1
        if (temp_distance < distance):
          distance = temp_distance
          p.clear()
          p.append(thePoints[i])
          p.append(thePoints[j])

  return distance, p[0], p[1], count

# divide and conquer
def closestPairDnC(thePoints):
  if (len(thePoints) <= 3):
    return closestPairBruteForce(thePoints)
  else:
    ptsCount = len(thePoints)
    pts1 = []
    pts2 = []
    midPts = thePoints[int(ptsCount//2)]
    min_distance = 0
    for i in range(int(ptsCount//2)):
      pts1.append(thePoints[i])
    for i in range(int(ptsCount//2), ptsCount):
      pts2.append(thePoints[i])
    d1, p11, p12, c1 = closestPairDnC(pts1)
```

```
    d2, p21, p22, c2 = closestPairDnC(pts2)
    if (d1 < d2):
      min_distance = d1
    else:
      min_distance = d2

    greyArea = []
    for i in range(len(thePoints)):
      if ((thePoints[i] != midPts[0]) and (abs(thePoints[i][0] -
midPts[0]) < min_distance) and (abs(thePoints[i][1] - midPts[1]) <
min_distance)):
        greyArea.append(thePoints[i])

    gDist, gp1, gp2, cg = closestPairGreyArea(greyArea, min_distance)
    if (gDist < min_distance):
      return gDist, gp1, gp2, (c1+c2+cg)
    else:
      if (min_distance == d1):
        return d1, p11, p12, (c1+c2+cg)
      else:
        return d2, p21, p22, (c1+c2+cg)

# points in grey area
def closestPairGreyArea(thePoints, dist):
  distance = dist
  p = []
  count = 0
  for i in range(len(thePoints)-1):
    for j in range(i+1, len(thePoints)):
      temp_distance = euclidean_distance(thePoints[i], thePoints[j])
      count += 1
      if (temp_distance < distance):
        distance = temp_distance
        p.clear()
        p.append(thePoints[i])
        p.append(thePoints[j])

  if (len(p) == 0):
    return distance, None, None, count
  else:
    return distance, p[0], p[1], count


### MAIN PROGRAM ###
if __name__ == "__main__":
  # input number of points by user
  n = int(input("Jumlah titik yang ingin dibangkitkan: "))

  # generate points
  points = []
  for i in range(n):
    point = []
    for j in range(3):
      point.append(random.randint(0, 100))
    points.append(point)

  print("kumpulan titik yang dihasilkan:")
  print(points)
```

```
  print("")
  sortedPoints = sortListOfPoints(points)
  print("kumpulan titik yang telah diurutkan:")
  print(sortedPoints)
  print("")

  # find closest pair
  startTime = datetime.now()
  distance, p1, p2, count = closestPairDnC(points)
  finishTime = datetime.now()
  processingTime = finishTime - startTime

  # show result
  print("HASIL PENCARIAN:")
  print("Jarak terdekat: " + str(distance))
  print("Titik terdekat pertama: " + str(p1[0]) + " " + str(p1[1]) + "
" + str(p1[2]))
  print("Titik terdekat kedua: " + str(p2[0]) + " " + str(p2[1]) + " "
+ str(p2[2]))
  print("Banyaknya operasi perhitungan rumus Euclidian: " + str(count))
  print("Waktu pemrosesan: " + str(processingTime.total_seconds()))
  print("(pemrosesan dilakukan menggunakan laptop Lenovo ideapad gaming
3)")
```

Bab 3
Contoh Masukan dan Luaran

1. Masukan n = 16

```
D:\Semester 10\IF2211 Strategi Algoritma\Tugas\Tucil 2\Tucil2_13518134\src>python main.py
Jumlah titik yang ingin dibangkitkan: 16
kumpulan titik yang dihasilkan:
[[90, 1, 37], [92, 97, 62], [23, 14, 48], [91, 19, 47], [69, 60, 67], [99, 97, 56], [7, 61, 21], [81, 67, 34], [25, 95, 83], [65, 88
, 29], [80, 74, 98], [3, 5, 84], [80, 60, 8], [69, 79, 99], [72, 68, 29], [67, 60, 84]]

kumpulan titik yang telah diurutkan:
[[3, 5, 84], [7, 61, 21], [23, 14, 48], [25, 95, 83], [65, 88, 29], [67, 60, 84], [69, 60, 67], [69, 79, 99], [72, 68, 29], [80, 60,
 8], [80, 74, 98], [81, 67, 34], [90, 1, 37], [91, 19, 47], [92, 97, 62], [99, 97, 56]]

HASIL PENCARIAN:
Jarak terdekat: 12.124355652982141
Titik terdekat pertama: 80 74 98
Titik terdekat kedua: 69 79 99
Banyaknya operasi perhitungan rumus Euclidian: 31
Waktu pemrosesan: 0.0
(pemrosesan dilakukan menggunakan laptop Lenovo ideapad gaming 3)
```

2. Masukan n = 64

```
D:\Semester 10\IF2211 Strategi Algoritma\Tugas\Tucil 2\Tucil2_13518134\src>python main.py
Jumlah titik yang ingin dibangkitkan: 64
kumpulan titik yang dihasilkan:
[[43, 52, 92], [21, 56, 40], [7, 26, 39], [7, 78, 90], [10, 16, 27], [53, 65, 81], [26, 34, 78], [49, 93, 21], [3, 72, 32], [25, 68,
 32], [43, 99, 37], [97, 23, 43], [64, 63, 50], [29, 68, 8], [7, 24, 77], [90, 100, 90], [78, 53, 73], [14, 76, 63], [14, 66, 75], [
71, 90, 82], [77, 88, 8], [61, 35, 80], [85, 13, 45], [45, 71, 33], [71, 24, 33], [80, 37, 47], [89, 45, 37], [3, 40, 74], [39, 92,
93], [89, 62, 6], [56, 18, 15], [40, 85, 0], [41, 62, 41], [12, 82, 79], [11, 85, 27], [41, 78, 62], [52, 87, 72], [32, 64, 46], [94
, 51, 0], [76, 36, 96], [41, 35, 44], [40, 25, 88], [42, 89, 9], [32, 32, 68], [93, 84, 70], [32, 16, 39], [100, 23, 66], [76, 95, 1
7], [97, 10, 77], [15, 32, 71], [14, 66, 49], [86, 100, 17], [68, 33, 80], [61, 74, 39], [84, 2, 41], [0, 33, 99], [55, 57, 34], [23
, 18, 23], [3, 1, 92], [50, 86, 1], [69, 12, 23], [52, 87, 53], [12, 59, 58], [91, 78, 41]]

kumpulan titik yang telah diurutkan:
[[0, 33, 99], [3, 1, 92], [3, 40, 74], [3, 72, 32], [7, 24, 77], [7, 26, 39], [7, 78, 90], [10, 16, 27], [11, 85, 27], [12, 59, 58],
 [12, 82, 79], [14, 66, 75], [14, 66, 49], [14, 76, 63], [15, 32, 71], [21, 56, 40], [23, 18, 23], [25, 68, 32], [26, 34, 78], [29,
68, 8], [32, 16, 39], [32, 32, 68], [32, 64, 46], [39, 92, 93], [40, 25, 88], [40, 85, 0], [41, 35, 44], [41, 62, 41], [41, 78, 62],
 [42, 89, 9], [43, 52, 92], [43, 99, 37], [45, 71, 33], [49, 93, 21], [50, 86, 1], [52, 87, 72], [52, 87, 53], [53, 65, 81], [55, 57
, 34], [56, 18, 15], [61, 35, 80], [61, 74, 39], [64, 63, 50], [68, 33, 80], [69, 12, 23], [71, 24, 33], [71, 90, 82], [76, 36, 96],
 [76, 95, 17], [77, 88, 8], [78, 53, 73], [80, 37, 47], [84, 2, 41], [85, 13, 45], [86, 100, 17], [89, 45, 37], [89, 62, 6], [90, 10
0, 90], [91, 78, 41], [93, 84, 70], [94, 51, 0], [97, 10, 77], [97, 23, 43], [100, 23, 66]]

HASIL PENCARIAN:
Jarak terdekat: 10.488088481701515
Titik terdekat pertama: 41 62 41
Titik terdekat kedua: 32 64 46
Banyaknya operasi perhitungan rumus Euclidian: 140
Waktu pemrosesan: 0.001
(pemrosesan dilakukan menggunakan laptop Lenovo ideapad gaming 3)
```

3. Masukan n = 128

```
D:\Semester 10\IF2211 Strategi Algoritma\Tugas\Tucil 2\Tucil2_13518134\src>python main.py
Jumlah titik yang ingin dibangkitkan: 128
kumpulan titik yang dihasilkan:
[[56, 6, 16], [68, 4, 64], [69, 36, 94], [34, 66, 90], [8, 16, 80], [26, 41, 98], [20, 45, 74], [65, 40, 85], [59, 92, 7], [0, 23, 8
], [10, 100, 73], [89, 24, 51], [15, 40, 69], [80, 84, 21], [97, 21, 25], [98, 80, 17], [43, 65, 5], [20, 14, 21], [12, 63, 60], [22
, 64, 30], [78, 31, 59], [52, 66, 45], [20, 97, 81], [69, 50, 35], [11, 13, 79], [78, 58, 86], [71, 63, 74], [96, 60, 99], [8, 61, 4
2], [24, 4, 16], [40, 53, 48], [47, 58, 68], [35, 45, 83], [15, 25, 32], [41, 32, 13], [33, 88, 85], [3, 47, 50], [38, 38, 56], [70,
 30, 10], [62, 14, 11], [21, 97, 42], [96, 78, 3], [91, 61, 55], [73, 53, 40], [65, 62, 85], [2, 89, 93], [74, 23, 10], [9, 49, 11],
 [75, 86, 46], [75, 91, 54], [41, 15, 7], [76, 57, 96], [39, 32, 90], [0, 40, 52], [81, 47, 20], [99, 93, 15], [30, 94, 74], [85, 16
, 37], [17, 27, 99], [92, 35, 38], [42, 21, 57], [59, 57, 63], [22, 90, 35], [52, 80, 65], [58, 63, 25], [77, 69, 19], [40, 42, 35],
 [98, 55, 98], [11, 24, 72], [26, 4, 54], [94, 74, 53], [67, 81, 30], [32, 79, 53], [27, 59, 96], [88, 74, 65], [70, 91, 13], [9, 56
, 42], [65, 17, 97], [8, 19, 50], [68, 87, 22], [99, 37, 84], [28, 95, 56], [83, 97, 0], [67, 79, 95], [54, 92, 36], [57, 60, 68], [
81, 81, 31], [42, 96, 72], [70, 1, 98], [97, 15, 35], [2, 93, 68], [13, 30, 60], [97, 0, 0], [38, 62, 14], [96, 76, 8], [92, 78, 54]
, [69, 61, 65], [15, 17, 44], [48, 42, 50], [4, 49, 0], [7, 62, 17], [16, 97, 90], [11, 18, 85], [12, 8, 76], [96, 45, 75], [22, 29,
 86], [99, 88, 10], [5, 77, 73], [92, 26, 100], [18, 35, 87], [28, 86, 32], [73, 78, 37], [89, 52, 42], [0, 38, 44], [23, 23, 32], [
38, 71, 60], [75, 86, 18], [83, 5, 70], [48, 55, 48], [79, 9, 44], [65, 11, 54], [25, 94, 13], [39, 84, 53], [63, 0, 31], [13, 14, 8
7], [62, 13, 40], [38, 78, 66], [76, 8, 85]]

kumpulan titik yang telah diurutkan:
```

```
kumpulan titik yang telah diurutkan:
[[0, 23, 8], [0, 38, 44], [0, 40, 52], [2, 89, 93], [2, 93, 68], [3, 47, 50], [4, 49, 0], [5, 77, 73], [7, 62, 17], [8, 16, 80], [8,
 19, 50], [8, 61, 42], [9, 49, 11], [9, 56, 42], [10, 100, 73], [11, 13, 79], [11, 18, 85], [11, 24, 72], [12, 8, 76], [12, 63, 60],
 [13, 14, 87], [13, 30, 60], [15, 17, 44], [15, 25, 32], [15, 40, 69], [16, 97, 90], [17, 27, 99], [18, 35, 87], [20, 14, 21], [20,
45, 74], [20, 97, 81], [21, 97, 42], [22, 29, 86], [22, 64, 30], [22, 90, 35], [23, 23, 32], [24, 4, 16], [25, 94, 13], [26, 4, 54],
 [26, 41, 98], [27, 59, 96], [28, 86, 32], [28, 95, 56], [30, 94, 74], [32, 79, 53], [33, 88, 85], [34, 66, 90], [35, 45, 83], [38,
38, 56], [38, 62, 14], [38, 71, 60], [38, 78, 66], [39, 32, 90], [39, 84, 53], [40, 42, 35], [40, 53, 48], [41, 15, 7], [41, 32, 13]
, [42, 21, 57], [42, 96, 72], [43, 65, 5], [47, 58, 68], [48, 42, 50], [48, 55, 48], [52, 66, 45], [52, 80, 65], [54, 92, 36], [56,
6, 16], [57, 60, 68], [58, 63, 25], [59, 57, 63], [59, 92, 7], [62, 13, 40], [62, 14, 11], [63, 0, 31], [65, 11, 54], [65, 17, 97],
[65, 40, 85], [65, 62, 85], [67, 79, 95], [67, 81, 30], [68, 4, 64], [68, 87, 22], [69, 36, 94], [69, 50, 35], [69, 61, 65], [70, 1,
 98], [70, 30, 10], [70, 91, 13], [71, 63, 74], [73, 53, 40], [73, 78, 37], [74, 23, 10], [75, 86, 46], [75, 86, 18], [75, 91, 54],
[76, 8, 85], [76, 57, 96], [77, 69, 19], [78, 31, 59], [78, 58, 86], [79, 9, 44], [80, 84, 21], [81, 47, 20], [81, 81, 31], [83, 5,
70], [83, 97, 0], [85, 16, 37], [88, 74, 65], [89, 24, 51], [89, 52, 42], [91, 61, 55], [92, 26, 100], [92, 35, 38], [92, 78, 54], [
94, 74, 53], [96, 45, 75], [96, 60, 99], [96, 76, 8], [96, 78, 3], [97, 0, 0], [97, 15, 35], [97, 21, 25], [98, 55, 98], [98, 80, 17
], [99, 37, 84], [99, 88, 10], [99, 93, 15]]

HASIL PENCARIAN:
Jarak terdekat: 6.164414002968976
Titik terdekat pertama: 59 57 63
Titik terdekat kedua: 57 60 68
Banyaknya operasi perhitungan rumus Euclidian: 248
Waktu pemrosesan: 0.000989
(pemrosesan dilakukan menggunakan laptop Lenovo ideapad gaming 3)
```

4. **Masukan n = 1000 (tidak menampilkan titik-titik yang dihasilkan untuk menyederhanakan tampilan)**

```
D:\Semester 10\IF2211 Strategi Algoritma\Tugas\Tucil 2\Tucil2_13518134\src>python main.py
Jumlah titik yang ingin dibangkitkan: 1000
kumpulan titik yang dihasilkan:

kumpulan titik yang telah diurutkan:

HASIL PENCARIAN:
Jarak terdekat: 2.0
Titik terdekat pertama: 99 78 55
Titik terdekat kedua: 97 78 55
Banyaknya operasi perhitungan rumus Euclidian: 2085
Waktu pemrosesan: 0.006015
(pemrosesan dilakukan menggunakan laptop Lenovo ideapad gaming 3)
```

Bab 4
Pranala Github

Berikut adalah tautan repositori dari program yang telah dibuat:
https://github.com/raihaniqbal24/Tucil2_13518134

Checklist Kelengkapan

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		✔