

	<div>LEMBAR JAWABAN</div> <div>Semester Ganjil/Gesap Tahun Ajaran 2024/2025</div>	 <div>Fakultas Informatika School of Computing Telkom University</div>																		
Maaf atas keterlambatan dalam pengumpulan tugas ini. Namun, saya dapat memastikan bahwa saya telah mengerjakan tugas ini dengan sungguh-sungguh dan penuh perhatian. Terima kasih atas pengertiannya.																				
<table><tr><td>Nama Mahasiswa</td><td>:</td><td>Tirta Aditya</td></tr><tr><td>NIM</td><td>:</td><td>1203220057</td></tr><tr><td>Prodi</td><td>:</td><td>INFORMATIKA</td></tr><tr><td>Mata Kuliah</td><td>:</td><td>IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK 12. INTRODUCTION & UNIT TESTING " Pengklasifikasi Segitiga dengan Unit Test "</td></tr><tr><td>Link GitHub</td><td>:</td><td>https://github.com/Tirta7/12.-INTRODUCTION-UNIT-TESTING.git</td></tr><tr><td>Tanggal Pengerjaan</td><td>:</td><td>dd/mm/yyyy</td></tr></table>		Nama Mahasiswa	:	Tirta Aditya	NIM	:	1203220057	Prodi	:	INFORMATIKA	Mata Kuliah	:	IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK 12. INTRODUCTION & UNIT TESTING " Pengklasifikasi Segitiga dengan Unit Test "	Link GitHub	:	https://github.com/Tirta7/12.-INTRODUCTION-UNIT-TESTING.git	Tanggal Pengerjaan	:	dd/mm/yyyy	<div>Tanda Tangan Mahasiswa</div> <div></div> <div>TIRTA ADITYA</div>
Nama Mahasiswa	:	Tirta Aditya																		
NIM	:	1203220057																		
Prodi	:	INFORMATIKA																		
Mata Kuliah	:	IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK 12. INTRODUCTION & UNIT TESTING " Pengklasifikasi Segitiga dengan Unit Test "																		
Link GitHub	:	https://github.com/Tirta7/12.-INTRODUCTION-UNIT-TESTING.git																		
Tanggal Pengerjaan	:	dd/mm/yyyy																		

1. Analisis Flow Graph:
- [Mulai] → (1) Cek nilai negatif/nol

↓

(2) Cek ketidaksamaan segitiga

↓

(3) Cek segitiga sama sisi

↓

(4) Cek segitiga siku-siku

↓

(5) Cek segitiga sama kaki

↓

(6) Kembalikan segitiga bebas

↓

[Selesai]
2. Kompleksitas Siklomatik:
- Edges (E) = 7 (jumlah garis penghubung)
 - Nodes (N) = 7 (jumlah titik/node)
 - Kompleksitas Siklomatik = $E - N + 2 = 7 - 7 + 2 = 2$
 - Ini menunjukkan kita membutuhkan minimal 2 kasus uji untuk mencakup semua jalur
3. Kumpulan Jalur Independen:
4. Jalur nilai negatif/nol
5. Jalur pelanggaran ketidaksamaan segitiga
6. Jalur segitiga sama sisi
7. Jalur segitiga siku-siku
8. Jalur segitiga sama kaki
9. Jalur segitiga bebas
10. Rancangan Kasus Uji: Saya telah mengimplementasikan kasus uji untuk setiap jalur:
- test_nilai_negatif: Menguji nilai input negatif
 - test_nilai_nol: Menguji nilai input nol
 - test_ketidaksamaan_segitiga: Menguji segitiga yang tidak valid
 - test_segitiga_sama_sisi: Menguji kasus sama sisi
 - test_segitiga_siku_siku: Menguji kasus siku-siku
 - test_segitiga_sama_kaki: Menguji kasus sama kaki
 - test_segitiga_bebas: Menguji kasus segitiga bebas
 -
- Fitur Utama Implementasi:
- Menggunakan nilai toleransi untuk perbandingan angka desimal
 - Menangani input bilangan bulat dan desimal
 - Mengimplementasikan semua klasifikasi segitiga yang diminta
 - Mencakup kasus uji komprehensif untuk setiap jalur
 - Menggunakan framework unittest Python untuk pengujian

Dan jika kita menjalankan **Demo Pengklasifikasi Segitiga**, outputnya akan seperti ini:

FileEditSelectionViewGoRunTerminalHelp

demoSegitiga.py C:\TIRTA\TUGAS TELKOM\SEMESTER 5\IMPLEMENTASI DAN PENGUJIAN PL IF-02-01 [MH

1from pengklasifikasiSegitiga import PengklasifikasiSegitiga

2

3def demo_klasifikasi():

4 klasifikasi = PengklasifikasiSegitiga()

5

6 print("=== DEMO PENGKLASIFIKASI SEGITIGA ===")

7 print("\n1. Contoh Input Bilangan Bulat:")

8 print("-----")

9

10 test_cases = [

11 (3, 3, 3), # Sama sisi

12 (3, 4, 5), # Siku-siku

13 (2, 2, 3), # Sama kaki

14 (3, 4, 6), # Bebas

15 (1, 2, 4), # Bukan segitiga

16 (-1, 2, 3), # Nilai negatif

17 (0, 2, 3), # Nilai nol

18]

19

20 for a, b, c in test_cases:

21 hasil = klasifikasi.klasifikasi_segitiga(a, b, c)

22 print(f"Sisi-sisi: {a}, {b}, {c} => {hasil}")

23

24 print("\n2. Contoh Input Bilangan Desimal:")

25 print("-----")

26 decimal_cases = [

27 (2.99, 3.01, 3.00), # Sama sisi (dengan toleransi)

28 (3.0, 4.0, 5.0), # Siku-siku

29 (2.5, 2.5, 3.0), # Sama kaki

30 (3.1, 4.2, 6.3), # Bebas

31]

32

33 for a, b, c in decimal_cases:

34 hasil = klasifikasi.klasifikasi_segitiga(a, b, c)

35 print(f"Sisi-sisi: {a}, {b}, {c} => {hasil}")

36

37 if __name__ == "__main__":

38 demo_klasifikasi()

39

PROBLEMSOUTPUTTERMINAL

PS C:\Users\Tirta Aditya> python -u "C:\TIRTA\TUGAS TELKOM\SEMESTER 5\IMPLEMENTASI DAN PENGUJIAN PL IF-02-01\demoSegitiga.py"

=== DEMO PENGKLASIFIKASI SEGITIGA ===

1. Contoh Input Bilangan Bulat:

Sisi-sisi: 3, 3, 3 => SEGITIGA SAMA SISI

Sisi-sisi: 3, 4, 5 => SEGITIGA SIKU-SIKU

Sisi-sisi: 2, 2, 3 => SEGITIGA SAMA KAKI

Sisi-sisi: 3, 4, 6 => SEGITIGA BEBAS

Sisi-sisi: 1, 2, 4 => BUKAN SEGITIGA

Sisi-sisi: -1, 2, 3 => BUKAN SEGITIGA

Sisi-sisi: 0, 2, 3 => BUKAN SEGITIGA

2. Contoh Input Bilangan Desimal:

Sisi-sisi: 2.99, 3.01, 3.0 => SEGITIGA SAMA SISI

Sisi-sisi: 3.0, 4.0, 5.0 => SEGITIGA SIKU-SIKU

Sisi-sisi: 2.5, 2.5, 3.0 => SEGITIGA SAMA KAKI

Sisi-sisi: 3.1, 4.2, 6.3 => SEGITIGA BEBAS

PS C:\Users\Tirta Aditya>

Dan jika kita menjalankan **unit test**, outputnya akan seperti ini:

FileEditSelectionViewGoRunTerminalHelp

pengklasifikasiSegitiga.py

C:\TIRTA\TUGAS TELKOM\SEMESTER 5\IMPLEMENTASI DAN PENGUJIAN PL IF-02-01 [MHJ]\

1class PengklasifikasiSegitiga:

2def __init__(self):

3self.TOLERANSI = 0.02 # Toleransi untuk perbandingan angka desimal

4

5def hampir_sama(self, a, b):

6return abs(a - b) <= self.TOLERANSI

7

8def klasifikasi_segitiga(self, a, b, c):

9# Langkah 1: Periksa nilai negatif atau nol

10if a <= 0 or b <= 0 or c <= 0:

11return "BUKAN SEGITIGA"

12

13# Urutkan sisi-sisi untuk memudahkan perbandingan

14sisi = sorted([a, b, c])

15terkecil, tengah, terbesar = sisi

16

17# Langkah 2: Periksa ketidaksamaan segitiga

18if terbesar >= terkecil + tengah:

19return "BUKAN SEGITIGA"

20

21# Langkah 3: Periksa segitiga sama sisi

22if self.hampir_sama(a, b) and self.hampir_sama(b, c):

23return "SEGITIGA SAMA SISI"

24

25# Langkah 4: Periksa segitiga siku-siku

26# Menggunakan teorema Pythagoras dengan toleransi

27if self.hampir_sama(terbesar * terbesar,

28terkecil * terkecil + tengah * tengah):

29return "SEGITIGA SIKU-SIKU"

30

31# Langkah 5: Periksa segitiga sama kaki

32if (self.hampir_sama(a, b) or

33self.hampir_sama(b, c) or

34self.hampir_sama(a, c)):

35return "SEGITIGA SAMA KAKI"

36

37# Langkah 6: Jika bukan semua di atas, maka segitiga bebas

38return "SEGITIGA BEBAS"

39

40import unittest

PROBLEMSOUTPUTTERMINAL

PS C:\Users\Tirta Aditya> python -u "c:\TIRTA\TUGAS TELKOM\SEMESTER 5\IMPLEMENTASI DAN
.....

Ran 7 tests in 0.000s

OK
PS C:\Users\Tirta Aditya>