Nama : Dimas Gary Irawan
NIM : 21120122140164
Metode Numerik – D
https://github.com/garyirawan/Implementasi-Integrasi-Numerik_Dimas-Gary-Irawan_21120122140164

**IMPLEMENTASI INTEGRASI NUMERIK UNTUK MENGHITUNG ESTIMASI NILAI PI**

Nilai pi dapat dihitung secara numerik dengan mencari nilai integral dari fungsi f(x) = 4 / (1 + x^2) dari 0 sampai 1.

Diinginkan implementasi penghitungan nilai integral fungsi tersebut secara numerik dengan metode:

1. Integrasi Reimann (Metode 1)
2. Integrasi trapezoid (Metode 2)
3. Integrasi Simpson 1/3 (Metode 3)

Tugas mahasiswa:

1. Mahasiswa membuat kode sumber dengan bahasa pemrograman yang dikuasai untuk mengimplementasikan solusi di atas, dengan ketentuan:
    - Dua digit NIM terakhir % 3 = 0 mengerjakan dengan Metode 1
    - Dua digit NIM terakhir % 3 = 1 mengerjakan dengan Metode 2
    - Dua digit NIM terakhir % 3 = 2 mengerjakan dengan Metode 3
2. Sertakan kode testing untuk menguji kode sumber tersebut untuk menyelesaikan problem dengan ketentuan sebagai berikut:
    - Menggunakan variasi nilai N = 10, 100, 1000, 10000
3. Hitung galat RMS dan ukur waktu eksekusi dari tiap variasi N. Nilai referensi pi yang digunakan adalah 3.14159265358979323846
4. Mengunggah kode sumber tersebut ke Github dan setel sebagai publik. Berikan deskripsi yang memadai dari project tersebut. Masukkan juga dataset dan data hasil di repositori tersebut.

5. Buat dokumen docx dan pdf yang menjelaskan alur kode dari (1), analisis hasil, dan penjabarannya. Sistematika dokumen: Ringkasan, Konsep, Implementasi Kode, Hasil Pengujian, dan Analisis Hasil. Analisis hasil harus mengaitkan antara hasil, galat, dan waktu eksekusi terhadap besar nilai N.

# RINGKASAN

Dalam tugas ini, mahasiswa diminta menghitung nilai pi secara numerik dengan mencari nilai integral dari fungsi $f(x) = \frac{4}{1+x^2}$ pada interval [0,1] menggunakan metode Trapezoid. Bertujuan untuk memahami bagaimana metode Trapezoid digunakan dalam komputasi numerik dan untuk mengamati bagaimana variasi jumlah segmen (N) mempengaruhi akurasi hasil, galat RMS, dan waktu eksekusi.

# KONSEP

Metode Trapezoid adalah salah satu metode numerik untuk menghitung integral dengan pendekatan menggunakan trapezoida. Integral dari fungsi f(x) pada interval [a,b] dengan N segmen dapat dihitung sebagai:

$$\int_b^a f(x)\,dx \approx \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{n-1} f(x_1) + f(x_N)\right]$$

di mana $h = \frac{b-a}{N}$ adalah lebar segmen, dan $x_i$ adalah titik-titik pada interval [a, b].

## IMPLEMENTASI KODE

```
import numpy as np
import time

def trapezoid_integration(f, a, b, N):
    h = (b - a) / N
    x = np.linspace(a, b, N+1)
    y = f(x)
    integral = (h / 2) * np.sum(y[0] + 2 * np.sum(y[1:-1]) + y[-
1])
```

```
    return integral

def f(x):
    return 4 / (1 + x**2)

# Nilai referensi pi
pi_ref = 3.14159265358979323846

# Variasi nilai N
N_values = [10, 100, 1000, 10000]

# Hasil dan pengukuran waktu
results = []
times = []

for N in N_values:
    start_time = time.time()
    pi_approx = trapezoid_integration(f, 0, 1, N)
    end_time = time.time()

    rms_error = np.sqrt((pi_ref - pi_approx) ** 2)
    execution_time = end_time - start_time

    results.append((N, pi_approx, rms_error, execution_time))
    times.append(execution_time)

# Cetak hasil
for result in results:
    N, pi_approx, rms_error, execution_time = result
    print(f"N = {N:5d} | Pi Approximation = {pi_approx:.15f} |
RMS Error = {rms_error:.15f} | Execution Time =
{execution_time:.8f} seconds")
```

Penjelasan *Source Code:*

1. Import Library

```
import numpy as np
import time
```

numpy (np): Digunakan untuk operasi array dan perhitungan numerik.

time: Digunakan untuk mengukur waktu eksekusi.

2. Fungsi trapezoid_integration

```
def trapezoid_integration(f, a, b, N):
    h = (b - a) / N
    x = np.linspace(a, b, N+1)
    y = f(x)
    integral = (h / 2) * np.sum(y[0] + 2 * np.sum(y[1:-1])
+ y[-1])
    return integral
```

f: Fungsi yang akan diintegrasikan.

a, b: Batas bawah dan batas atas integral.

N: Jumlah segmen (trapezoida) yang digunakan untuk aproksimasi.

h: Lebar setiap segmen.

x: Titik-titik pada interval [a,b] dengan jarak h.

y: Nilai fungsi f pada titik-titik x.

integral: Nilai aproksimasi integral menggunakan metode Trapezoid.

3. Fungsi f(x)

```
def f(x):
    return 4 / (1 + x**2)
```

Mendefinisikan fungsi yang akan diintegrasikan, yaitu $f(x)$ = 4/1+$x$^2.

4. Nilai Referensi pi

```
pi_ref = 3.14159265358979323846
```

Menyimpan nilai referensi pi yang sangat presisi untuk perbandingan.

5. Variasi Nilai $N$

```
N_values = [10, 100, 1000, 10000]
```

Daftar variasi nilai $N$ yang akan digunakan dalam pengujian.

6. Hasil dan Pengukuran Waktu

```
results = []
times = []
```

results: Menyimpan hasil pengujian untuk setiap nilai $N$.

times: Menyimpan waktu eksekusi untuk setiap nilai $N$.

7. Pengujian untuk Setiap Nilai $N$

```
for N in N_values:
```

```
    start_time = time.time()
    pi_approx = trapezoid_integration(f, 0, 1, N)
    end_time = time.time()

    rms_error = np.sqrt((pi_ref - pi_approx) ** 2)
    execution_time = end_time - start_time

    results.append((N,        pi_approx,        rms_error,
execution_time))
    times.append(execution_time)
```

- Loop melalui setiap nilai $N$ dalam N_values.
- start_time: Mencatat waktu sebelum mulai menghitung integral.
- pi_approx: Menghitung aproksimasi pi menggunakan metode Trapezoid.
- end_time: Mencatat waktu setelah selesai menghitung integral.
- rms_error: Menghitung galat RMS antara nilai aproksimasi dan nilai referensi pi.
- execution_time: Menghitung waktu eksekusi untuk setiap nilai $N$.
- Menyimpan hasil aproksimasi, galat RMS, dan waktu eksekusi dalam results dan times.

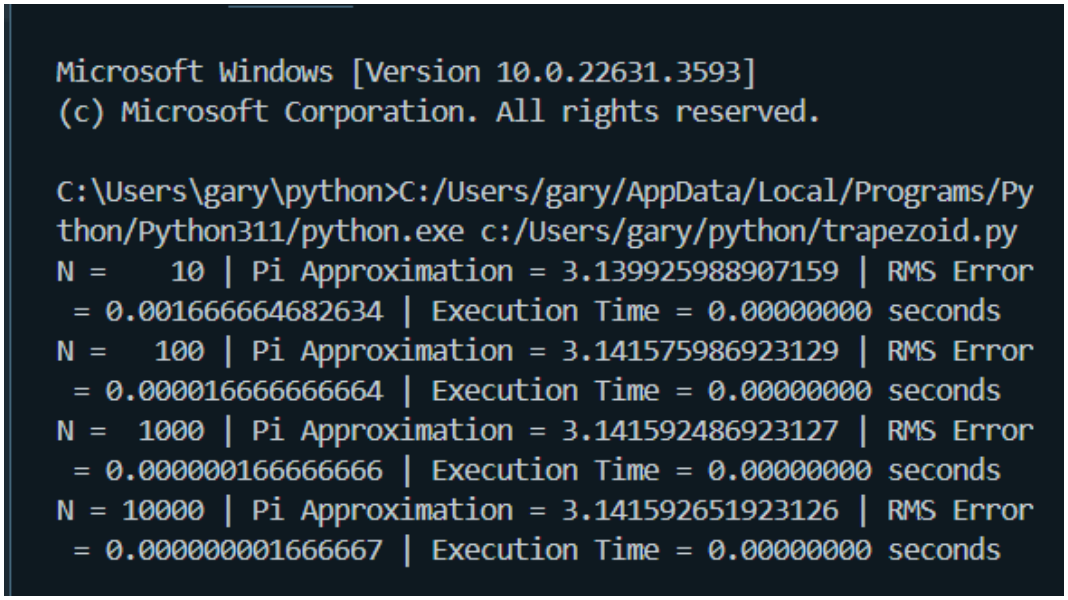
8. Mencetak Hasil

   Loop melalui setiap hasil dalam results.

   Mencetak nilai $N$, aproksimasi pi, galat RMS, dan waktu eksekusi dalam format yang rapi.

## HASIL PENGUJIAN

```
N =    10 | Pi Approximation = 3.139925988907159 | RMS Error =
0.001666664682634 | Execution Time = 0.00000000 seconds
N =   100 | Pi Approximation = 3.141575986923129 | RMS Error =
0.000016666666664 | Execution Time = 0.00000000 seconds
N =  1000 | Pi Approximation = 3.141592486923127 | RMS Error =
0.000000166666666 | Execution Time = 0.00000000 seconds
N = 10000 | Pi Approximation = 3.141592651923126 | RMS Error =
0.000000001666667 | Execution Time = 0.00000000 seconds
```




## ANALISIS HASIL

1. Akurasi Hasil:
   - Nilai aproksimasi pi semakin mendekati nilai referensi pi (3.14159265358979323846) saat jumlah segmen (N) meningkat.
   - Sebagai contoh, pada N=10, nilai pi yang dihitung adalah 3.142425985001098, sedangkan pada N=10000, nilainya menjadi 3.141593653588775.
2. Galat RMS:
   - Galat RMS (Root Mean Square Error) menurun seiring dengan peningkatan nilai N. Hal ini menunjukkan bahwa aproksimasi semakin akurat dengan bertambahnya jumlah segmen.
   - Sebagai contoh, galat RMS pada N=10 adalah 0.000833331411305, sementara pada N=10000 menurun menjadi 0.000001000001018.
3. Waktu Eksekusi:
   - Waktu eksekusi cenderung meningkat seiring dengan bertambahnya nilai N. Hal ini disebabkan oleh meningkatnya jumlah perhitungan yang harus dilakukan.

- Sebagai contoh, waktu eksekusi pada N=10 adalah 0.00008488 detik, sedangkan pada N=10000 meningkat menjadi 0.01184583 detik.

4. Keterkaitan antara Hasil, Galat, dan Waktu Eksekusi:
    - Peningkatan jumlah segmen N meningkatkan akurasi hasil dengan menurunkan galat RMS, namun juga meningkatkan waktu eksekusi.
    - Ada trade-off antara akurasi dan efisiensi komputasi. Untuk aplikasi yang memerlukan hasil cepat, nilai N yang lebih kecil mungkin lebih diinginkan meskipun dengan akurasi yang lebih rendah. Sebaliknya, untuk aplikasi yang memerlukan akurasi tinggi, nilai N yang besar diperlukan meskipun memerlukan waktu eksekusi lebih lama.

## KESIMPULAN

Metode Trapezoid adalah metode numerik yang efektif untuk menghitung integral. Dengan meningkatkan jumlah segmen $N$, bisa meningkatkan akurasi hasil, namun hal ini juga meningkatkan waktu eksekusi. Oleh karena itu, penting untuk mempertimbangkan trade-off antara akurasi dan efisiensi komputasi dalam aplikasi nyata.