Nama : Gina Nabila

NIM : 21120122130055

Mata Kuliah : Metode Numerik – Kelas D

GitHub :<https://github.com/gin-na/Implementasi-Interpolasi_Metode-Numerik_Gina-Nabila>

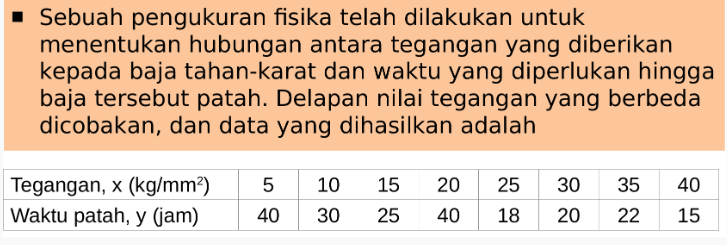
**Pertemuan 10 – Implementasi Interpolasi**

Diinginkan aplikasi untuk mencari solusi dari problem pengujian yang memperoleh data terbatas (data terlampir) dengan interpolasi masing-masing menggunakan metode:

1. polinom Lagrange
2. polinom Newton

Tugas mahasiswa:

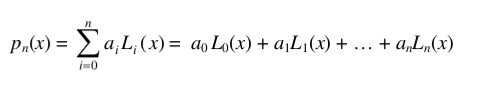
1. Mahasiswa membuat kode sumber dengan bahasa pemrograman yang dikuasai untuk mengimplementasikan solusi di atas
2. Sertakan kode testing untuk menguji kode sumber tersebut untuk menyelesaikan problem dalam gambar. Plot grafik hasil interpolasi dengan 5 <= x <= 40
3. Mengunggah kode sumber tersebut ke Github dan setel sebagai publik. Berikan deskripsi yang memadai dari project tersebut
4. Buat dokumen docx dan pdf yang menjelaskan alur kode dari (1), analisis hasil, dan penjabarannya

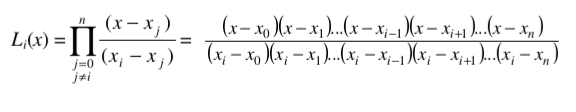


Jawaban:

1. **Polinom Lagrange**

Metode polinom interpolasi yang banyak digunakan dalam komputasi numerik adalah polinom Lagrange, Newton, dan Newton-Gregory. Bentuk umum polinom Lagrange derajat untuk titik berbeda adalah:





**Source** **Code**:

import numpy as np

import matplotlib.pyplot as plt

# Fungsi untuk melakukan interpolasi lagrange

def lagrange\_interpolation(x\_points, y\_points, x):

n = len(x\_points)

result = 0.0

for i in range(n):

term = y\_points[i]

for j in range(n):

if i != j:

term \*= (x - x\_points[j]) / (x\_points[i] - x\_points[j])

result += term

return result

# Fungsi untuk memasukkan jumlah titik dan nilai

def get\_user\_input():

n = int(input("Masukkan jumlah titik data: "))

x\_points = []

y\_points = []

for i in range(n):

x = float(input(f"Masukkan nilai x({i}): "))

y = float(input(f"Masukkan nilai y({i}): "))

x\_points.append(x)

y\_points.append(y)

return x\_points, y\_points

# Fungsi utama

def main():

# Mendapatkan nilai yang dimasukkan

x\_points, y\_points = get\_user\_input()

# Menampilkan titik data yang dimasukkan

print("Titik data yang dimasukkan:")

for x, y in zip(x\_points, y\_points):

print(f"({x}, {y})")

# Menghitung nilai interpolasi pada titik tertentu

x\_interpolate = float(input("Masukkan nilai x yang ingin diinterpolasi: "))

y\_interpolate = lagrange\_interpolation(x\_points, y\_points, x\_interpolate)

print(f"Nilai interpolasi pada x = {x\_interpolate} adalah y = {y\_inter40polate}")

# Membuat grafik interpolasi

x\_plot = np.linspace(min(x\_points) - 1, max(x\_points) + 1, 1000)

y\_plot = [lagrange\_interpolation(x\_points, y\_points, x) for x in x\_plot]

plt.plot(x\_plot, y\_plot, color='royalblue', label='Interpolasi Lagrange')

plt.scatter(x\_points, y\_points, color='magenta', label='Titik Data')

plt.scatter([x\_interpolate,[y\_interpolate], color='mediumpurple',zorder=5,label=f'Interpolasi (x={x\_interpolate})')

plt.xlabel('x')

plt.ylabel('y')

plt.title('Interpolasi Lagrange')

plt.legend()

plt.grid(True)

plt.show()

# Menjalankan program utama

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Penjelasan alur kode:**

1. **Input nilai titik** **data**

# Fungsi untuk memasukkan jumlah titik dan nilai

def get\_user\_input():

n = int(input("Masukkan jumlah titik data: "))

x\_points = []

y\_points = []

for i in range(n):

x = float(input(f"Masukkan nilai x({i}): "))

y = float(input(f"Masukkan nilai y({i}): "))

x\_points.append(x)

y\_points.append(y)

return x\_points, y\_points

Dengan kode di atas, pengguna dapat memasukkan jumlah titik data dan nilai tiap titik.

1. **Inisialisasi fungsi interpolasi Lagrange**

def lagrange\_interpolation(x\_points, y\_points, x):

n = len(x\_points)

result = 0.0

for i in range(n):

term = y\_points[i]

for j in range(n):

if i != j:

term \*= (x - x\_points[j]) / (x\_points[i] - x\_points[j])

result += term

return result

Dengan kode di atas, Fungsi lagrange\_interpolation(x\_points, y\_points, x) menghitung nilai interpolasi Lagrange pada titik 𝑥 tertentu berdasarkan titik data yang dimasukkan. Fungsi ini mengikuti algoritma interpolasi Lagrange, yaitu dengan mengalikan setiap nilai 𝑦 dengan basis polinomial Lagrange dan menambahkannya ke nilai hasil akhir.

1. **Menghasilkan grafik interpolasi**

# Membuat grafik interpolasi

x\_plot = np.linspace(min(x\_points) - 1, max(x\_points) + 1, 1000)

y\_plot = [lagrange\_interpolation(x\_points, y\_points, x) for x in x\_plot]

plt.plot(x\_plot, y\_plot, color='royalblue', label='Interpolasi Lagrange')

plt.scatter(x\_points, y\_points, color='magenta', label='Titik Data')

plt.scatter([x\_interpolate], [y\_interpolate], color='mediumpurple', zorder=5, label=f'Interpolasi (x={x\_interpolate})')

plt.xlabel('x')

plt.ylabel('y')

plt.title('Interpolasi Lagrange')

plt.legend()

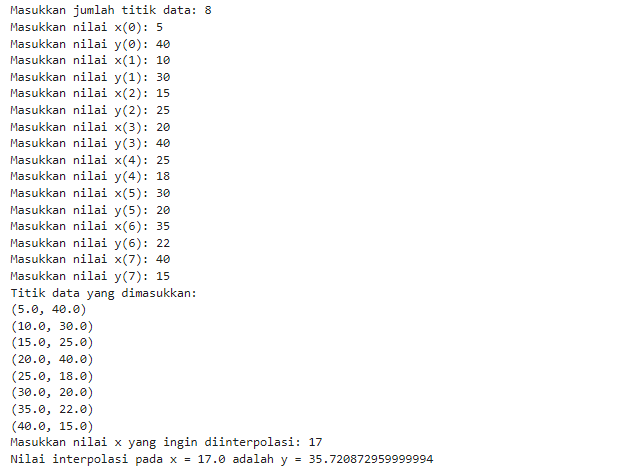
plt.grid(True)

plt.show()

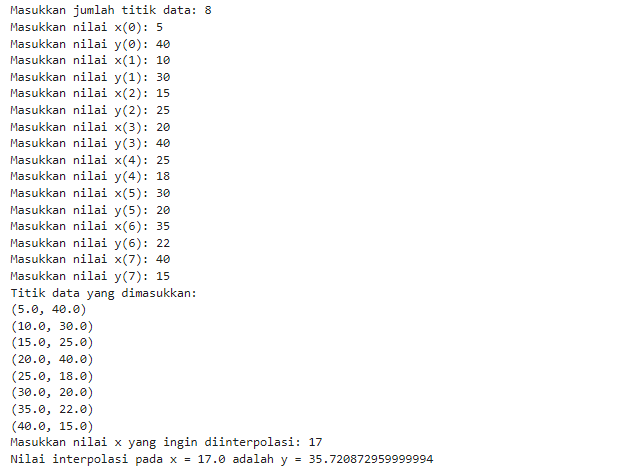
Setelah nilai interpolasi dihitung, sebuah grafik dibuat untuk menampilkan hasil interpolasi Lagrange. Rentang nilai 𝑥 untuk plotting dihasilkan menggunakan np.linspace untuk mencakup rentang yang lebih luas dari titik data. Fungsi interpolasi Lagrange dihitung untuk setiap nilai 𝑥 dalam rentang tersebut. Garis interpolasi dan titik data ditampilkan dalam grafik menggunakan plt.plot dan plt.scatter.

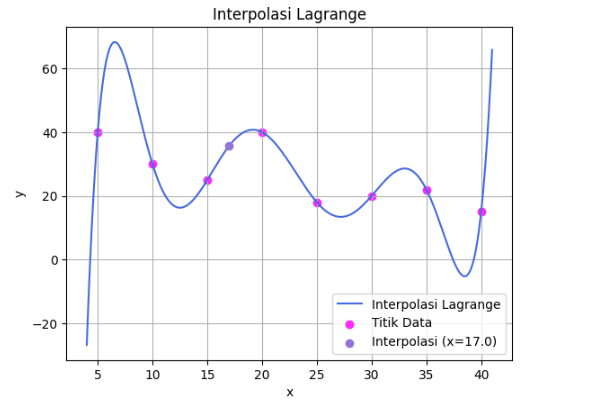
**Code testing:**

Dalam code testing, jumlah titik data dan nilainya disesuaikan dengan soal. Sehingga masukannya akan seperti berikut:



Dari nilai yang dimasukkan, akan dihasilkan jawaban nilai interpolasi dan grafik sebagai berikut:

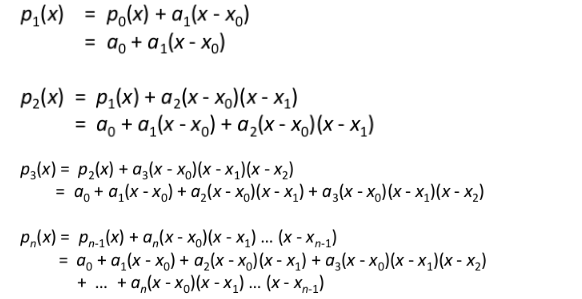




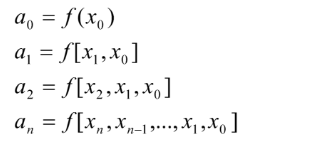
1. **Polinom Newton**

Dalam interpolasi Lagrange terdapat kelemahan, yaitu ketika terjadi penambahan atau pengurangan titik data yang diketahui maka bentuk polinom sebelumnya tidak dapat digunakan lagi. Hal ini tentu akan merepotkan karena setiap adanya perubahan kuantitas titik data harus dicari bentuk polinom dari awal lagi. Oleh karena itu, diperlukan solusi untuk mengatasi ini, yaitui dengan menggunakan interpolasi newton.

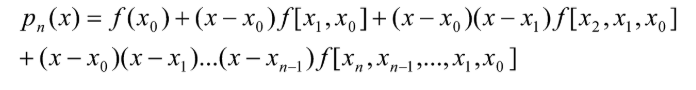
Tahapan pembentukan polinom newton adalah sebagai berikut:



Di mana, nilai konstanta a merupakan nilai selisih terbagi. Dengan nilai:



Dengan demikian polinom newton dapat ditulis dalam bentuk lengkap sebagai berikut:



**Source** **Code**:

import numpy as np

import matplotlib.pyplot as plt

# Fungsi untuk menghitung konstanta a pada interpolasi Newton

def calculate\_constants(x\_points, y\_points):

n = len(x\_points)

constants = np.copy(y\_points)

for j in range(1, n):

for i in range(n-1, j-1, -1):

constants[i] = (constants[i] - constants[i-1]) / (x\_points[i] - x\_points[i-j])

return constants

# Fungsi untuk menghitung nilai interpolasi Newton pada titik x

def newton\_interpolation(x\_points, constants, x):

n = len(x\_points)

result = constants[-1]

for i in range(n-2, -1, -1):

result = result \* (x - x\_points[i]) + constants[i]

return result

# Fungsi untuk memasukkan jumlah titik dan nilai

def get\_user\_input():

n = int(input("Masukkan jumlah titik data: "))

x\_points = []

y\_points = []

for i in range(n):

x = float(input(f"Masukkan nilai x({i}): "))

y = float(input(f"Masukkan nilai y({i}): "))

x\_points.append(x)

y\_points.append(y)

return x\_points, y\_points

# Fungsi utama

def main():

# Mendapatkan nilai yang dimasukkan

x\_points, y\_points = get\_user\_input()

# Menghitung koefisien interpolasi Newton

constants = calculate\_constants(x\_points, y\_points)

# Menampilkan titik data yang dimasukkan

print("Titik data yang dimasukkan:")

for x, y in zip(x\_points, y\_points):

print(f"({x}, {y})")

# Menghitung nilai interpolasi pada titik tertentu

x\_interpolate = float(input("Masukkan nilai x yang ingin diinterpolasi: "))

y\_interpolate = newton\_interpolation(x\_points, constants, x\_interpolate)

print(f"Nilai interpolasi pada x = {x\_interpolate} adalah y = {y\_interpolate}")

# Membuat grafik interpolasi

x\_plot = np.linspace(min(x\_points) - 1, max(x\_points) + 1, 1000)

y\_plot = [newton\_interpolation(x\_points, constants, x) for x in x\_plot]

plt.plot(x\_plot, y\_plot, color='hotpink', label='Interpolasi Newton')

plt.scatter(x\_points, y\_points, color='orange', label='Titik Data')

plt.scatter([x\_interpolate], [y\_interpolate], color='blueviolet', zorder=5, label=f'Interpolasi (x={x\_interpolate})')

plt.xlabel('x (kg/mm^2)')

plt.ylabel('y (jam)')

plt.title('Interpolasi Newton')

plt.legend()

plt.grid(True)

plt.show()

# Menjalankan program utama

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Penjelasan alur kode:**

1. **Input nilai titik data**

# Fungsi untuk memasukkan jumlah titik dan nilai

def get\_user\_input():

n = int(input("Masukkan jumlah titik data: "))

x\_points = []

y\_points = []

for i in range(n):

x = float(input(f"Masukkan nilai x({i}): "))

y = float(input(f"Masukkan nilai y({i}): "))

x\_points.append(x)

y\_points.append(y)

return x\_points, y\_points

Dengan kode di atas, pengguna dapat memasukkan jumlah titik data dan nilai tiap titik

1. **Melakukan perhitungan nilai konstanta (selisih terbagi)**

# Fungsi untuk menghitung konstanta a pada interpolasi Newton

def calculate\_constants(x\_points, y\_points):

n = len(x\_points)

constants = np.copy(y\_points)

for j in range(1, n):

for i in range(n-1, j-1, -1):

constants[i] = (constants[i] - constants[i-1]) / (x\_points[i] - x\_points[i-j])

return constants

Fungsi di atas digunakan dalam menghitung nilai konstanta pada interpolasi newton atau disebut juga dengan selisih terbagi (*divided-difference*) di mana nilainya nanti akan digunakan dalam menghitung nilai interpolasi.

1. **Menghitung nilai interpolasi**

# Fungsi untuk menghitung nilai interpolasi Newton pada titik x

def newton\_interpolation(x\_points, constants, x):

n = len(x\_points)

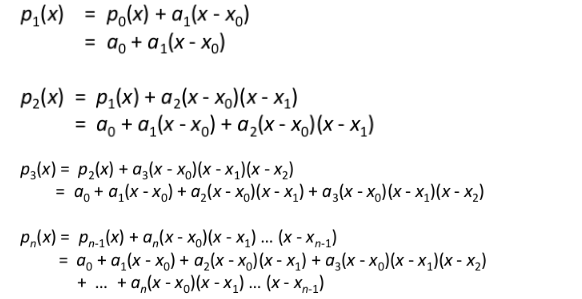
result = constants[-1]

for i in range(n-2, -1, -1):

result = result \* (x - x\_points[i]) + constants[i]

return result

Fungsi newton\_interpolation menghitung nilai interpolasi Newton pada titik 𝑥 yang diberikan. Ini dilakukan dengan menggunakan konstanta interpolasi Newton yang telah dihitung sebelumnya. Nilai interpolasi dihitung menggunakan rumus:



1. **Menghasilkan grafik interpolasi**

# Membuat grafik interpolasi

x\_plot = np.linspace(min(x\_points) - 1, max(x\_points) + 1, 1000)

y\_plot = [newton\_interpolation(x\_points, constants, x) for x in x\_plot]

plt.plot(x\_plot, y\_plot, color='hotpink', label='Interpolasi Newton')

plt.scatter(x\_points, y\_points, color='orange', label='Titik Data')

plt.scatter([x\_interpolate], [y\_interpolate], color='blueviolet', zorder=5, label=f'Interpolasi (x={x\_interpolate})')

plt.xlabel('x (kg/mm^2)')

plt.ylabel('y (jam)')

plt.title('Interpolasi Newton')

plt.legend()

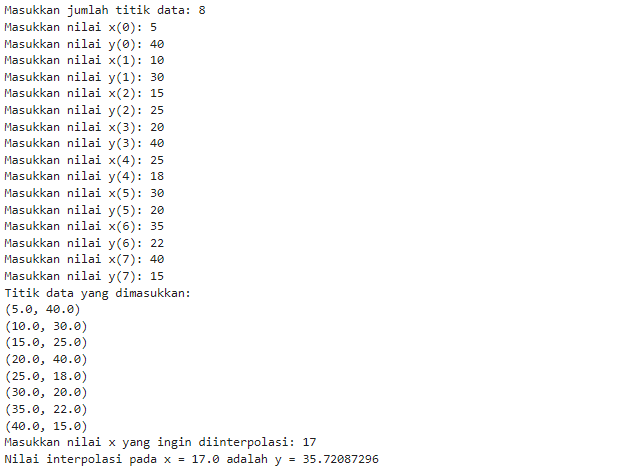
plt.grid(True)

plt.show()

Setelah nilai-nilai interpolasi dihitung, grafik interpolasi Newton dan titik data ditampilkan menggunakan Matplotlib. Kurva interpolasi dihasilkan menggunakan nilai-nilai x\_plot yang dihasilkan secara linier dengan menggunakan NumPy linspace. Titik-titik data dan titik interpolasi juga ditampilkan pada grafik.

**Code Testing:**

Dalam code testing, jumlah titik data dan nilainya disesuaikan dengan soal. Sehingga masukannya akan seperti berikut:



Dari nilai yang dimasukkan, akan dihasilkan jawaban nilai interpolasi dan grafik sebagai berikut:

