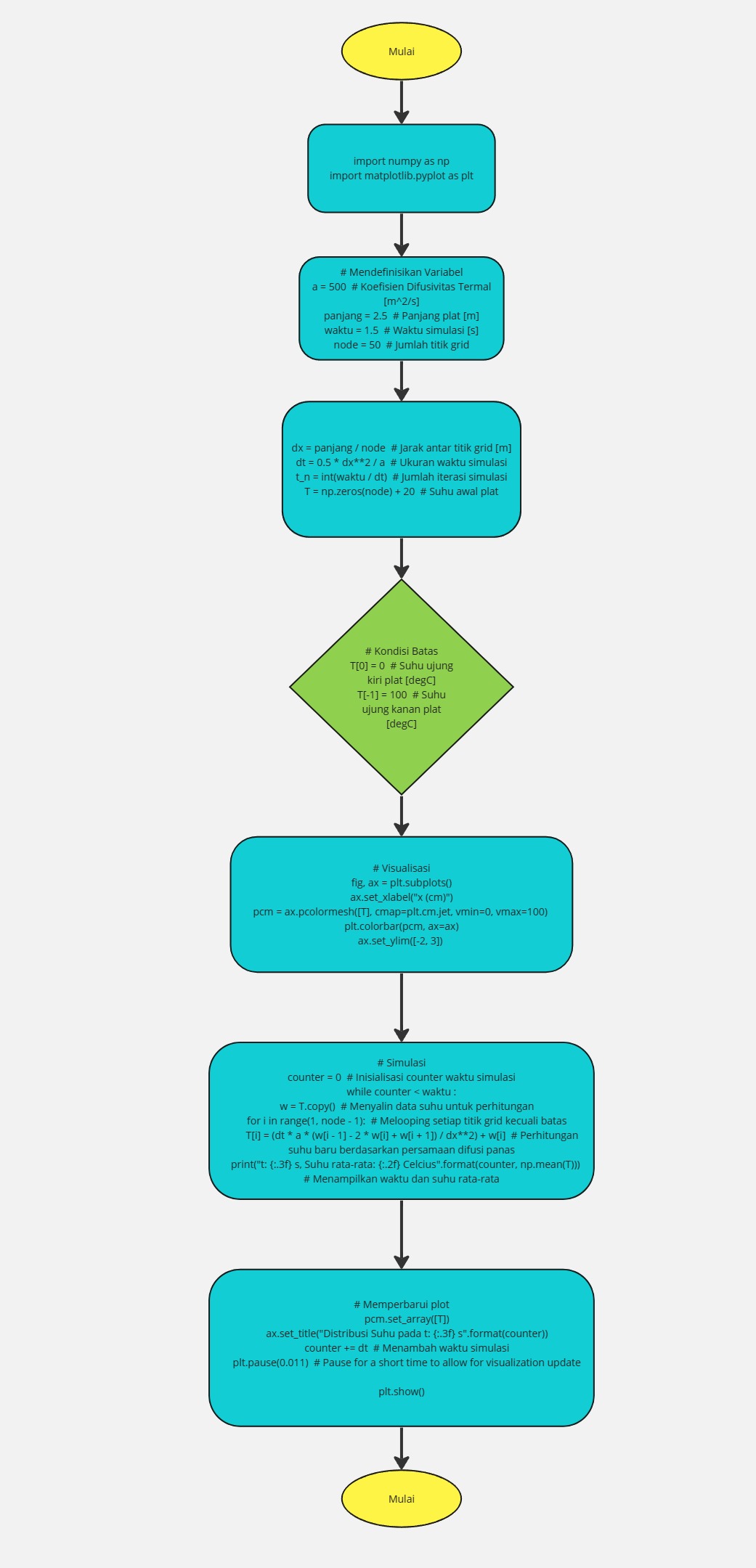
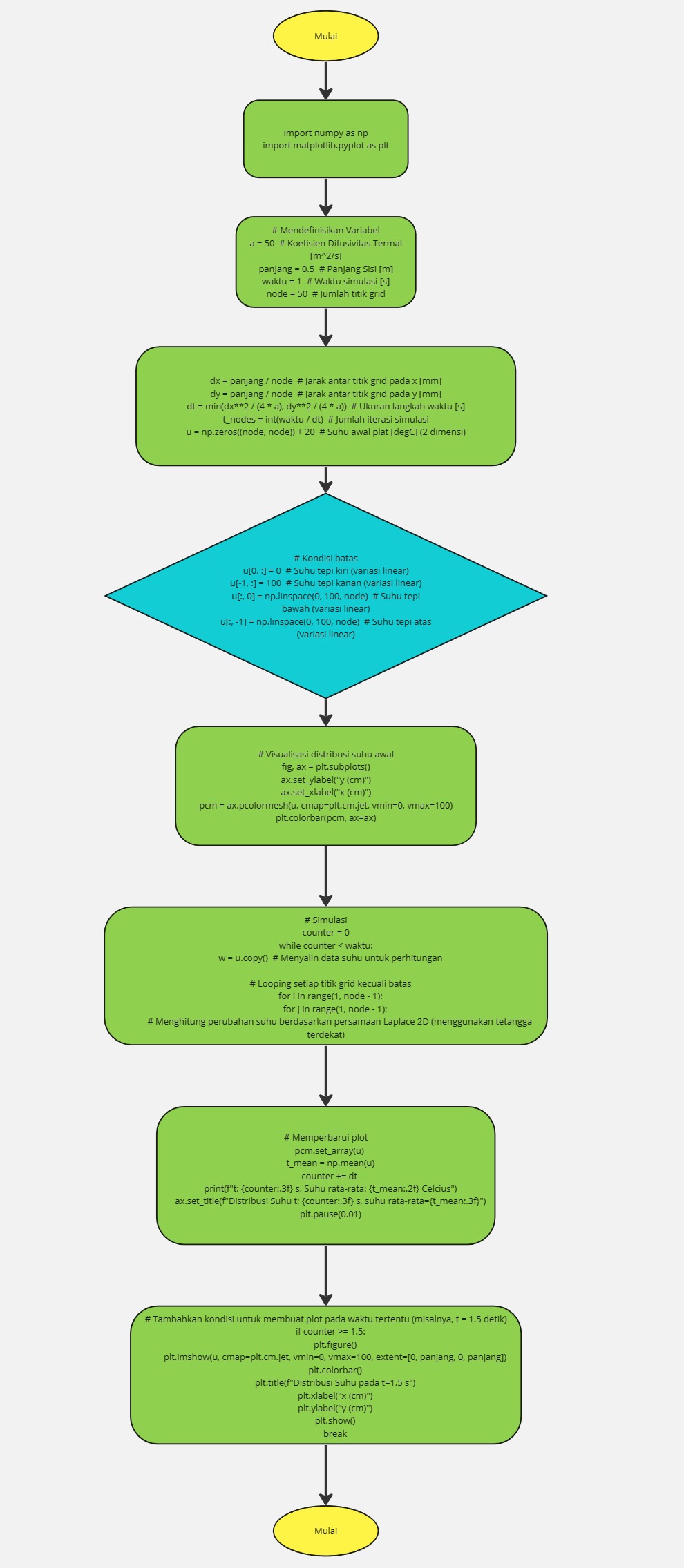
Paundra Maxi Hafiansyah

1217030029

1.





Library

1. Numpy sebagai operasi array numerik.
2. Matplotlib sebagai visualisasi data dalam bentuk grafik dan plot data.

Variabel dan Parameter

1. a sebagai koefisien difusi termal.
2. panjang sebagai panjang pelat.
3. waktu sebagai waktu saat simulasi.
4. node sebagai jumlah titik grid.

Inisiasi grid dan waktu

1. dx sebagai jarak antara titik grid.
2. dt sebagai ukuran waktu simulasi.
3. T sebagai suhu awal plat.

Kondisi batas

1. T[0] suhu ujung kiri plat.
2. T[-1] suhu ujung kanan plat.

Visualisasi

Menampilkan data dalam bentuk plot.

Simulasi

Membuat simulasi menggunakan iterasi waktu, pada setiap iterasi menghitung setiap titik grid berdasarkan persamaan difusi panas.

Menampilkan plot data

Menampilkan plot data sebagai output program.

2.

Konduksi panas dalam satu dimensi dan dua dimensi merupakan dua kasus yang berbeda dalam pemodelan fenomena transfer panas. Perbedaan utamanya terletak pada dimensi ruang dan kompleksitas masalah yang harus diatasi. Metode Finite Difference digunakan untuk menyelesaikan persamaan diferensial parsial yang menggambarkan konduksi panas dalam kasus-kasus ini.

Konduksi Panas 1 Dimensi:

a. Dimensi Ruang:

* Sumbu Satu Dimensi: Konduksi panas dalam satu dimensi biasanya diwakili oleh suatu batang, plat, atau media yang panjangnya lebih signifikan daripada dimensinya yang lain.

b. Persamaan Diferensial:

* Persamaan Panas 1D: Persamaan diferensial parsial untuk konduksi panas satu dimensi biasanya berbentuk seperti ini:

c. Grid Finite Difference:

* Grid Satu Dimensi: Pada metode finite difference, grid hanya diperlukan pada satu dimensi (biasanya sumbu x). Maka, perubahan suhu hanya bergantung pada posisi pada sumbu x.

Konduksi Panas 2 Dimensi:

a. Dimensi Ruang:

* + Bidang Dua Dimensi: Konduksi panas dalam dua dimensi terjadi pada bidang atau permukaan yang memiliki dimensi panjang dan lebar.

b. Persamaan Diferensial:

* + Persamaan Panas 2D:Persamaan diferensial parsial untuk konduksi panas dua dimensi umumnya berbentuk seperti ini:

c. Grid Finite Difference:

* + Grid Dua Dimensi: Pada metode finite difference untuk konduksi panas dua dimensi, grid diperlukan pada kedua sumbu x dan y. Perubahan suhu dihitung berdasarkan turunan kedua suhu terhadap kedua sumbu.

Pada 1 dimensi saat waktu 0 sekon

import numpy as np

import matplotlib.pyplot as plt

# Mendefinisikan Variabel

a = 500  # Koefisien Difusivitas Termal [m^2/s]

panjang = 2.5  # Panjang plat [m]

waktu = 1.5  # Waktu simulasi [s]

node = 50  # Jumlah titik grid

dx = panjang / node  # Jarak antar titik grid [m]

dt = 0.5 \* dx\*\*2 / a  # Ukuran waktu simulasi [s]

t\_n = int(waktu / dt)  # Jumlah iterasi simulasi

T = np.zeros(node) + 20  # Suhu awal plat [degC]

# Kondisi Batas

T[0] = 0  # Suhu ujung kiri plat [degC]

T[-1] = 100  # Suhu ujung kanan plat [degC]

# Visualisasi

fig, ax = plt.subplots()

ax.set\_xlabel("x (cm)")

pcm = ax.pcolormesh([T], cmap=plt.cm.jet, vmin=0, vmax=100)  # Plot distribusi suhu

plt.colorbar(pcm, ax=ax)

ax.set\_ylim([-2, 3])  # Batas skala y

counter = 0  # Inisialisasi counter waktu simulasi

w = T.copy()  # Menyalin data suhu untuk perhitungan

for i in range(1, node - 1):  # Melooping setiap titik grid kecuali batas

        T[i] = (dt \* a \* (w[i - 1] - 2 \* w[i] + w[i + 1]) / dx\*\*2) + w[i]  # Perhitungan suhu baru berdasarkan persamaan difusi panas

counter += dt  # Menambah waktu simulasi

print("t: {:.3f} s, Suhu rata-rata: {:.2f} Celcius".format(counter, np.mean(T)))  # Menampilkan waktu dan suhu rata-rata

# Memperbarui plot

pcm.set\_array([T])

ax.set\_title("Distribusi Suhu pada t: {:.3f} s".format(counter))

plt.show()

pada 1 dimensi saat waktu 1,5 sekon

import numpy as np

import matplotlib.pyplot as plt

# Mendefinisikan Variabel

a = 500  # Koefisien Difusivitas Termal [m^2/s]

panjang = 2.5  # Panjang plat [m]

waktu = 1.5  # Waktu simulasi [s]

node = 50  # Jumlah titik grid

dx = panjang / node  # Jarak antar titik grid [m]

dt = 0.5 \* dx\*\*2 / a  # Ukuran waktu simulasi [s]

t\_n = int(waktu / dt)  # Jumlah iterasi simulasi

T = np.zeros(node) + 20  # Suhu awal plat [degC]

# Kondisi Batas

T[0] = 0  # Suhu ujung kiri plat [degC]

T[-1] = 100  # Suhu ujung kanan plat [degC]

# Visualisasi

fig, ax = plt.subplots()

ax.set\_xlabel("x (cm)")

pcm = ax.pcolormesh([T], cmap=plt.cm.jet, vmin=0, vmax=100)  # Plot distribusi suhu

plt.colorbar(pcm, ax=ax)

ax.set\_ylim([-2, 3])  # Batas skala y

# Simulasi

counter = 0  # Inisialisasi counter waktu simulasi

while counter < waktu :

    w = T.copy()  # Menyalin data suhu untuk perhitungan

    for i in range(1, node - 1):  # Melooping setiap titik grid kecuali batas

        T[i] = (dt \* a \* (w[i - 1] - 2 \* w[i] + w[i + 1]) / dx\*\*2) + w[i]  # Perhitungan suhu baru berdasarkan persamaan difusi panas

    print("t: {:.3f} s, Suhu rata-rata: {:.2f} Celcius".format(counter, np.mean(T)))  # Menampilkan waktu dan suhu rata-rata

    # Memperbarui plot

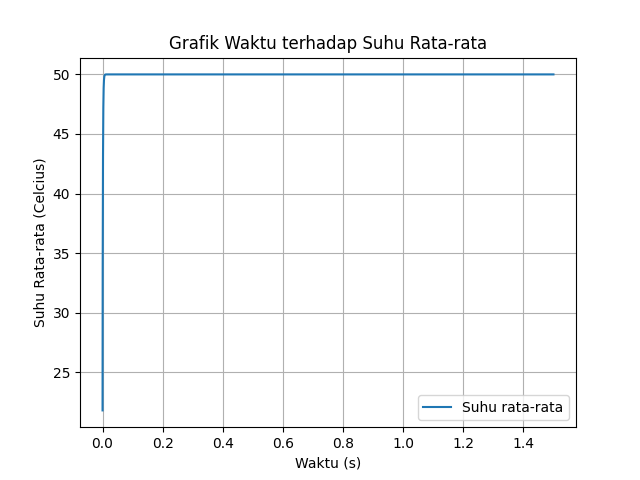
    pcm.set\_array([T])

    ax.set\_title("Distribusi Suhu pada t: {:.3f} s".format(counter))

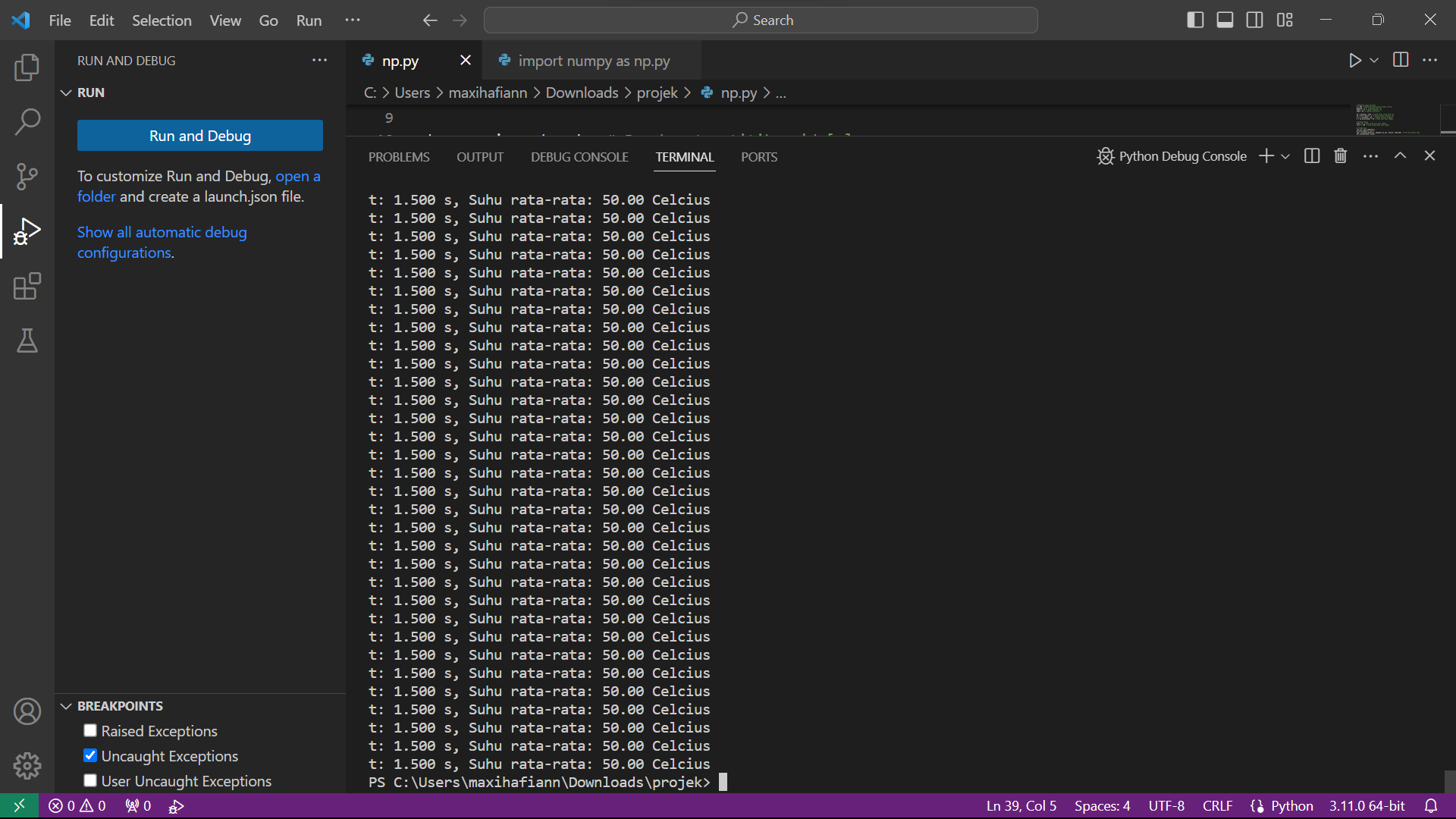
    counter += dt  # Menambah waktu simulasi

    plt.pause(0.01)  # Pause for a short time to allow for visualization update

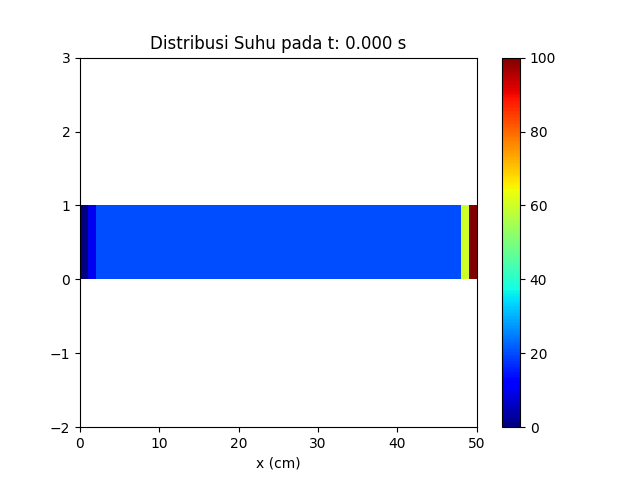
plt.show()



Grafik antara suhu terhadap waktu



Data suhu dan waktu



plot ketika waktu 0 sekon

pada 2 d saat waktu 0 sekon

import numpy as np

import matplotlib.pyplot as plt

# Mendefinisikan Variabel

a = 50  # Koefisien Difusivitas Termal [m^2/s]

panjang = 0.5  # Panjang plat [m]

waktu = 1.5  # Waktu simulasi [s]

node = 50  # Jumlah titik grid

dx = panjang / node  # Jarak antar titik grid pada x [mm]

dy = panjang / node  # Jarak antar titik grid pada y [mm]

dt = min(dx \*\* 2 / (4 \* a), dy \*\* 2 / (4 \* a))  # Ukuran langkah waktu [s] (pilih yang lebih kecil agar stabil)

t\_nodes = int(waktu / dt)  # Jumlah iterasi simulasi

u = np.zeros((node, node)) + 20  # Suhu awal plat [degC] (2 dimensi)

# Kondisi batas

u[0, :] = 0   # Suhu tepi kiri (variasi linear)

u[-1, :] = 100   # Suhu tepi kanan (variasi linear)

u[:, 0] = np.linspace(0, 100, node)  # Suhu tepi bawah (variasi linear)

u[:, -1] = np.linspace(0, 100, node)  # Suhu tepi atas (variasi linear)

# Visualisasi distribusi suhu awal

fig, ax = plt.subplots()

ax.set\_ylabel("y (cm)")

ax.set\_xlabel("x (cm)")

pcm = ax.pcolormesh(u, cmap=plt.cm.jet, vmin=0, vmax=100)

plt.colorbar(pcm, ax=ax)

counter = 0

w = u.copy()  # Menyalin data suhu untuk perhitungan

# Looping setiap titik grid kecuali batas

for i in range(1, node - 1):

    for j in range(1, node - 1):

        # Menghitung perubahan suhu berdasarkan persamaan Laplace 2D (menggunakan tetangga terdekat)

        dd\_ux = (w[i - 1, j] - 2 \* w[i, j] + w[i + 1, j]) / dx \*\* 2

        dd\_uy = (w[i, j - 1] - 2 \* w[i, j] + w[i, j + 1]) / dy \*\* 2

        u[i, j] = dt \* a \* (dd\_ux + dd\_uy) + w[i, j]  # Suhu baru dihitung dan ditambahkan ke suhu lama

t\_mean = np.mean(u)

print(f"t: {counter:.3f} s, Suhu rata-rata: {t\_mean:.2f} Celcius")

# Memperbarui plot dan menampilkan waktu simulasi

pcm.set\_array(u)

ax.set\_title(f"Distribusi Suhu t: {counter:.3f} s, suhu rata-rata={t\_mean:.2f}")

plt.show()

pada 2d saat waktu 1,5 sekon

import numpy as np

import matplotlib.pyplot as plt

# Mendefinisikan Variabel

a = 50  # Koefisien Difusivitas Termal [m^2/s]

panjang = 0.5  # Panjang plat [m]

waktu = 1.5  # Waktu simulasi [s]

node = 50  # Jumlah titik grid

dx = panjang / node  # Jarak antar titik grid pada x [mm]

dy = panjang / node  # Jarak antar titik grid pada y [mm]

dt = min(dx \*\* 2 / (4 \* a), dy \*\* 2 / (4 \* a))  # Ukuran langkah waktu [s] (pilih yang lebih kecil agar stabil)

t\_nodes = int(waktu / dt)  # Jumlah iterasi simulasi

u = np.zeros((node, node)) + 20  # Suhu awal plat [degC] (2 dimensi)

# Kondisi batas

u[0, :] = 0   # Suhu tepi kiri (variasi linear)

u[-1, :] = 100   # Suhu tepi kanan (variasi linear)

u[:, 0] = np.linspace(0, 100, node)  # Suhu tepi bawah (variasi linear)

u[:, -1] = np.linspace(0, 100, node)  # Suhu tepi atas (variasi linear)

# Visualisasi distribusi suhu awal

fig, ax = plt.subplots()

ax.set\_ylabel("y (cm)")

ax.set\_xlabel("x (cm)")

pcm = ax.pcolormesh(u, cmap=plt.cm.jet, vmin=0, vmax=100)

plt.colorbar(pcm, ax=ax)

# Simulasi

counter = 0

while counter < waktu:

    w = u.copy()  # Menyalin data suhu untuk perhitungan

    # Looping setiap titik grid kecuali batas

    for i in range(1, node - 1):

        for j in range(1, node - 1):

            # Menghitung perubahan suhu berdasarkan persamaan Laplace 2D (menggunakan tetangga terdekat)

            dd\_ux = (w[i - 1, j] - 2 \* w[i, j] + w[i + 1, j]) / dx \*\* 2

            dd\_uy = (w[i, j - 1] - 2 \* w[i, j] + w[i, j + 1]) / dy \*\* 2

            u[i, j] = dt \* a \* (dd\_ux + dd\_uy) + w[i, j]  # Suhu baru dihitung dan ditambahkan ke suhu lama

    pcm.set\_array(u)  # Memperbarui plot dan menampilkan waktu simulasi

    t\_mean = np.mean(u)

    counter += dt  # Menambah waktu simulasi

    print(f"t: {counter:.3f} s, Suhu rata-rata: {t\_mean:.2f} Celcius")

    ax.set\_title(f"Distribusi Suhu t: {counter:.3f} s, suhu rata-rata={t\_mean:.3f}")

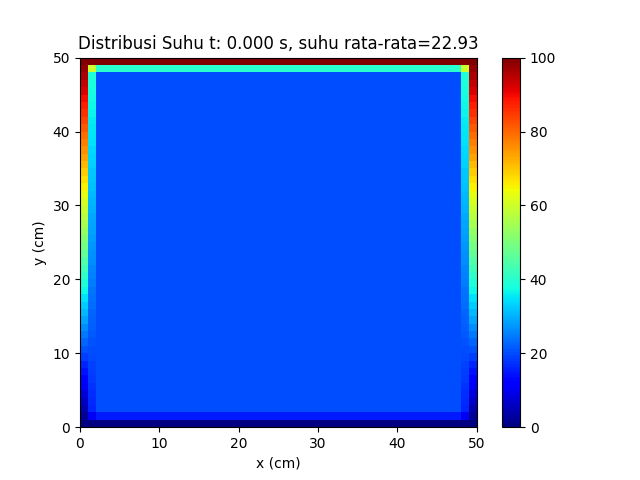
    plt.pause(0.1)

# Menampilkan suhu rata-rata pada waktu tertentu

suhu\_rata\_rata\_waktu\_tertentu = np.mean(u)

print(f"Suhu rata-rata pada t={waktu}s: {suhu\_rata\_rata\_waktu\_tertentu:.2f} Celcius")

plt.show()



plot saat waktu 0 sekon