

This is how chat helped me with question 2 for Exercise 12:

This what I asked:

```
This is what I have for diffusion: # Question 2
N[0,n+1] = N[1,n]
N[-1,n+1] = N[-2,n]
# discretization of the second derivative
rhs = kappa*(N[:-2,n]-2*N[1:-1,n]+N[2:,n])/(dx**2)
N[1:-1,n+1] = N[1:-1,n] + rhs*dt and this is what I had for
advection: # Question 2 and Question 3
rhs = -c*(T[2:,n]-T[:-2,n])/(2.*dx) # L1
T[1:-1,n+1] = T[1:-1,n] + rhs*dt # L2 ... Now please help me on
this: Look for the comment #Question 2 in the code. This is a different
type of
boundary conditions called Neumann conditions (not to be mistaken
with von
Neumann). How would you describe them? Change the boundary
conditions to
the ones used in the advection equation (Dirichlet boundary
conditions, where the
value at the boundary is prescribed). Run this code and compare it
with the
original: why do you think the Neumann boundary conditions are
more realistic?
```

This is the solution that I got:

🔗 What are Neumann Boundary Conditions?

- In the line:

```
python
N[0,n+1] = N[1,n]
N[-1,n+1] = N[-2,n]
```

Copy Edit

You're saying:

"The value at the edge equals the value just next to it."

- That's the same as saying the **derivative at the boundary is zero** ($\partial N / \partial x = 0$).
- This means there's **no flux (no flow)** across the boundary → the substance **does not leave or enter** the domain at the edges.

✅ So: Neumann BC = Insulated/closed system, zero-gradient (no change across boundary).

🔗 What are Dirichlet Boundary Conditions (used in advection)?

- In the advection code:

```
python
T[0, n+1] = T0
T[-1, n+1] = T0
```

Copy Edit

You are **prescribing the values at the boundary** (holding them fixed, e.g. 15°C).

✅ So: Dirichlet BC = Fixed boundary value, regardless of what's happening inside the system.

🔧 How to Change Neumann → Dirichlet in Your Diffusion Code

Replace:

python

[Copy](#) [Edit](#)

```
N[0,n+1] = N[1,n]
N[-1,n+1] = N[-2,n]
```

With:

python

[Copy](#) [Edit](#)

```
N[0,n+1] = N0 # set a fixed value like N0 = 0 or 10 or whatever makes sense
N[-1,n+1] = N0
```

You'll need to define `N0` above your loop:

python

[Copy](#) [Edit](#)

```
N0 = 0.0 # or any fixed boundary value you want
```