In [94]:
```
from gurobipy import *
import csv

##### 20 week Simulation ##########

# ----------------------------------
# Parameters
# ----------------------------------

# ----------------------------------
# WEEKLY PARAMETERS
# ----------------------------------


# Number of available vaccines for each week
V = {
    1: 150000,
    2: 150000,
    3: 150000,
    4: 150000,
    5: 150000,
    6: 150000,
    7: 150000,
    8: 150000,
    9: 150000,
    10: 150000,
    11: 150000,
    12: 150000,
    13: 150000,
    14: 150000,
    15: 150000,
    16: 150000,
    17: 150000,
    18: 150000,
    19: 150000,
    20: 150000
}

# Number of available Vaccinating staff for the week
E_V = {
    1: 205,
    2: 205,
    3: 205,
    4: 205,
    5: 205,
    6: 205,
    7: 205,
    8: 205,
    9: 205,
    10: 205,
    11: 205,
    12: 205,
    13: 205,
    14: 205,
```

```
    15: 205,
    16: 205,
    17: 205,
    18: 205,
    19: 205,
    20: 205
}

# Number of available Administrative staff for the week
E_A = {
    1: 789,
    2: 789,
    3: 789,
    4: 789,
    5: 789,
    6: 789,
    7: 789,
    8: 789,
    9: 789,
    10: 789,
    11: 789,
    12: 789,
    13: 789,
    14: 789,
    15: 789,
    16: 789,
    17: 789,
    18: 789,
    19: 789,
    20: 789
}

# Initial vaccine coverage C[i] in each county (as proportion vaccinated)
C = {
    "Franklin": 0.0,
    "Delaware": 0.0,
    "Licking": 0.0,
    "Fairfield": 0.0,
    "Union": 0.0,
    "Marion": 0.0,
    "Knox": 0.0,
    "Pickaway": 0.0,
    "Logan": 0.0,
    "Madison": 0.0,
    "Crawford": 0.0,
    "Morrow": 0.0,
    "Hardin": 0.0,
    "Fayette": 0.0,
    "Wyandot": 0.0
}

# Maximum available clinic square footage S[i] for each county
S = {
    "Franklin": 452082,
    "Delaware": 123766,
    "Licking": 27336,
```

```
    "Fairfield": 64526,
    "Union": 314164,
    "Marion": 55737,
    "Knox": 376007,
    "Pickaway": 55507,
    "Logan": 73185,
    "Madison": 49019,
    "Crawford": 50491,
    "Morrow": 144270,
    "Hardin": 30751,
    "Fayette": 159251,
    "Wyandot": 72137
}

#### Immutable Parameters #####

# County population P[i]
P = {
    "Franklin": 1356303, "Delaware": 237966, "Licking": 184898,
    "Fairfield": 167762, "Union": 71721, "Marion": 64976, "Knox": 63848,
    "Pickaway": 62158, "Logan": 46085, "Madison": 45531, "Crawford": 41626,
    "Morrow": 35927, "Hardin": 30402, "Fayette": 28782,"Wyandot": 21394
}

# Set of counties in Region 4
I = [
    "Franklin", "Delaware", "Licking", "Fairfield", "Union",
    "Marion", "Knox", "Pickaway", "Logan", "Madison",
    "Crawford", "Morrow", "Hardin", "Fayette", "Wyandot"
]

# Set of vaccination team types (administrator-to-vaccinator staffing ratios)
K = [
    "1to1",
    "3to1",
    "7to1"
]

# Number of staff in each team
n_V = {"1to1": 1, "3to1": 1, "7to1": 1}
n_A = {"1to1": 1, "3to1": 3, "7to1": 7}

# Hourly vaccination rates for each team
R = {"1to1": 6, "3to1": 19, "7to1": 30}

# Work hours per week
hours_per_week = 40

# Maximum number of people each team type can vaccinate per week
M = {k: vacc_per_hour[k] * hours_per_week for k in K}

# Square footage requirement per team
sqft = 2500

# ---------------------------------
# Results storage
```

```
# ----------------------------------
results = []

# ============================================================
#                  20-WEEK SIMULATION LOOP
# ============================================================

for week in range(1, 21):

    # --------------------------
    # Model
    # --------------------------
    m = Model("Ohio Region 4 Vaccine Planning - 20 Week Sim")
    m.Params.OutputFlag = 0

    # Decision variables
    t = m.addVars(I, K, vtype=GRB.INTEGER, lb=0, name="t")
    d = m.addVars(I, vtype=GRB.INTEGER, lb=0, name="d")
    z = m.addVars(I, vtype=GRB.INTEGER, lb=0, name="z")

    # Objective
    m.setObjective(quicksum(z[i] for i in I), GRB.MAXIMIZE)

    # Constraints

    # 1. Weekly vaccine supply
    m.addConstr(quicksum(d[i] for i in I) <= V[week], name="VaccineSupply")

    # 2. Vaccinator availability
    m.addConstr(quicksum(n_V[k] * t[i, k] for i in I for k in K) <= E_V[week],
                name="TotalVaccinators")

    # 3. Admin staff availability
    m.addConstr(quicksum(n_A[k] * t[i, k] for i in I for k in K) <= E_A[week],
                name="TotalAdmin")

    for i in I:

        # 4. Dose feasibility
        m.addConstr(z[i] <= d[i], name=f"FeasibleDoses[{i}]")

        # 5. Herd immunity cap
        m.addConstr(z[i] <= (0.8 - C[i]) * P[i], name=f"HerdImmunity[{i}]")

        # 6. Weekly participation limit
        m.addConstr(z[i] <= 0.1 * P[i] * (1 - C[i]), name=f"Participation[{i}]")

        # 7. Building capacity
        m.addConstr(sqft * quicksum(t[i, k] for k in K) <= S[i], name=f"Building[{i

        # 8. Throughput limitation
        m.addConstr(z[i] <= quicksum(M[k] * t[i, k] for k in K), name=f"Throughput[


    # 9. Fairness constraint
    for i in I:
```

```
        for j in I:

            # Only compare different counties
            if i != j:

                m.addConstr((C[i] * P[i] + z[i]) * P[j] - (C[j] * P[j] + z[j]) * P[
                m.addConstr((C[j] * P[j] + z[j]) * P[i] - (C[i] * P[i] + z[i]) * P[

# --------------------------
#  SOlve
# --------------------------

    # Solve
    m.optimize()

    # Extract weekly results
    opt_z = {i: z[i].X for i in I}
    opt_d = {i: d[i].X for i in I}
    opt_t = {(i, k): t[i, k].X for i in I for k in K}

    # Store results
    for i in I:
        coverage_start = C[i]
        coverage_end   = C[i] + opt_z[i] / P[i]

        # Compute staff sent to county i
        vacc_staff_i = sum(rV[k] * opt_t[(i, k)] for k in K)
        admin_staff_i = sum(rA[k] * opt_t[(i, k)] for k in K)

        results.append({
            "week": week,
            "county (i)": i,
            "coverage (C_i)": coverage_start,
            "updated coverage": coverage_end,
            "doses_sent (d_i)": opt_d[i],
            "vaccinated (z_i)": opt_z[i],
            "teams_1to1 (t_i,1to1)": opt_t[(i, "1to1")],
            "teams_3to1 (t_i,3to1)": opt_t[(i, "3to1")],
            "teams_7to1 (t_i,7to1)": opt_t[(i, "7to1")],
            "total vaccinators": vacc_staff_i,
            "total admin staff": admin_staff_i
        })

    # Update coverage for next week
    C = {i: C[i] + opt_z[i] / P[i] for i in I}

# ============================================================
#                  WRITE RESULTS
# ============================================================

with open("group22_output_20weeks.csv", "w", newline="") as f:
    writer = csv.DictWriter(f, fieldnames=results[0].keys())
    writer.writeheader()
    writer.writerows(results)

print("\nSaved results to group22_output_20weeks.csv")
```

```
Saved results to group22_output_20weeks.csv
```

In [ ]: