Treść Zadań

Zadanie 1: Problem transportowy

Rozwiązać problem minimalizacji kosztów transportu dla danych: supply = [30, 35, 60], demand = [25, 25, 40, 30], costs = [[5, 5, 6, 2], [1, 7, 4, 2], [6, 3, 2, 1]].

Zadanie 2: Problem optymalizacji liniowej

Rozwiązać problem minimalizacji funkcji celu: -x1 - 2x2 + 2x3 przy ograniczeniach: 2x1 - x2 + x3 = 10, x1 - x2 = 2 oraz xj >= 0 dla j = 1, 2, 3.

Kod Źródłowy

Poniżej zamieszczono kod źródłowy dla obu zadań.

------------------- ZADANIE 1 -------------------

```python
from pulp import LpProblem, LpMinimize, LpVariable, lpSum

# Zadanie z Kolokwium 2024 Wersja 4
#      | 25 | 25 | 40 | 30
#   ----------------------
#   30 | 5  | 5  | 6  | 2
#   35 | 1  | 7  | 4  | 2
#   60 | 6  | 3  | 2  | 1
# Dane problemu
supply = [30, 35, 60]  # Dostępność z magazynów

demand = [25, 25, 40, 30]  # Zapotrzebowanie odbiorców

# Koszty transportu
costs = [
    [5, 5, 6, 2],
    [1, 7, 4, 2],
    [6, 3, 2, 1]
]

# Tworzenie problemu minimalizacyjnego
problem = LpProblem("Zagadnienie_Transportowe", LpMinimize)

# Zmienne decyzyjne
decision_variables = [
    [LpVariable(f"x_{i}_{j}", lowBound=0) for j in range(len(demand))]
    for i in range(len(supply))
]

# Funkcja celu (minimalizacja kosztów)
problem += lpSum(
    decision_variables[i][j] * costs[i][j]
    for i in range(len(supply))
    for j in range(len(demand))
), "Koszty Transportu"

# Ograniczenia podaży
for i in range(len(supply)):
    problem += lpSum(decision_variables[i][j] for j in range(len(demand))) <= supply[i], f"Ograniczenie_podazy_{i}"

# Ograniczenia popytu
for j in range(len(demand)):
    problem += lpSum(decision_variables[i][j] for i in range(len(supply))) == demand[j], f"Ograniczenie_popytu_{j}"

# Rozwiązywanie problemu
problem.solve()

# Wyniki
print("Status:", problem.status)
print("Minimalny koszt transportu:", problem.objective.value())
for i in range(len(supply)):
    for j in range(len(demand)):
        print(f"Transport z magazynu {i} do odbiorcy {j}: {decision_variables[i][j].varValue}")
```


```
from pulp import LpProblem, LpMinimize, LpVariable, lpSum

# Zadanie z Kolokwium 2024 Wersja 4
#     | 25 | 25 | 40 | 30
#   ----------------------
#   30 | 5  | 5  | 6  | 2
#   35 | 1  | 7  | 4  | 2
#   60 | 6  | 3  | 2  | 1
# Dane problemu
supply = [30, 35, 60]  # Dostępność z magazynów

demand = [25, 25, 40, 30]  # Zapotrzebowanie odbiorców
```

```
# Koszty transportu
costs = [
    [5, 5, 6, 2],
    [1, 7, 4, 2],
    [6, 3, 2, 1]
]

# Tworzenie problemu minimalizacyjnego
problem = LpProblem("Zagadnienie_Transportowe", LpMinimize)

# Zmienne decyzyjne
decision_variables = [
    [LpVariable(f"x_{i}_{j}", lowBound=0) for j in range(len(demand))]
    for i in range(len(supply))
]

# Funkcja celu (minimalizacja kosztów)
problem += lpSum(
    decision_variables[i][j] * costs[i][j]
    for i in range(len(supply))
    for j in range(len(demand))
), "Koszty Transportu"

# Ograniczenia podaży
for i in range(len(supply)):
    problem += lpSum(decision_variables[i][j] for j in range(len(demand))) <= supply[i], f"Ograniczenie_podazy_{i}"

# Ograniczenia popytu
for j in range(len(demand)):
    problem += lpSum(decision_variables[i][j] for i in range(len(supply))) == demand[j], f"Ograniczenie_popytu_{j}"

# Rozwiązywanie problemu
problem.solve()

# Wyniki
print("Status:", problem.status)
print("Minimalny koszt transportu:", problem.objective.value())
for i in range(len(supply)):
    for j in range(len(demand)):
        print(f"Transport z magazynu {i} do odbiorcy {j}: {decision_variables[i][j].varValue}")
```

------------------- ZADANIE 2 -------------------

```python
from scipy.optimize import linprog

# Zadanie z Kolokwium 2024 Wersja 4
# Funkcja celu: -x1 - 2x2 + 2x3 -> MIN
# Ograniczenia:
# 2x1 - x2 + x3 = 10
# x1 - x2 = 2
# xj >= 0 dla j = 1, 2, 3

# Dane
dane_funkcji_celu = [-1, -2, 2]
macierz_rownan = [
    [2, -1, 1],
    [1, -1, 0]
]
wartosci_rownan = [
    10,
    2
]

# Ograniczenia zmiennych decyzyjnych
ograniczenia = [(0, float('inf'))] * 3

# Rozwiazywanie problemu optymalizacji
wynik = linprog(
    c=dane_funkcji_celu,
    A_eq=macierz_rownan,
    b_eq=wartosci_rownan,
    bounds=ograniczenia,
    method="highs"
)

# Wynik
if wynik.success:
    print("Optymalizacja zakonczona sukcesem!")
    print(f"Minimalna wartosc funkcji celu (Zmin): {wynik.fun}")
    print(f"Zmienne decyzyjne: {wynik.x}")
else:
    print("Optymalizacja nie powiodla sie.")
    print(f"Powod: {wynik.message}")
```

```python
from scipy.optimize import linprog

# Zadanie z Kolokwium 2024 Wersja 4
# Funkcja celu: -x1 - 2x2 + 2x3 -> MIN
# Ograniczenia:
# 2x1 - x2 + x3 = 10
# x1 - x2 = 2
# xj >= 0 dla j = 1, 2, 3

# Dane
dane_funkcji_celu = [-1, -2, 2]
macierz_rownan = [
    [2, -1, 1],
    [1, -1, 0]
]
wartosci_rownan = [
    10,
    2
]

# Ograniczenia zmiennych decyzyjnych
ograniczenia = [(0, float('inf'))] * 3
```

```
# Rozwiazywanie problemu optymalizacji
wynik = linprog(
    c=dane_funkcji_celu,
    A_eq=macierz_rownan,
    b_eq=wartosci_rownan,
    bounds=ograniczenia,
    method="highs"
)

# Wynik
if wynik.success:
    print("Optymalizacja zakonczona sukcesem!")
    print(f"Minimalna wartosc funkcji celu (Zmin): {wynik.fun}")
    print(f"Zmienne decyzyjne: {wynik.x}")
else:
    print("Optymalizacja nie powiodla sie.")
    print(f"Powod: {wynik.message}")
```

Wyniki

Wyniki z wykonania kodu obejmują minimalne koszty transportu oraz optymalne wartości zmiennych decyzyjnych. Poniżej zamieszczono zrzuty ekranu z wynikami.

Zadanie Transportowe

```
At line 2 NAME          MODEL
At line 3 ROWS
At line 12 COLUMNS
At line 49 RHS
At line 57 BOUNDS
At line 58 ENDATA
Problem MODEL has 7 rows, 12 columns and 24 elements
Coin0008I MODEL read with 0 errors
Option for timeMode changed from cpu to elapsed
Presolve 7 (0) rows, 12 (0) columns and 24 (0) elements
0  Obj 0 Primal inf 120 (4)
6  Obj 250
Optimal - objective value 250
Optimal objective 250 - 6 iterations time 0.002
Option for printingOptions changed from normal to all
Total time (CPU seconds):       0.00   (Wallclock seconds):       0.00

Status: 1
Minimalny koszt transportu: 250.0
Transport z magazynu 0 do odbiorcy 0: 0.0
Transport z magazynu 0 do odbiorcy 1: 0.0
Transport z magazynu 0 do odbiorcy 2: 0.0
Transport z magazynu 0 do odbiorcy 3: 30.0
Transport z magazynu 1 do odbiorcy 0: 25.0
Transport z magazynu 1 do odbiorcy 1: 0.0
Transport z magazynu 1 do odbiorcy 2: 5.0
Transport z magazynu 1 do odbiorcy 3: 0.0
Transport z magazynu 2 do odbiorcy 0: 0.0
Transport z magazynu 2 do odbiorcy 1: 25.0
Transport z magazynu 2 do odbiorcy 2: 35.0
Transport z magazynu 2 do odbiorcy 3: 0.0
Press any key to continue . . .
```

Funkcja celu: -x1 - 2x2 + 2x3 -> MIN

Ograniczenia:

2x1 - x2 + x3 = 10

x1 - x2 = 2

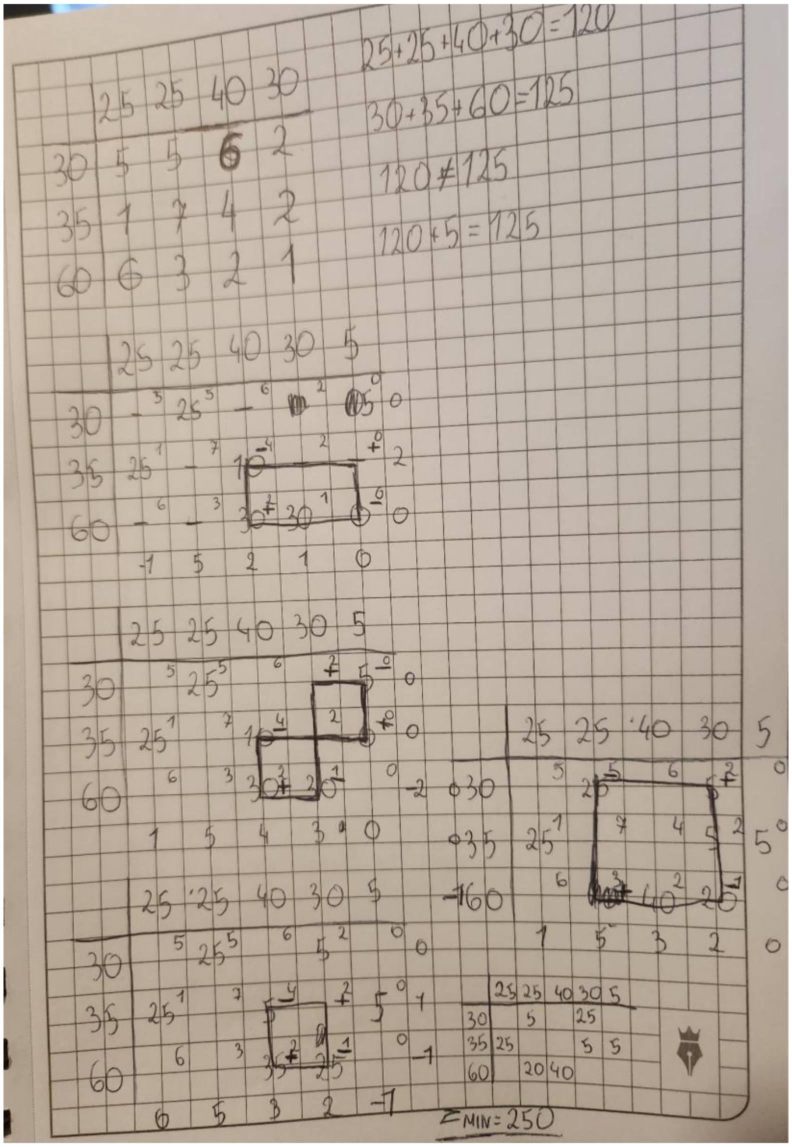
xj >= 0 dla j = 1, 2, 3

```
Optymalizacja zakonczona sukcesem!
Minimalna wartosc funkcji celu (Zmin): -20.0
Zmienne decyzyjne: [8. 6. 0.]
Press any key to continue . . .
```

Rozwiązanie na Kolokwium

Poniżej zamieszczono zdjęcia przedstawiające rozwiązanie na kolokwium.

Zadanie 1 (Transportowe)



Zadanie 2

$-x - 2x_2 + 2x_3 \to MIN$

$2x_1 - x_2 + x_3 = 10$

$x_1 - x_2 = 2$

$2x_1 - x_2 + x_3 + x_4 = 10$

$x_1 - x_2 + x_5 = 2$

$-x_1 - 2x_2 + 2x_3 + Mx_4 + Mx_5 \to MIN$

			-1	-2	2	-M	-M	
B	C_B	b	A_1	A_2	A_3	A_4	A_5	
A_4	-M	10	2	-1	1	1	0	5
A_5	-M	2	1	-1	0	0	1	2
	Δ	12M	1+3M	2-2M	-2+2M	0	0	
A_4	-M	6	0	1	1	1	-2	6
A_1	-1	2	1	-1	0	0	1	-
	Δ	2+6M	0	3+M	-2+M	0	-1-3M	
A_2	-2	6	0	1	1	1	-2	
A_1	-1	8	1	0	1	1	-1	
	Δ	-20	0	0	-5	-3-M	5-M	

$x_1 = 8 \quad x_3 = 0$

$x_2 = 6 \quad Z_{MIN} = -20$

Zakończenie

Sprawozdanie przedstawia rozwiązanie dwóch zadań optymalizacyjnych. Oba problemy zostały poprawnie zaimplementowane i rozwiązane w środowisku Python. Uzyskane wyniki są zgodne z oczekiwaniami.

Data: 2025-01-25

Autor: ezehe