例 1.21 最小费用流问题实现

算法功能:寻找实现流量需要的最小费用流
修改参数：start_nodes：路径开始节点
end_nodes:路径终止节点
capacities:路径流量容量
unit_costs:路径所需费用
supplies:各节点流量要求
输出：最小费用
流量路径及费用表格
"""

```
from ortools.graph import pywrapgraph
start_nodes = [ 0, 0,  1, 1,  2,  3, 3]  # 路径开始节点
end_nodes   = [ 1, 3,  2, 4,  4,  2, 4]  # 路径终止节点
capacities  = [ 4, 3,  4, 5,  3,  2, 1  ]  # 路径流量容量
unit_costs  = [ 1, 2,  1, 4,  2,  2, 4]  # 路径所需费用

supplies = [7, 0, 0, 0, -7]  # 各节点流量要求，正为输出流量，负为输入流量（此例子为 0 节点输出 7 流量到 4 节点）
min_cost_flow = pywrapgraph.SimpleMinCostFlow()
# 添加每条路径
for i in range(0, len(start_nodes)):
    min_cost_flow.AddArcWithCapacityAndUnitCost(start_nodes[i], end_nodes[i],
                                                capacities[i], unit_costs[i])

for i in range(0, len(supplies)):
    min_cost_flow.SetNodeSupply(i, supplies[i])
if min_cost_flow.Solve() == min_cost_flow.OPTIMAL:
    print('Minimum cost:', min_cost_flow.OptimalCost())
    print('')
    print('  Arc    Flow / Capacity  Cost')
    for i in range(min_cost_flow.NumArcs()):
      cost = min_cost_flow.Flow(i) * min_cost_flow.UnitCost(i)
      print('%1s -> %1s   %3s  / %3s       %3s' % (
          min_cost_flow.Tail(i),
          min_cost_flow.Head(i),
          min_cost_flow.Flow(i),
          min_cost_flow.Capacity(i),
          cost))
else:
    print('There was an issue with the min cost flow input.')
```