# Решение задачи о прокате оборудования в непрерывном времени

# Ввод параметров  
print("Введите параметры задачи о прокате оборудования в непрерывном времени:\n")  
lambda\_1 = float(input("Введите λ1 (интенсивность отказов при нормальной эксплуатации, λ1 < λ2): "))  
lambda\_2 = float(input("Введите λ2 (интенсивность отказов при жесткой эксплуатации): "))  
mu\_1 = float(input("Введите μ1 (интенсивность восстановления при обычном ремонте): "))  
mu\_2 = float(input("Введите μ2 (интенсивность восстановления при ускоренном ремонте, μ1 < μ2): "))  
  
c11 = float(input("Введите c11 (доход при нормальной эксплуатации, c11 < c12): "))  
c12 = float(input("Введите c12 (доход при жесткой эксплуатации): "))  
c21 = float(input("Введите c21 (издержки при обычном ремонте, c21 < c22): "))  
c22 = float(input("Введите c22 (издержки при ускоренном ремонте): "))  
  
if not (lambda\_1 < lambda\_2):  
 print("Условие λ1 < λ2 не выполнено.")  
if not (mu\_1 < mu\_2):  
 print("Условие μ1 < μ2 не выполнено.")   
if not (c11 < c12):  
 print("Условие c11 < c12 не выполнено.")   
if not (c21 < c22):  
 print("Условие c21 < c22 не выполнено.")

Введите параметры задачи о прокате оборудования в непрерывном времени:

def compute\_stationary\_distribution(lambda\_i, mu\_j):  
 pi\_1 = mu\_j / (lambda\_i + mu\_j)  
 pi\_2 = lambda\_i / (lambda\_i + mu\_j)  
 return pi\_1, pi\_2

def compute\_average\_reward(pi\_1, pi\_2, c\_1i, c\_2j):  
 return pi\_1 \* c\_1i - pi\_2 \* c\_2j

def main():  
 strategies = []  
 # Стратегия f11: нормальная эксплуатация, обычный ремонт  
 pi1, pi2 = compute\_stationary\_distribution(lambda\_1, mu\_1)  
 avg\_reward = compute\_average\_reward(pi1, pi2, c11, c21)  
 strategies.append(('f11', avg\_reward))  
  
 # Стратегия f12: нормальная эксплуатация, ускоренный ремонт  
 pi1, pi2 = compute\_stationary\_distribution(lambda\_1, mu\_2)  
 avg\_reward = compute\_average\_reward(pi1, pi2, c11, c22)  
 strategies.append(('f12', avg\_reward))  
  
 # Стратегия f21: жесткая эксплуатация, обычный ремонт  
 pi1, pi2 = compute\_stationary\_distribution(lambda\_2, mu\_1)  
 avg\_reward = compute\_average\_reward(pi1, pi2, c12, c21)  
 strategies.append(('f21', avg\_reward))  
  
 # Стратегия f22: жесткая эксплуатация, ускоренный ремонт  
 pi1, pi2 = compute\_stationary\_distribution(lambda\_2, mu\_2)  
 avg\_reward = compute\_average\_reward(pi1, pi2, c12, c22)  
 strategies.append(('f22', avg\_reward))  
  
 # Определение оптимальной стратегии  
 optimal\_strategy = max(strategies, key=lambda x: x[1])  
  
 # Вывод результатов  
 print("\nСредние доходы для каждой стратегии:")  
 for strat, reward in strategies:  
 print(f"Стратегия {strat}: Средний доход = {reward:.2f}")  
  
 print(f"\nОптимальная стратегия: {optimal\_strategy[0]} с средним доходом = {optimal\_strategy[1]:.2f}")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Средние доходы для каждой стратегии:  
Стратегия f11: Средний доход = 75.00  
Стратегия f12: Средний доход = 83.64  
Стратегия f21: Средний доход = 75.00  
Стратегия f22: Средний доход = 96.92  
  
Оптимальная стратегия: f22 с средним доходом = 96.92