In [3]:
```
# Practice 4: Uses dictionary values
employees = [
    {
        "ID": "BS190920",
        "Name": "Benjamin Samuels",
        "Rate": 51.50,
        "Hours/week": 45 * 2,
        "Unionized": True
    },
    {
        "ID": "PF921231",
        "Name": "Patrizia Florence",
        "Rate": 35.00,
        "Hours/week": 23 * 2,
        "Unionized": False
    },
    {
        "ID": "DL109960",
        "Name": "Dario Libano",
        "Rate": 14.00,
        "Hours/week": 50 * 2,
        "Unionized": True
    }
]

def biweekly(hours, rate):
    # Calculate regular and overtime hours
    regular = min(hours, 80)
    overtime = max(hours - 80, 0)
    if hours <= 80:
        regular_rate = regular * rate
        overtime_rate = 0
    else:
        regular_rate = 80 * rate
        overtime_rate = overtime * rate * 1.5
    total = regular_rate + overtime_rate
    return total, overtime_rate, regular_rate

def union_fees(total, is_unionized):
    # Calculate union fees based on total earnings
    if is_unionized:
        union_fee = 0.01 * total
    else:
        union_fee = 0
    return round(union_fee, 2)

def federal_tax(total):
    # Calculate federal tax based on total annual income
    if total <= 11600:
        federal_tax_amount = 0.10 * total
    elif 11601 <= total <= 47150:
        federal_tax_amount = (11600 * 0.10) + ((total - 11600) * 0.12)
    elif 47151 <= total <= 100525:
        federal_tax_amount = (11600 * 0.10) + ((47150 - 11600) * 0.12) + ((tot
    elif 100526 <= total <= 191950:
        federal_tax_amount = (11600 * 0.10) + ((47150 - 11600) * 0.12) + ((100
    return round(federal_tax_amount, 2)

def other_deductions(total, rate):
    # Calculate other deductions such as retirement, state tax, social securit
    annual_salary = rate * 80 * 26  # Assuming 80 hours per biweekly period
    retirement = 0.045 * total
    state_tax = 0.06 * total
    if annual_salary >= 168600:
        social_security = 10453.20
    else:
        social_security = 0.062 * total
    if total > 200000:
        medicaid = 0.0145 * total + 0.009 * (total - 200000)
    else:
        medicaid = 0.0145 * total
    return round(retirement, 2), round(state_tax, 2), round(social_security, 2

def total_deductions(total, is_unionized, rate):
    # Calculate total deductions including union fees, federal tax, retirement
    union_fee = union_fees(total, is_unionized)
    federal_tax_amount = federal_tax(total)  # No need to multiply by 26 here
    retirement, state_tax, social_security, medicaid = other_deductions(total,
    total_deduction = union_fee + federal_tax_amount + retirement + state_tax
    return total_deduction

def net_pay(biweekly_total, total_deductions):
    # Calculate net pay after deductions
    net_pay = biweekly_total - total_deductions  # No need to multiply by 26 h
    return round(net_pay, 2)

print("\nFor Bi-weekly Payments:")
for employee in employees:
    print("Employee ID:", employee["ID"])
    print("Name:", employee["Name"].title())
    total_payment, overtime_payment, regular_payment = biweekly(employee["Hour
    print("Wages total:")
    print("\tRegular time: $", round(regular_payment, 2))
    print("\tOvertime: $", round(overtime_payment, 2))
    print("\nDeductions:")
    print("\tUnion fees: $", round(union_fees(total_payment, employee["Unioniz
    print("\tRetirement fund: $", round(0.045 * total_payment, 2))
    print("\tState taxes: $", round(0.06 * total_payment, 2))
    print("\tFederal taxes: $", round(federal_tax(total_payment), 2))  # Corre
    print("\tSocial Security: $", round(0.062 * total_payment, 2))
    print("\tMedicaid: $", round(0.0145 * total_payment, 2))

    # Calculate net pay
    total_deduction = total_deductions(total_payment, employee["Unionized"], e
    net_pay_amount = net_pay(total_payment, total_deduction)
    print("\tNet-pay: $", round(net_pay_amount, 2))
    print("-------------------------------------------------")

print("For Annual Gross Pay:")
for employee in employees:
    print("Employee ID:", employee["ID"])
    print("Name:", employee["Name"].title())
    total_payment, overtime_payment, regular_payment = biweekly(employee["Hour
    total_payment_accumulated = (total_payment * 26) + (overtime_payment * 26)
    print("Wages total accum: $", round(total_payment_accumulated, 2))

    # Calculate and print deductions
    union_fee_accum = union_fees(total_payment, employee["Unionized"]) * 26  #
    retirement_accum = 0.045 * total_payment_accumulated
    state_tax_accum = 0.06 * total_payment_accumulated
    federal_tax_accum = federal_tax(total_payment_accumulated)  # Corrected fe
    social_security_accum = 0.062 * total_payment_accumulated
    medicaid_accum = 0.0145 * total_payment_accumulated
    total_deduction_accumulated = total_deductions(total_payment_accumulated,
    net_pay_accumulated = net_pay(total_payment_accumulated, total_deduction_a

    print("\nDeductions accum:")
    print("\tUnion fees accum: $", round(union_fee_accum, 2))
    print("\tRetirement fund accum: $", round(retirement_accum, 2))
    print("\tState taxes accum: $", round(state_tax_accum, 2))
    print("\tFederal taxes accum: $", round(federal_tax_accum, 2))
    print("\tSocial Security accum: $", round(social_security_accum, 2))
    print("\tMedicaid accum: $", round(medicaid_accum, 2))
    print("\tNet-pay accumulated: $", round(net_pay_accumulated, 2))
    print("------------------------------------------------")
```

```
For Bi-weekly Payments:
Employee ID: BS190920
Name: Benjamin Samuels
Wages total:
        Regular time: $ 4120.0
        Overtime: $ 772.5

Deductions:
        Union fees: $ 48.93
        Retirement fund: $ 220.16
        State taxes: $ 293.55
        Federal taxes: $ 489.25
        Social Security: $ 303.33
        Medicaid: $ 70.94
        Net-pay: $ 3466.34
------------------------------------------------
Employee ID: PF921231
Name: Patrizia Florence
Wages total:
        Regular time: $ 1610.0
        Overtime: $ 0

Deductions:
        Union fees: $ 0
        Retirement fund: $ 72.45
        State taxes: $ 96.6
        Federal taxes: $ 161.0
        Social Security: $ 99.82
        Medicaid: $ 23.35
        Net-pay: $ 1156.78
------------------------------------------------
Employee ID: DL109960
Name: Dario Libano
Wages total:
        Regular time: $ 1120.0
        Overtime: $ 420.0

Deductions:
        Union fees: $ 15.4
        Retirement fund: $ 69.3
        State taxes: $ 92.4
        Federal taxes: $ 154.0
        Social Security: $ 95.48
        Medicaid: $ 22.33
        Net-pay: $ 1091.09
------------------------------------------------
For Annual Gross Pay:
Employee ID: BS190920
Name: Benjamin Samuels
Wages total accum: $ 147290.0

Deductions accum:
        Union fees accum: $ 1272.18
        Retirement fund accum: $ 6628.05
        State taxes accum: $ 8837.4
        Federal taxes accum: $ 28392.1
        Social Security accum: $ 9131.98
        Medicaid accum: $ 2135.7
        Net-pay accumulated: $ 90691.87
------------------------------------------------
Employee ID: PF921231
Name: Patrizia Florence
Wages total accum: $ 41860.0

Deductions accum:
        Union fees accum: $ 0
        Retirement fund accum: $ 1883.7
        State taxes accum: $ 2511.6
        Federal taxes accum: $ 4791.2
        Social Security accum: $ 2595.32
        Medicaid accum: $ 606.97
        Net-pay accumulated: $ 29471.21
------------------------------------------------
Employee ID: DL109960
Name: Dario Libano
Wages total accum: $ 50960.0

Deductions accum:
        Union fees accum: $ 400.4
        Retirement fund accum: $ 2293.2
        State taxes accum: $ 3057.6
        Federal taxes accum: $ 6264.2
        Social Security accum: $ 3159.52
        Medicaid accum: $ 738.92
        Net-pay accumulated: $ 34936.96
------------------------------------------------
```

```
In [ ]:
```