```python
import numpy as np

# (A) RDT&E COSTS
# Parameters Variables
######################################################################## FROM RFP
#current year (year our technology level is considered at - note our date of entry into service is 2035)
t_year = 2025
#base year (ideally use 1989 like the text book)
b_year = 1989
#no. of total engines per aircraft
n_engines = 1
#production quantity
Q = 1000
#total production quantity times number of engines per aircraft
N_eng_total = Q * n_engines
# no. of turboprop engines
n_engines_turbo = 1
######################################################################## AIRCRAFT SPECIFIC VARIABLES
#maximum velocity (knots)
V = 213
#engine maximum thrust (lb)
T_max = 500
#max mach number
M_max = 0.322
#turbine inlet temperature (Rankine)
T_turbine_inlet = 3600
#Range in nmi
R = 530
########################################################################HOURLY RATE AND OTHER EQUATIONS
#Cost escalation factor (t_CEF/b_CEF)
CEF = (5.17053 + 0.104981 * (t_year - 2006))/(5.17053 + 0.104981 * (b_year - 2006))
#Engineering hourly rate (wrap rate - salary plus all other costs like benefits and administrative costs - typically the salary is half the wrap)
R_engineer = 2.576 * 2025 - 5058
#Tooling hourly rate
R_tooling = 2.883 * 2025 - 5666
#Quality control hourly rate
R_qc = 2.60 * 2025 - 5112
#Manufacturing hourly rate
R_manufacturing = 2.316 * 2025 - 4552

########################################################################RELATIVELY CONSTANT VARIABLES
# Number of flight test aircraft -> Raymer
FTA = 4
#Airline Factor -> Meta
AF = 0.8
#Route Factor -> Meta
K = 2.75
#Total takeoff shaft horsepower (all engines added up)
SHP_TO = 620
#Mission time (fuel weight / specfic fuel consumption / shaft horsepower)
mission_time = 642.34 / .6 / 1295
#Block time in hrs (Total time aircraft is in use for mission - from wheel block removal to wheel block placement - mission time plus a half hour buffer)
tb = .5 + mission_time
#Hours per year used
year_use = 344
#Price per gallon of fuel
P_f = 6.56
#fuel density (lbs/gal)
rho_f = 6.7
#Price of electricity in $/kWh (divide by 1000 for $/Wh)
P_elec = 0.43 / 1000
```

```
#Specific energy of battery in Wh/kg (divide by 2.205 for Wh/lb) (0 is no batteries)
#e_elec = e_b / 2.205
#Maintenance labor cost in USD/hr for the year of interest
RL = 155
# no. of hours between engine overhauls (usually between 3000 and 5000)
H_em = 5000
#Aircraft residual value factor (estimated over lifetime)
K_depreciation = 0.1
#Number of years the aircraft is used
n = 40
#hull insurance rate, usually assumed to be 2%
IR_a = 0.02
#Electric motor horsepower
motor_hp = 1000
#Battery storage
battery_kWh = 60
########################################################################### FROM WEIGHT CALC
#empty weight (lb)
W_e = 3981
#Fuel Weight (lb)
W_f = 693
#Battery weight (lb)
#W_b = battery_weight
#Airframe weight (Empty weight minus engine weight, battery weight, and motor weight)
W_A = W_e  - SHP_TO ** (0.9306) * 10 **(-0.1205)
#Max Takeoff Weight (lb)
MTOW = 8864

##################################################################################
# Define the functions for various cost calculations based on the provided formulas

#Engineering Hours
def engineering_hours(W_e, Q):
   engineering_hours = 4.86 * (W_e ** 0.777) * (V ** 0.894) * (Q ** 0.163)
   return engineering_hours


#Toolings Hours
def tooling_hours(W_e, V, Q):
   tooling_hours = 5.99 * (W_e ** 0.777) * (V ** 0.696) * (Q ** 0.263)
   return tooling_hours


#Manufacturing Hours
def manufacturing_hours(W_e, V, Q):
   manufacturing_hours = 7.37 * (W_e ** 0.82) * (V ** 0.484) * (Q ** 0.641)
   return manufacturing_hours


#QC Hours
def qc_hours(manufacturing_hours):
   qc_hours = 0.133 * manufacturing_hours
   return qc_hours

#Total RDT&E Cost
def RDTE_cost(engineering_hours, R_engineer, tooling_hours, R_tooling, manufacturing_hours,
R_manufacturing, qc_hours, R_qc):
    RDTE_cost = engineering_hours * R_engineer + tooling_hours * R_tooling +
manufacturing_hours * R_manufacturing + qc_hours * R_qc
    return RDTE_cost

engineering_hours = engineering_hours(W_e, Q)
tooling_hours = tooling_hours(W_e, V, Q)
manufacturing_hours = manufacturing_hours(W_e, V, Q)
qc_hours = qc_hours(manufacturing_hours)
RDTE_cost = RDTE_cost(engineering_hours, R_engineer, tooling_hours, R_tooling,
manufacturing_hours, R_manufacturing, qc_hours, R_qc)
```

```
print(f"The RDT&E cost to produce {Q} aircraft is ${RDTE_cost:,.2f}.\n")
# (B) Flyaway/Production

#Development Support Cost
def development_support_cost(W_e, V):
   development_support_cost = 45.42 * (W_e ** 0.630) * (V ** 1.3)
   return development_support_cost


#Flight Test Cost
def flight_test_cost(W_e, V, FTA):
   flight_test_cost = 1243.03 * (W_e ** 0.325) * (V ** 0.822) * (FTA ** 1.21)
   return flight_test_cost


#Manufacturing Materials Cost
def manufacturing_materials_cost(W_e, V, Q):
   manufacturing_materials_cost = 11.0 * (W_e ** 0.921) * (V ** 0.621) * (Q ** 0.799)
   return manufacturing_materials_cost


#Engineering Production Cost per engine
def engine_production_cost(T_max, M_max, T_turbine_inlet):
   engine_production_cost = 1548 * (0.043 * T_max + 243.25 * M_max) + 0.969 * T_turbine_inlet
- 2228
   return engine_production_cost

#Total flyaway cost (neglects avionics for now)
def C_flyaway(development_support_cost, flight_test_cost, manufacturing_materials_cost,
engine_production_cost, N_eng_total):
    C_flyaway = development_support_cost + flight_test_cost + manufacturing_materials_cost +
engine_production_cost * N_eng_total
    return C_flyaway / Q

#Total Aircraft Cost (to recoup RDT&E and Production)

def C_aircraft(RDTE_cost, Q, C_flyaway):
    C_aircraft = RDTE_cost / Q + C_flyaway
    return C_aircraft

#Engine cost per plane
def C_engine(SHP_TO, CEF):
    C_engine = 10 ** (2.5262 + 0.9465 * np.log10(SHP_TO)) * CEF
    return C_engine

#Airframe cost per plane (used for later cost calculations)
def C_airframe(C_aircraft, C_engine):
    C_airframe = C_aircraft - C_engine
    return C_airframe

development_support_cost = development_support_cost(W_e, V)
flight_test_cost = flight_test_cost(W_e, V, FTA)
manufacturing_materials_cost = manufacturing_materials_cost(W_e, V, Q)
engine_production_cost = engine_production_cost(T_max, M_max, T_turbine_inlet)
C_flyaway = C_flyaway(development_support_cost, flight_test_cost, manufacturing_materials_cost,
engine_production_cost, N_eng_total)
C_aircraft = C_aircraft(RDTE_cost, Q, C_flyaway)
C_engine = C_engine(SHP_TO, CEF)
C_airframe = C_airframe(C_aircraft, C_engine)


print(f"The flyaway cost is ${C_flyaway:,.2f}. \n"
      f"To make a 10% profit on the production cost the plane should cost
${1.1*C_flyaway:,.2f}.\n"
      f"To make a 10% profit on the total plane cost (production + RDT&E) the plane should
cost ${1.1 * C_aircraft:,.2f}.\n")
```

```
# (C) DOC: Direct Operating Costs
# DOC = COC + FOC

# Crew
def C_crew(AF, K, MTOW, tb, CEF):
    C_crew = AF * (K * (MTOW) ** 0.4 * tb) * CEF
    return C_crew

# Attendants (Not applicable)

# Fuel (multiplied by number of missions)
def C_fuel(W_f, P_f, rho_f):
    C_fuel = 1.02 * W_f * (P_f/rho_f)
    return C_fuel

# For hybrid electric propulsion (multiplied by number of missions)
'''def C_electric(W_b, P_elec, e_elec):
    C_electric = 1.05 * W_b * P_elec * e_elec
    return C_electric'''

# Oil (neglected for now)
'''def C_oil(W_f, tb, P_oil, rho_oil)
W_oil = 0.0125 * W_f * (tb/100)
C_oil = 1.02 * W_oil * (P_oil/rho_oil)'''

# Landing Fees
def C_airport(MTOW, CEF):
    C_airport = 1.5 *(MTOW/1000) * CEF
    return C_airport

# Navigation Fees
def C_navigation(CEF, R, tb, MTOW):
    C_navigation = 0.5 * CEF * ((1.852 * R)/(tb)) * ((0.00045359237 * MTOW)/(50)) ** 0.5
    return C_navigation

# Airframe Maintenance
def C_AirMain(W_A, RL, CEF, C_airframe, tb):
    C_ML = 1.03 * (3+ (0.067 * W_A)/(1000))* RL
    C_MM = 1.03 * (30 * CEF) + 0.79 * (10 ** (-5)) * C_airframe
    C_AirMain = (C_ML + C_MM) * tb
    return C_AirMain

# Engine Maintenance labor cost
def C_ML_engine(T_max, tb, RL):
    C_ML_engine = (0.645 + (0.05 * T_max/(10 ** 4)))* (0.566 + 0.434/tb)* RL
    return C_ML_engine

#Engine maintenance material cost
def C_MM_engine(T_max, tb, CEF):
    C_MM_engine = ((25 + (18*T_max/(10**4))) * (0.62 + 0.38/tb)) * (CEF)
    return C_MM_engine

# for turboprop engines
def C_ML_turbo(SHP_TO, n_engines_turbo, H_em, RL):
    C_ML_turbo = 1.03 * 1.3 *(0.4956 + 0.0532 * (SHP_TO/n_engines_turbo)/(1000)*
((1100)/(H_em))+ 0.1) * RL
    return C_ML_turbo

#Total engine maintenance
def C_EngMain(n_engines, n_engines_turbo, C_ML_engine, C_MM_engine, C_ML_turbo, tb):
    C_EngMain = ((n_engines - n_engines_turbo) * C_ML_engine + n_engines * C_MM_engine +
n_engines_turbo * C_ML_turbo) * tb
    return C_EngMain

#Electric motor cost
'''def C_motors(motor_hp):
    C_motors = 150*motor_hp
```

```python
    return C_motors'''

#Battery cost
'''def C_battery(battery_kWh):
    C_battery = 520 * battery_kWh
    return C_battery'''

#Annual Utilization
def U_annual(tb, day_use):
    U_annual = tb * day_use
    return U_annual

# Insurance
def C_insurance(tb, IR_a, C_aircraft, U_annual):
    C_insurance = ((IR_a * C_aircraft)/(U_annual)) * tb
    return C_insurance

# Depreciation
def C_depreciation(C_flyaway, K_depreciation, tb, n, U_annual):
    C_depreciation = (C_flyaway * (1- K_depreciation) * tb)/(n * U_annual)
    return C_depreciation

#COC
def COC(C_crew, C_fuel, C_airport, C_navigation, C_AirMain, C_EngMain):
    COC = C_crew + C_fuel + C_airport + C_navigation + C_AirMain + C_EngMain
    return COC

#Direct Operating Costs
def DOC(COC, C_insurance, C_depreciation):
    DOC = COC + C_insurance + C_depreciation
    return DOC

# Financing
def C_financing(DOC):
    C_financing = 0.07 * DOC
    return C_financing

# Registration taxes
def C_registration(MTOW, DOC):
    C_registration = (0.001 + (10**(-8)) * MTOW) * DOC
    return C_registration

def DOC_total(DOC, C_registration, C_financing):
    DOC_total = DOC + C_registration + C_financing
    return DOC_total

C_crew = C_crew(AF, K, MTOW, tb, CEF)
C_fuel = C_fuel(W_f, P_f, rho_f)
C_airport = C_airport(MTOW, CEF)
C_navigation = C_navigation(CEF, R, tb, MTOW)
C_AirMain = C_AirMain(W_A, RL, CEF, C_airframe, tb)
C_ML_engine = C_ML_engine(T_max, tb, RL)
C_MM_engine = C_MM_engine(T_max, tb, CEF)
C_ML_turbo = C_ML_turbo(SHP_TO, n_engines_turbo, H_em, RL)
C_EngMain = C_EngMain(n_engines, n_engines_turbo, C_ML_engine, C_MM_engine, C_ML_turbo, tb)
U_annual = U_annual(tb, year_use)
C_insurance = C_insurance(tb, IR_a, C_aircraft, U_annual)
C_depreciation = C_depreciation(C_flyaway, K_depreciation, tb, n, U_annual)
COC = COC(C_crew, C_fuel, C_airport, C_navigation, C_AirMain, C_EngMain)
DOC = DOC(COC, C_insurance, C_depreciation)
C_financing = C_financing(DOC)
C_registration = C_registration(MTOW, DOC)
DOC_total = DOC_total(DOC, C_registration, C_financing)

print(f"The Cash Operating Cost is ${COC:,.2f}. The Direct Operating Cost is
${DOC_total:,.2f}.")
```