```julia
#Problem 1

function FlightPathAnglesFromState(aircraft_state)

    euler_angles = EulerAngles(aircraft_state[4:6])
    wind_angles = AirRelativeVelocityVectorToWindAngles(aircraft_state[7:9])
    Vg = TransformFromBodyToInertial(aircraft_state[7:9], euler_angles)
    #=
    We know γ_a = γ when there is no wind.
    Thus, γ = γ_a = θ - α
    =#
    # γ = aircraft_state.θ - wind_angles.α
    γ = atan(Vg[3], sqrt(Vg[1]^2 + Vg[2]^2))
    χ = atan(Vg[2], Vg[1])

    return (Vg,χ,γ)
end


#=
filename = "ttwistor.mat"
aircraft_parameters = AircraftParameters(filename)

trim_definition = TrimDefinitionCT(18.0,0.0,1655,500.0)
state, control, results = GetTrimConditions(trim_definition, aircraft_parameters)
trim_variables = TrimVariablesCT(results.minimizer)
Vg, χ, γ = FlightPathAnglesFromState(state)
=#
#Definitions
x = AircraftState(
    100.0,
    200.0,
    -1655.0,
    -12*pi/180,
    9*pi/180,
    140*pi/180,
    15.0,
    -3.0,
    1.0,
    0.08*pi/180,
    -0.2*pi/180,
    0.0*pi/180
    )

wind_inertial = [0, 1, -2]

#P1
ϕ = x.ϕ
q = x.q
r = x.r
θ_dot = q*cos(ϕ) - r*sin(ϕ)

#P2
euler_angles = EulerAngles(x.ϕ, x.θ, x.ψ)
wind_body_frame = TransformFromInertialToBody(wind_inertial, euler_angles)
Va = x[7:9] - wind_body_frame   #Va vector in body frame
wind_angles = AirRelativeVelocityVectorToWindAngles(Va)
α = wind_angles.α

#P3
m = 10
w_dot = 0.05
θ = x.θ
u = x.u
p = x.p
v = x.p
g = 9.81
fZ = m*(w_dot -q*u + p*v)
fZ_gravity = m*g*cos(θ)*cos(ϕ)
fZ_aero = fZ - fZ_gravity
```