```
# Load the required packages
using ODE
using JLD

# Set precision to quadruple precision (QP)
# QP has 113 bits
set_bigfloat_precision(113);

# Define the right-hand function
function arenstorf(t,x)

    mu = parse(BigFloat,"0.012277471"); ms = 1 - mu;
    r1 = vecnorm(x[1:2]-[-mu,0]);
    r2 = vecnorm(x[1:2]-[ ms,0]);

    dx = Vector{BigFloat}(4);
    dx[1] = x[3];
    dx[2] = x[4];
    dx[3] = x[1] + 2*x[4] - ms*(x[1]+mu)/r1^3 - mu*(x[1]-ms)/r2^3;
    dx[4] = x[2] - 2*x[3] - ms*     x[2]/r1^3 - mu*     x[2]/r2^3;
    return dx;
end

# Set up the initial conditions
t0 = parse(BigFloat,"0.0");
T = parse(BigFloat,"17.065216560157962558891720624 9");
x0=[parse(BigFloat,"0.994"),
    parse(BigFloat,"0.0"),parse(BigFloat,"0.0"),
    parse(BigFloat,"-2.00158510637908252240537862224")];

# Set the tolerance as machine epsilon
Tol = parse(BigFloat,"1e-33");

# Solve and get the solution at T = tEnd
(t,x_ref) = ode78(arenstorf,x0,[t0,T];
reltol=Tol,abstol=Tol,points=:specified);

# Save the solution to a file
save("refSolAren.jld","x_ref",x_ref[2]);
```