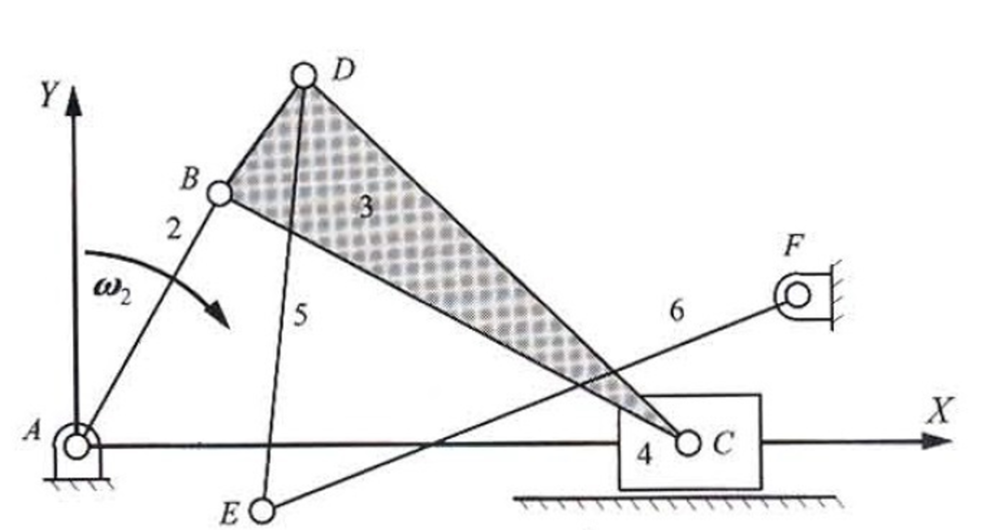
Mechanism 11 analytical solution



In[ ]:= `Clear[th2, th3]`

Here we input the first 3 position equations and their derivatives to get all the data related to theta 3 and the position of the slider.

In[ ]:= `positionequation1 = R2 * Cos[th2] + R3 * Cos[th3] - R4`

Out[ ]= -R4 + R2 Cos[th2] + R3 Cos[th3]

In[ ]:= `velocityequation1 = R2 * Sin[th2] * omega2 + R3 * Sin[th3] * omega3 + v4`

Out[ ]= v4 + omega2 R2 Sin[th2] + omega3 R3 Sin[th3]

In[ ]:= `accelerationequation1 = R2 * Cos[th2] * (omega2)^2 + R2 * Sin[th2] * alpha2 + R3 * Cos[th3] * omega3^2 + R3 * Sin[th3] * alpha3 + a4`

Out[ ]= a4 + omega2^2 R2 Cos[th2] + omega3^2 R3 Cos[th3] + alpha2 R2 Sin[th2] + alpha3 R3 Sin[th3]

In[ ]:= `positionequation2 = R2 * Sin[th2] + R3 * Sin[th3]`

Out[ ]= R2 Sin[th2] + R3 Sin[th3]

In[ ]:= `velocityequation2 = R2 * Cos[th2] * omega2 + R3 * Cos[th3] * omega3`

Out[ ]= omega2 R2 Cos[th2] + omega3 R3 Cos[th3]

In[ ]:= `accelerationequation2 = -R2 * Sin[th2] * omega2^2 + R2 * Cos[th2] * alpha2 - R3 * Sin[th3] * omega3^2 + R3 * Cos[th3] * alpha3`

Out[ ]= alpha2 R2 Cos[th2] + alpha3 R3 Cos[th3] - omega2^2 R2 Sin[th2] - omega3^2 R3 Sin[th3]

Now we plug in the length of each side. R2 is AB,R3 is BC and it is assumed that link AB is rotating with constant speed of 1000 rpm ccw

In[∘]:=
```
R2 = 14
R3 = 41
omega2 = 1000 * 2 * 3.14 / 60
alpha2 = 0
```

Out[∘]= 14

Out[∘]= 41

Out[∘]= 104.667

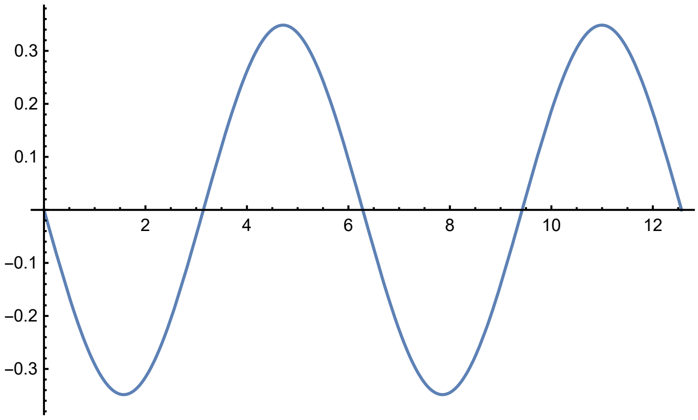
Out[∘]= 0

Here we start plotting the data related to the previous equations.when solving the equations we have a system of non linear equations so it is obvious that we always have 2 solutions including one refused one. the plots shows the accepted answer.

In[∘]:=
```
Plot[th3 /. NSolve[{positionequation1 == 0, positionequation2 == 0}, {th3, R4}][[2]],
 {th2, 0, 4 * Pi}]
```

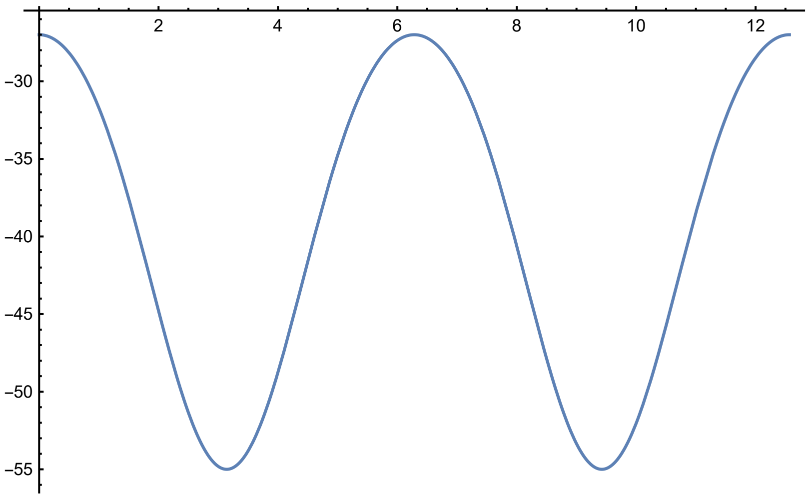
NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[∘]=



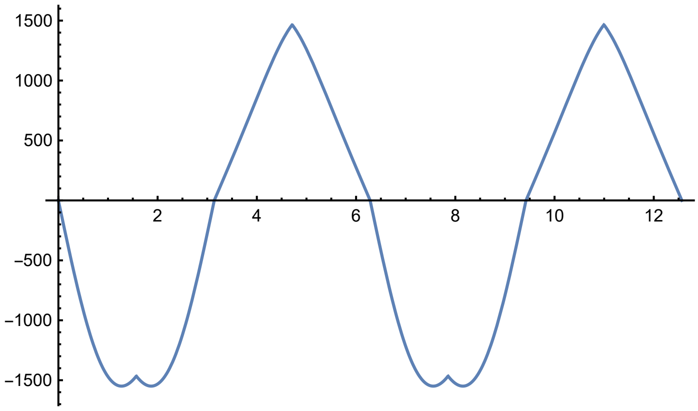
In[ ]:=
```
Plot[R4 /. NSolve[{positionequation1 == 0, positionequation2 == 0}, {th3, R4}][[1]],
 {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]=


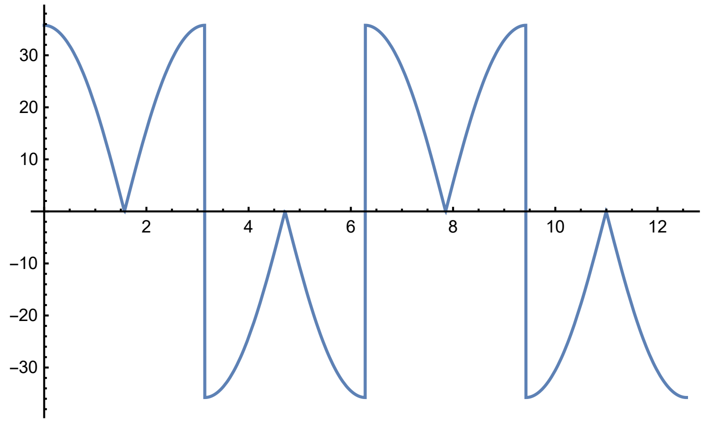
In[ ]:=
```
Plot[v4 /. NSolve[{positionequation1 == 0, positionequation2 == 0, velocityequation1 == 0,
     velocityequation2 == 0}, {th3, R4, v4, omega3}][[1]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]=


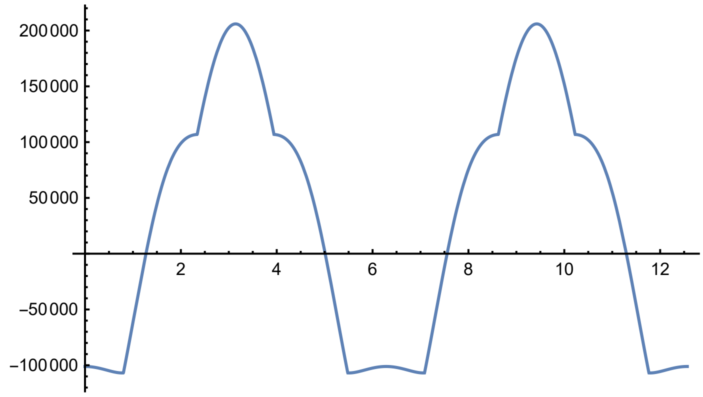
In[ ]:=
```
Plot[
 omega3 /. NSolve[{positionequation1 == 0, positionequation2 == 0, velocityequation1 == 0,
     velocityequation2 == 0}, {th3, R4, v4, omega3}][[2]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]=


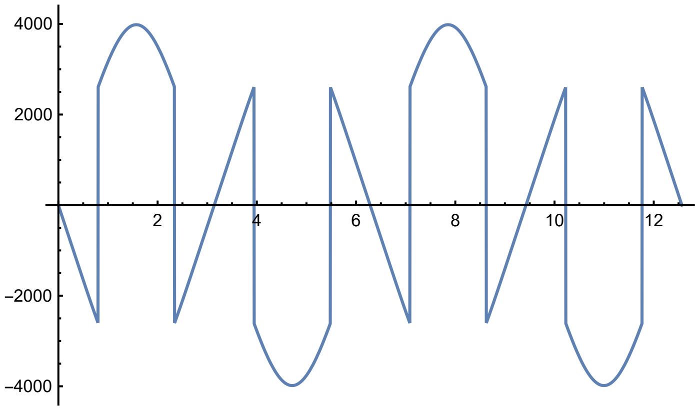
In[ ]:=
```
Plot[a4 /. NSolve[{positionequation1 == 0, positionequation2 == 0, velocityequation1 == 0,
     velocityequation2 == 0, accelerationequation1 == 0, accelerationequation2 == 0},
    {th3, R4, v4, omega3, a4, alpha3}][[2]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]=


In[ ]:=
```
Plot[
 alpha3 /. NSolve[{positionequation1 == 0, positionequation2 == 0, velocityequation1 == 0,
     velocityequation2 == 0, accelerationequation1 == 0, accelerationequation2 == 0},
    {th3, R4, v4, omega3, a4, alpha3}][[2]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]=


Here we plug in the next required equations to get the data related to theta 5,theta 6. here we find that we used the link R31 which is equivalent to the side BD in the triangle. there relation between it's angle and theta 3 is : theta31 = angle(DBC)-theta3 so angle DBC was evaluated using law of cosines and it is found in the equations.

In[ ]:=
```
positionequation3 =
 R2 * Cos[th2] + R31 * Cos[1.327102 - th3] + R5 * Cos[th5] + R6 * Cos[th6] - A
```

Out[ ]= 7.63087 − A + R31 Cos[1.3271 − th3] + R5 Cos[th5] + R6 Cos[th6]

In[ ]:=
```
positionequation4 =
 R2 * Sin[th2] + R31 * Sin[1.327102 - th3] + R5 * Sin[th5] + R6 * Sin[th6] - B
```

Out[ ]= 11.7375 − B + R31 Sin[1.3271 − th3] + R5 Sin[th5] + R6 Sin[th6]

In[ ]:=
```
velocityequation3 = -R2 * Sin[th2] * omega2 +
  R31 * Sin[1.327102 - th3] * omega3 - R5 * Sin[th5] * omega5 - R6 * Sin[th6] * omega6
```

Out[ ]= −1228.53 + omega3 R31 Sin[1.3271 − th3] − omega5 R5 Sin[th5] − omega6 R6 Sin[th6]

In[ ]:=
```
velocityequation4 = R2 * Cos[th2] * omega2 -
  R31 * Cos[1.327102 - th3] * omega3 + R5 * Cos[th5] * omega5 + R6 * Cos[th6] * omega6
```

Out[ ]= 798.697 − omega3 R31 Cos[1.3271 − th3] + omega5 R5 Cos[th5] + omega6 R6 Cos[th6]

In[ ]:=
```
accelerationequation3 =
 -R2 * Cos[th2] * omega2^2 - R2 * Sin[th2] * alpha2 - R31 * Cos[1.327102 - th3] * omega3^2 +
  R31 * Sin[1.327102 - th3] * alpha3 - R5 * Cos[th5] * omega5^2 -
  R5 * Sin[th5] * alpha5 - R6 * Cos[th6] * omega6^2 - R6 * Sin[th6] * alpha6
```

Out[ ]= $-83597. - \text{omega3}^2\ \text{R31}\ \text{Cos}[1.3271 - \text{th3}] - \text{omega5}^2\ \text{R5}\ \text{Cos}[\text{th5}] - \text{omega6}^2\ \text{R6}\ \text{Cos}[\text{th6}] +$
alpha3 R31 Sin[1.3271 − th3] − alpha5 R5 Sin[th5] − alpha6 R6 Sin[th6]

In[ ]:=
```
accelerationequation4 =
 - R2 * Sin[th2] * omega2^2 + R2 * Cos[th2] * alpha2 - R31 * Sin[1.327102 - th3] * omega3^2 -
  R31 * Cos[1.327102 - th3] * alpha3 - R5 * Sin[th5] * omega5^2 +
  R5 * Cos[th5] * alpha5 - R6 * Sin[th6] * omega6^2 + R6 * Cos[th6] * alpha6
```

Out[ ]= $-128\,586. - \text{alpha3 R31 Cos}[1.3271 - \text{th3}] + \text{alpha5 R5 Cos}[\text{th5}] + \text{alpha6 R6 Cos}[\text{th6}] - \text{omega3}^2\,\text{R31 Sin}[1.3271 - \text{th3}] - \text{omega5}^2\,\text{R5 Sin}[\text{th5}] - \text{omega6}^2\,\text{R6 Sin}[\text{th6}]$

now again we introduce the required constants. the vector Ai+Bj is the same as the vector starting from point A to point F. It is a constant one and the values of A and B are measured using linkage program.

In[ ]:=
```
R5 = 63
R6 = 76
R31 = 14
A = 59.6
B = 24.457
```

Out[ ]= 63

Out[ ]= 76

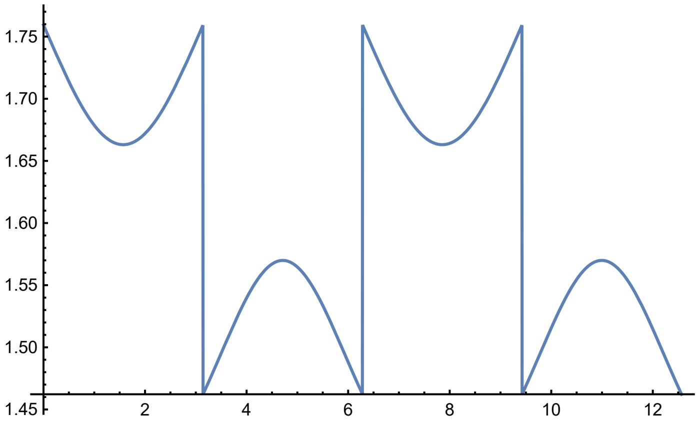
Out[ ]= 14

Out[ ]= 59.6

Out[ ]= 24.457

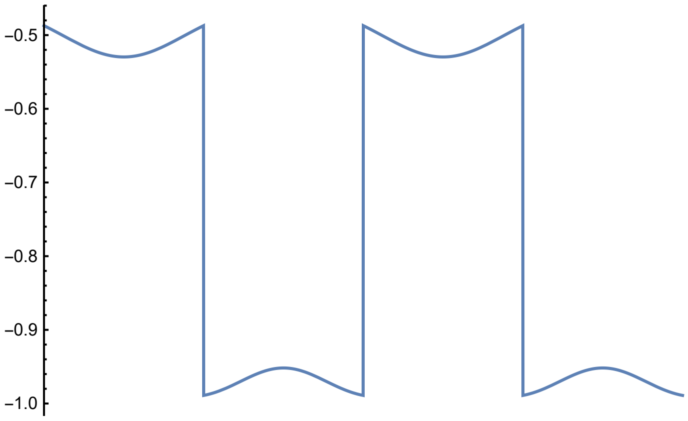
In[ ]:=
```
Plot[
 th5 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0},
    {th3, th5, th6}][[2]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

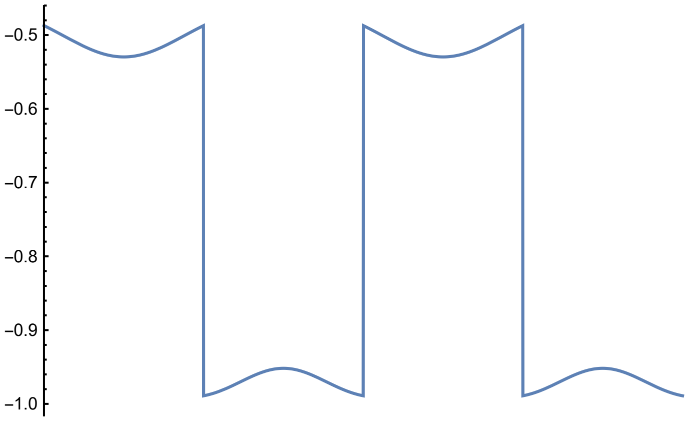
Out[ ]=


In[ ]:=
```
Plot[
 th6 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0},
    {th3, th5, th6}][[2]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

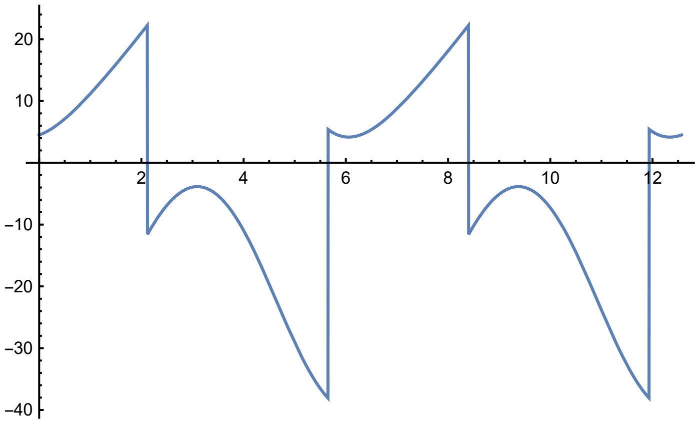
Out[ ]=


In[ ]:=
```
Show[%90, AxesStyle → Black]
```

Out[ ]=


In[ ]:=
```
Plot[
 omega5 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
     velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0},
    {th3, th5, th6, omega5, omega6, omega3}][[2]], {th2, 0, 4 * Pi}]
```

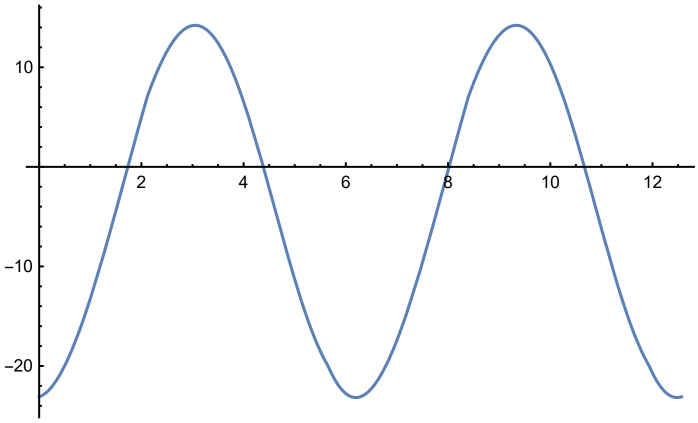
NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]=


In[ ]:=
```
Plot[
 omega6 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
     velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0},
    {th3, th5, th6, omega5, omega6, omega3}][[2]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

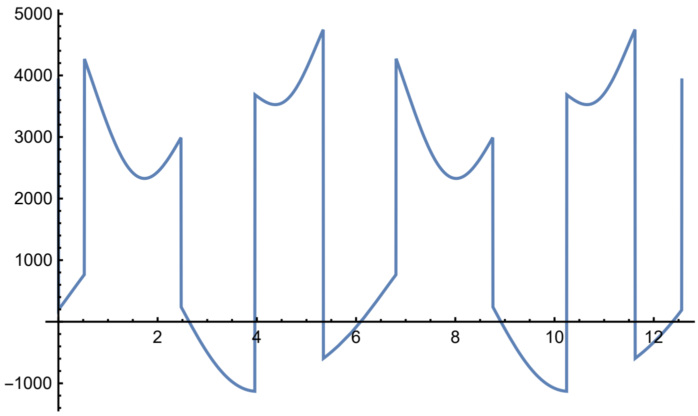
Out[ ]=



In[ ]:=
```
Plot[
 alpha5 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
     velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0,
     accelerationequation2 == 0, accelerationequation3 == 0, accelerationequation4 == 0},
    {th3, th5, th6, omega5, omega6, omega3, alpha3, alpha5, alpha6}][[2]], {th2, 0, 4 * Pi}]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

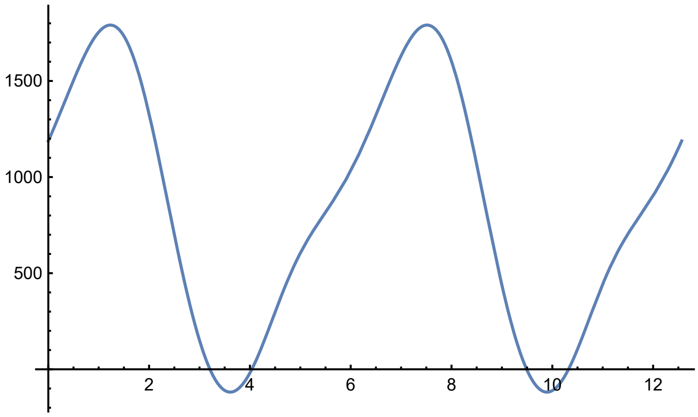
Out[ ]=



In[∘]:=
```
Plot[
 alpha6 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
     velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0,
     accelerationequation2 == 0, accelerationequation3 == 0, accelerationequation4 == 0},
    {th3, th5, th6, omega5, omega6, omega3, alpha3, alpha5, alpha6}][[1]], {th2, 0, 4 * Pi}]
```

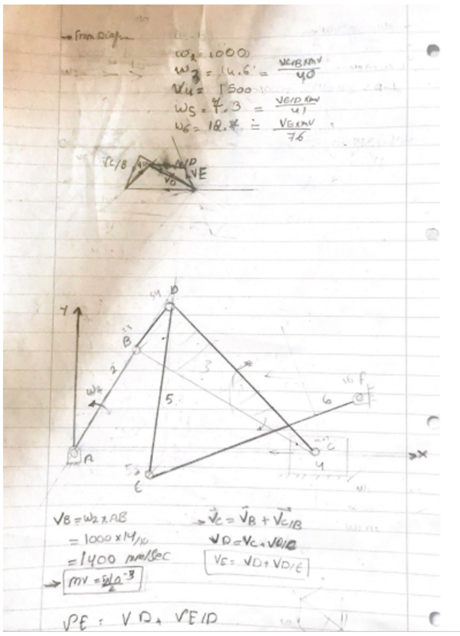
NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

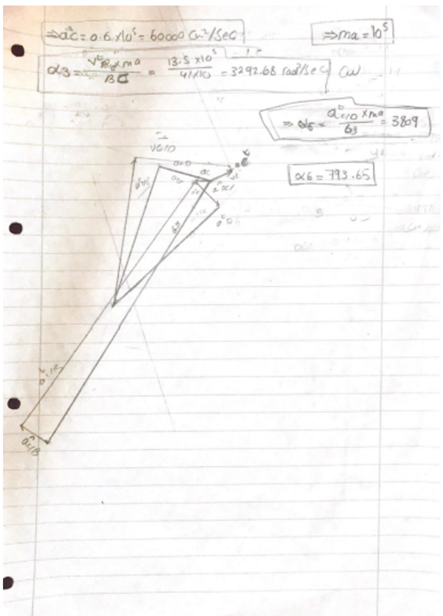
Out[∘]=



Now let’s make sure that the graphs are right

As the mechanism was drawn. theta2=57 so the graphical solution is evaluated at this instant.





```
th2 = 57 * 3.14 / 180
v4 /. NSolve[{positionequation1 == 0, positionequation2 == 0,
    velocityequation1 == 0, velocityequation2 == 0}, {th3, R4, v4, omega3}] [[1]]
```

Out[∘]= 0.994333

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[∘]= -1467.17

In[∘]:=
```
omega3 /. NSolve[{positionequation1 == 0, positionequation2 == 0,
    velocityequation1 == 0, velocityequation2 == 0}, {th3, R4, v4, omega3}] [[2]]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[∘]= 20.3314

In[∘]:=
```
omega5 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
    velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0},
   {th3, th5, th6, omega5, omega6, omega3}] [[2]]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[∘]= 6.06847

In[∘]:=
```
omega6 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
    velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0},
   {th3, th5, th6, omega5, omega6, omega3}] [[2]]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[∘]= -18.4986

In[∘]:=
```
alpha3 /. NSolve[{positionequation1 == 0, positionequation2 == 0,
    velocityequation1 == 0, velocityequation2 == 0, accelerationequation1 == 0,
    accelerationequation2 == 0}, {th3, R4, v4, omega3, a4, alpha3}] [[2]]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[∘]= 3149.74

In[∘]:=
```
alpha5 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
    velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0,
    accelerationequation2 == 0, accelerationequation3 == 0, accelerationequation4 == 0},
   {th3, th5, th6, omega5, omega6, omega3, alpha3, alpha5, alpha6}] [[2]]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[∘]= 3572.4

In[ ]:=
```
alpha6 /. NSolve[{positionequation2 == 0, positionequation3 == 0, positionequation4 == 0,
    velocityequation2 == 0, velocityequation3 == 0, velocityequation4 == 0,
    accelerationequation2 == 0, accelerationequation3 == 0, accelerationequation4 == 0},
  {th3, th5, th6, omega5, omega6, omega3, alpha3, alpha5, alpha6}] [[1]]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]= 1819.

In[ ]:=
```
a4 /. NSolve[{positionequation1 == 0, positionequation2 == 0,
    velocityequation1 == 0, velocityequation2 == 0, accelerationequation1 == 0,
    accelerationequation2 == 0}, {th3, R4, v4, omega3, a4, alpha3}] [[2]]
```

NSolve: Inverse functions are being used by NSolve, so some solutions may not be found; use Reduce for complete solution information.

Out[ ]= -62 865.5