In[1030]:=

```
Clear["Global`*"]
(* Task 2 *)
equation := {a * y''[x] + p * y'[x] + q * y[x] == 0}
functions := {y[x]}
variables := {x, 0, 20}
initial := { y[0] == 1, y'[0] == 0}
(*Symbolic , general solution*)
DSolve[equation , functions , {x}]
(* Parameters*)
a := 1
p := 1
q := 1
delta = p * p - 4 * a * q
biggerPole := (-p + Sqrt[delta]) / (2 * a)
Re[biggerPole ]
(*Symbolic solution*)
symbolic := DSolve[Join[equation , initial], functions , {x}]
symbolic
Plot[Evaluate[functions /. symbolic],
 variables , PlotLabel → Style["SYMBOLIC ", FontSize → 18]]
(*Numerical solution . Also Task 3*)
numerical := NDSolve[Join[equation , initial], functions , variables]
numerical
(*numerical :=NDSolve[Join[equation , initial], functions , variables]*)
Plot[Evaluate[functions /. numerical], variables ,
 PlotLabel → Style["NUMERICAL ", FontSize → 18]]
(*About delta: it decides about oscillations
   and stability . Complex delta means oscillations ,
real delta means there's no oscillations . Positive POLES means
 UNSTABLE . If there're no positive POLES, it means it's stable*)
```

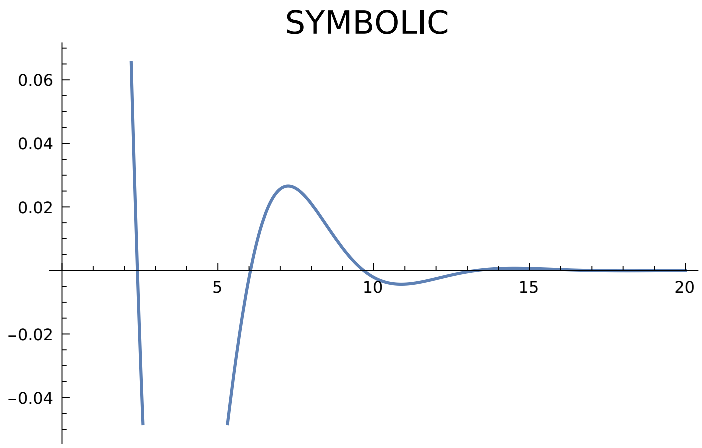
Out[1035]= $\left\{\left\{y[x] \to e^{\frac{1}{2}\left(-\frac{p}{a}-\frac{\sqrt{p^2-4 a q}}{a}\right) x} \mathbb{c}_1 + e^{\frac{1}{2}\left(-\frac{p}{a}+\frac{\sqrt{p^2-4 a q}}{a}\right) x} \mathbb{c}_2\right\}\right\}$

Out[1039]= -3

Out[1041]= $-\frac{1}{2}$

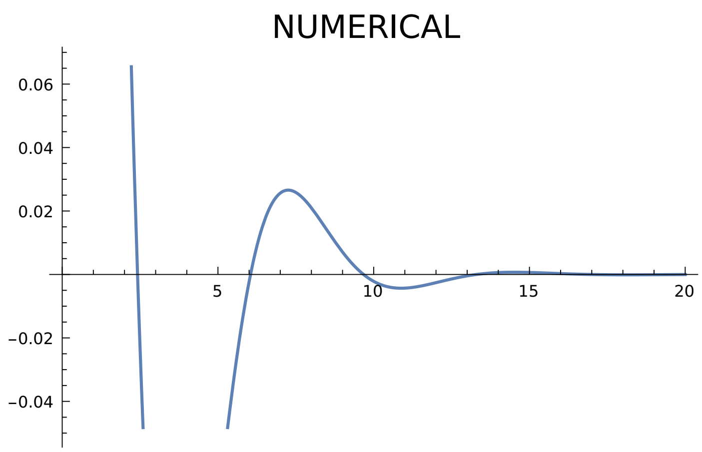
Out[1043]= $\left\{\left\{y[x] \to \frac{1}{3} e^{-x/2} \left(3 \operatorname{Cos}\left[\frac{\sqrt{3}\, x}{2}\right] + \sqrt{3} \operatorname{Sin}\left[\frac{\sqrt{3}\, x}{2}\right]\right)\right\}\right\}$

Out[1044]=



Out[1046]= {{y[x] → InterpolatingFunction[Domain : {{0., 20.}} Output : scalar][x]}}

Out[1047]=

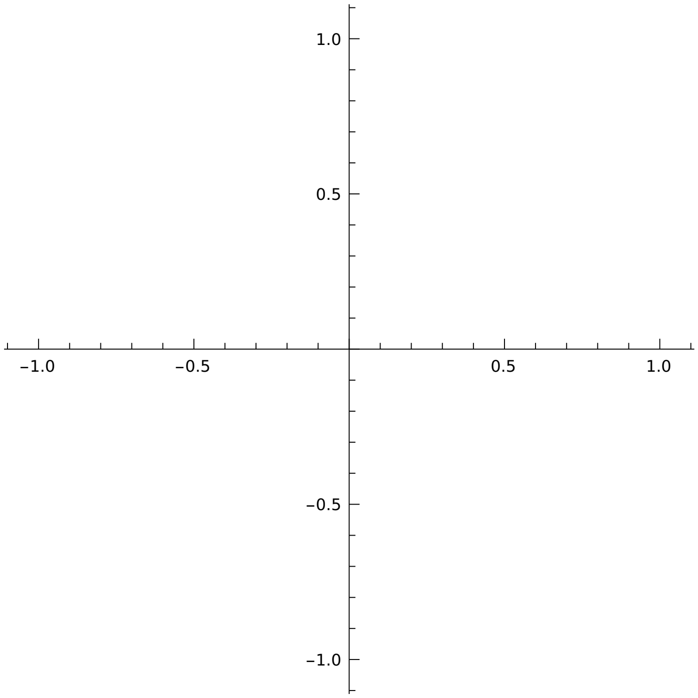


```
Clear["Global`*"]
(*Task 4*)
equation := {x1'[t] == x2[t],
x2'[t] == -x1[t] - k * x2[t]}
initial := {x1[0] == 10,
x2[0] == 10}
functions := {x1[t], x2[t]}
variables := {t, 0, 30}
equations := Join[equation, initial]
(* Meat goes here*)(*just kidding,
this isn't finished, im just trying with smol scale*)
result := NDSolve[equations /. {k → 1}, functions, variables]
{xs, ts} = NDSolveValue[equations /. {k → 1}, functions, variables]
ParametricPlot[Evaluate[{xs[t], ys[t]}], variables, PlotRange → All]
(*wrong
results := Table[NDSolve[equations/.{k→ii},functions, variables],{ii,1,5,1}]
For[i:=1, i ≤Length[results], i++,
tmp := results⟦i⟧;
ParametricPlot[functions/.tmp,variables] // Print; (*DOESN'T WORK*)
Plot[Evaluate[functions/.tmp], variables,
     PlotLabel →Style[StringForm["Picture number = ``", i]] // Print;
]*)
```

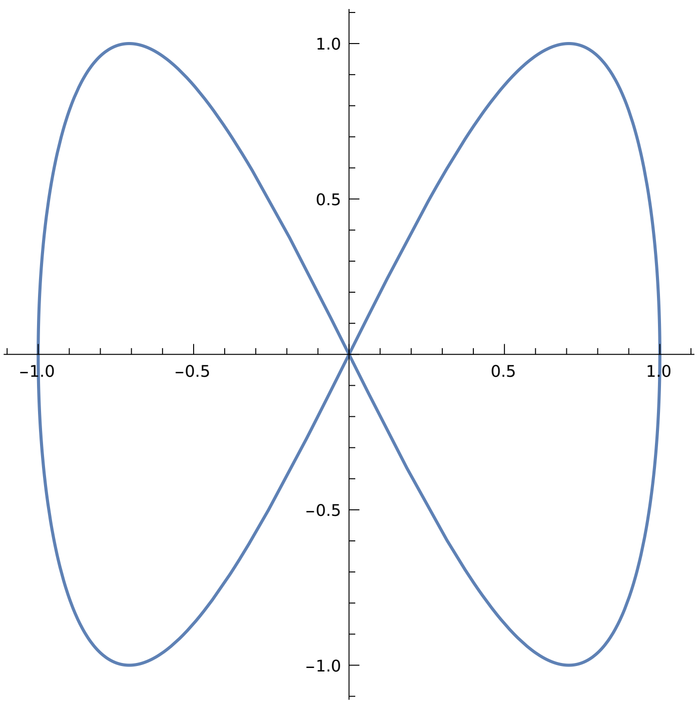
Out[8]=

```
{InterpolatingFunction[ Domain: {{0., 30.}} Output: scalar ][t],
 InterpolatingFunction[ Domain: {{0., 30.}} Output: scalar ][t]}
```

Out[9]=



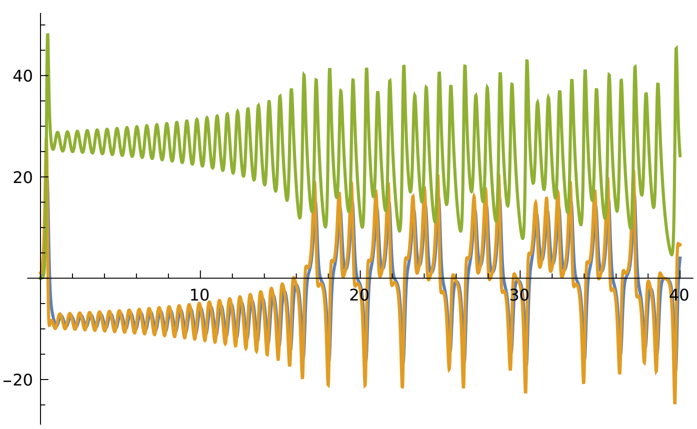
Out[182]=



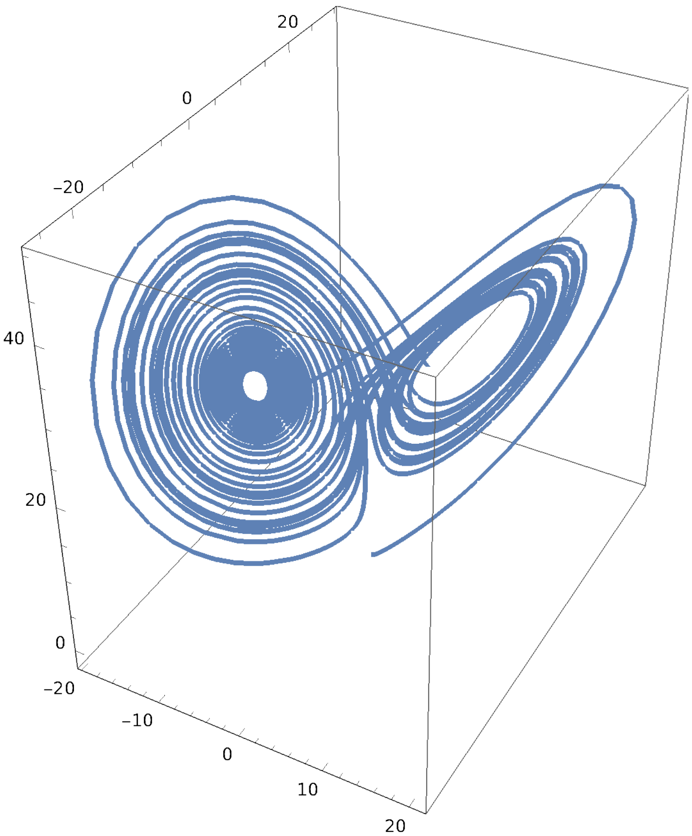
In[173]:=

```
Clear["Global`*"]
(*Task 5*)
equation := {x'[t] == a*(y[t] - x[t]),
y'[t] == x[t] *(c - z[t]) - y[t],
z'[t] == x[t]* y[t] - b * z[t]
}
functions := {x[t], y[t], z[t]}
variables := {t}
(*parameters*)
a := 10 (*alpha*)
b := 8/3 (*beta*)
c := 28 (*ro*)
initial := {x[0] == 0, y[0] == 1, z[0] == 0}
(*workspace*)
equations := Join[equation, initial]
tmin := 0
tmax := 40
(*Meat goes here*)
result := NDSolve[equations, functions, {t, tmin, tmax}]
(*{{X,Y,Z}}:=Evaluate[functions/.result]*)
Plot[Evaluate[functions /. result], {t, tmin, tmax}] (*Time plot - i think*)
(*State space plot ??*)
(*XZ plane - ??*)
ParametricPlot3D [Evaluate[functions /. result], {t, tmin, tmax}]
```

Out[185]=



Out[186]=



In[548]:= `1 * (2 * Sqrt[m * k])`

Out[548]= 2

In[1]:=
```
Clear["Global`*"]
(*Task 6 + list 4 task 2*)
(*parameters*)
k := 1
m := 1
tmin := 0
tmax := 10
(*initial conditions*)
initial = {
x[0] == 1, (*initial position*)
x'[0] == 0.5 (*initial velocity*)
}
(*workspace*)
external := -c * x'[t]
F := -k * x[t] + external
equation := {m * x''[t] == F}
functions := {x[t], x'[t]}
variables := {t, tmin, tmax}
equations := Join[equation, initial]
dzeta = c / (2 * Sqrt[m * k])
(*meat*)
result := NDSolve[equations, functions, variables]
(*Plot[Evaluate[{x[t]}/.result], variables,PlotRange->All]
ParametricPlot[Evaluate[{x[t],x'[t]}/.result],{t,tmin,tmax},PlotRange->All]*)
Manipulate[Plot[Evaluate[{x[t]} /. result /. {c -> cc}], variables, PlotRange -> All,
  PlotLabel -> StringTemplate["dzeta = `1`"][cc / (2 * Sqrt[m * k])]], {cc, 0, 10}]
```

Out[6]= {x[0] == 1, x′[0] == 0.5}

Out[13]= $\frac{c}{2}$

Out[15]=

cc

Plot[{x[t]} /. result, variables, PlotRange → All,
PlotLabel → StringTemplate[dzeta = `1`][$\frac{\text{FE`cc\$\$4}}{2\sqrt{m\,k}}$]]

```
Clear["Global`*"]
(*Task 4 retry*)
(*x1 = x, x2 = y*)
equations := {x'[t] == y[t], y'[t] == -x[t] - k * y[t]}
functions := {x[t], y[t]}
variables := {t, 0, 10}
```

NDSolve : The number of constraints (0) (initial conditions) is not equal to the total differential order of the system plus the number of discrete variables (2).

ReplaceAll : {NDSolve [{x′[t] == y[t], y′[t] == −x[t] − y[t]}, {x[t], y[t]}, {t, 0, 10 }]} is neither a list of replacement rules nor a valid dispatch table , and so cannot be used for replacing .

NDSolve : 0.000204286 cannot be used as a variable .

ReplaceAll : {NDSolve [{x′[0.000204286] == y[0.000204286], y′[0.000204286] == −x[0.000204286] − y[0.000204286]}, {x[0.000204286], y[0.000204286]}, {0.000204286 , 0, 10 }]} is neither a list of replacement rules nor a valid dispatch table , and so cannot be used for replacing .

NDSolve : 0.000204286 cannot be used as a variable .

ReplaceAll : {NDSolve [{x′[0.000204286] == y[0.000204286], y′[0.000204286] == −1. x[0.000204286] − 1. y[0.000204286]}, {x[0.000204286], y[0.000204286]}, {0.000204286 , 0., 10. }]} is neither a list of replacement rules nor a valid dispatch table , and so cannot be used for replacing .

General : Further output of ReplaceAll ::reps will be suppressed during this calculation .

NDSolve : 0.204286 cannot be used as a variable .

General : Further output of NDSolve ::dsvar will be suppressed during this calculation .

Out[7]=