P3:

Primero se definen los parámetros

```
Fin = 1000 (*kg/h*);
Fout = Fin;
Tin = 20 + 273 (*K*);
T0 = 15 + 273(*K*);

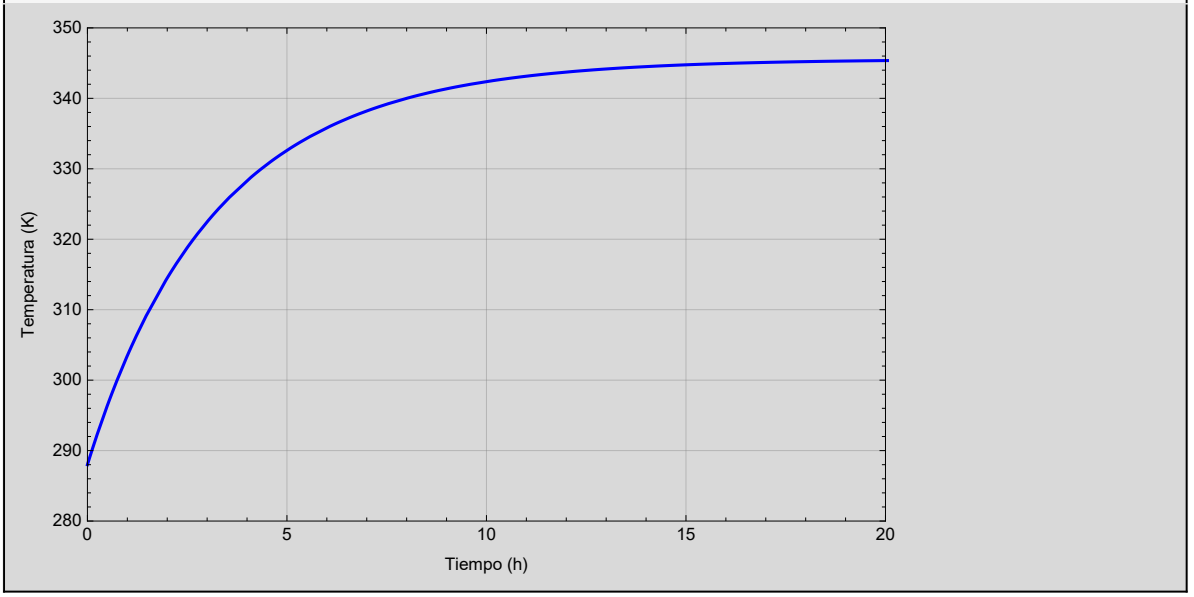

a = -325.1370;
b = 41.3787 * 10^-1;
c = -14.03070 * 10^-3;
d = 1.70354 * 10^-5;
Mr = 46(*Kg/kmol*);
m = 10000 (*kg*);

Tinf = 373.15;
hA = 5000;
```

Se plantea el balance:

$m/\mathrm{Mr} * \mathrm{Cp}\,\frac{dT}{dt} = F1/mr \int_{T}^{T1} \mathrm{Cp}\,dT + Q$

```
Clear[sol]
Q[T_] = hA * (Tinf - T);
(*Resolver con NDSolve o NDSolveValue*)
sol = NDSolveValue[{m * (a + b * T[t] + c * T[t]^2 + d * T[t]^3) * T'[t] / Mr ==
      Q[T[t]] + Fin * (Integrate[(a + b * x + c * x^2 + d * x^3) / Mr, {x, T[t], Tin}]),
    T[0] == T0}, T, {t, 0, 100}];
(*Plot Bonito*)
Plot[sol[t], {t, 0, 100}, Frame → True, PlotStyle → Blue,
 FrameLabel → {"Tiempo (h)", "Temperatura (K)"}, FrameStyle → Black,
 GridLines → Automatic, PlotRange → {{0, 20}, {280, 350}}]
```



Ahora para calcular el valor estacionario, dT/dt se va a 0 (en estacionario no hay variación con el tiempo) y luego se

despeja T.

```
Temp = NSolve[
  {Q[T] == -Fin * (Integrate[(a + b * x + c * x^2 + d * x^3) / Mr, {x, T, Tin}]), T > 0},
  T, Reals]
Tsteady = Evaluate[T /. Temp[[1]]];
```

```
{{T → 345.565}}
```

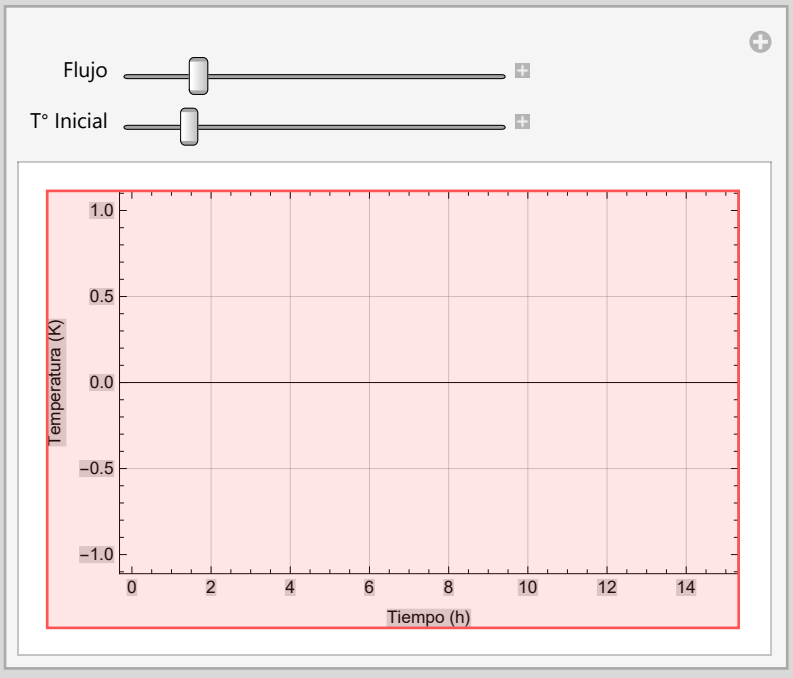
Para el tiempo

```
FindRoot[sol[t] == 340, {t, 5}]
```

```
{t → 7.99952}
```

Pueden jugar con los parámetros aca:

```
Manipulate[sol = NDSolveValue[{m * (a + b * T[t] + c * T[t]^2 + d * T[t]^3) * T'[t] / Mr ==
      Q[T[t]] + F * (Integrate[(a + b * x + c * x^2 + d * x^3) / Mr, {x, T[t], Tin}]),
    T[0] == Tini}, T, {t, 0, 60}];
 Plot[sol[t], {t, 0, 15}, Frame → True, PlotStyle → Blue, FrameLabel →
   {"Tiempo (h)", "Temperatura (K)"}, FrameStyle → Black, GridLines → Automatic],
 {{F, Fin, "Flujo"}, Fin / 2, 2 * Fin}, {{Tini, T0, "T° Inicial"}, T0 - 10, T0 + 10}]
```



NDSolveValue: Encountered non-numerical value for a derivative at t == 0.`.

NDSolveValue: 0.0003064285714285714` cannot be used as a variable.

NDSolveValue: 0.0003064285714285714` cannot be used as a variable.

NDSolveValue: 0.30642887755102044` cannot be used as a variable.

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