In [2]:
```
from sympy import *
```

Example 1: Newton's Law of Cooling (160 degree turkey removed from oven in a 70 degree room, temperature is 155 degrees after 5 minutes. When will the temperature be 140 degrees?)

In [3]:
```
# FIRST STEP: Solve the IVP from Newton's Law of Cooling: dT/dt = k*(T-70), T(0)=160
k,t=symbols('k t')
T=Function('T')
deq=diff(T(t),t)-k*(T(t)-70)
Tsoln=dsolve(deq,T(t),ics={T(0):160})
print('Solution of the ODE is',Tsoln)
```

```
Solution of the ODE is Eq(T(t), 90*exp(k*t) + 70)
```

In [10]:
```
# SECOND STEP: Use the 5 minute temperature to find k
keqn=Tsoln.rhs.subs(t,5)-155  # NOTE: 'rhs' refers to the right hand side of the equation
ksoln=solve(keqn,k)
print(ksoln) # last (5th) solution is real: remember Python starts counting at ZERO!
print('Using the 5 minute temperature reading,',Tsoln.subs(k,ksoln[4]))
```

```
[log(17**(1/5)*2**(4/5)*3**(3/5)/6) - 4*I*pi/5, log(17**(1/5)*2**(4/5)*3**(3/5)/6) - 2*I*pi/5, log(17**(1/5)*2**(4/5)*3**(3/5)/6) + 2*I*pi/5, log(17**(1/5)*2**(4/5)*3**(3/5)/6) + 4*I*pi/5, log(7344**(1/5)/6)]
Using the 5 minute temperature reading, Eq(T(t), 90*exp(t*log(7344**(1/5)/6)) + 70)
```

In [12]:
```
# Finally, use the new equation to solve for t when T = 140
Teqn=Tsoln.rhs.subs(k,ksoln[4])
tcooled=solve(Teqn-140,t)
print(tcooled) #only one solution this time...the 0th one
print('The turkey is cool enough to eat after',tcooled[0],'or',tcooled[0].evalf(),'minutes.')
```

```
[log((7/9)**(1/log(7344**(1/5)/6)))]
The turkey is cool enough to eat after log((7/9)**(1/log(7344**(1/5)/6))) or 21.9840274945891 minutes.
```

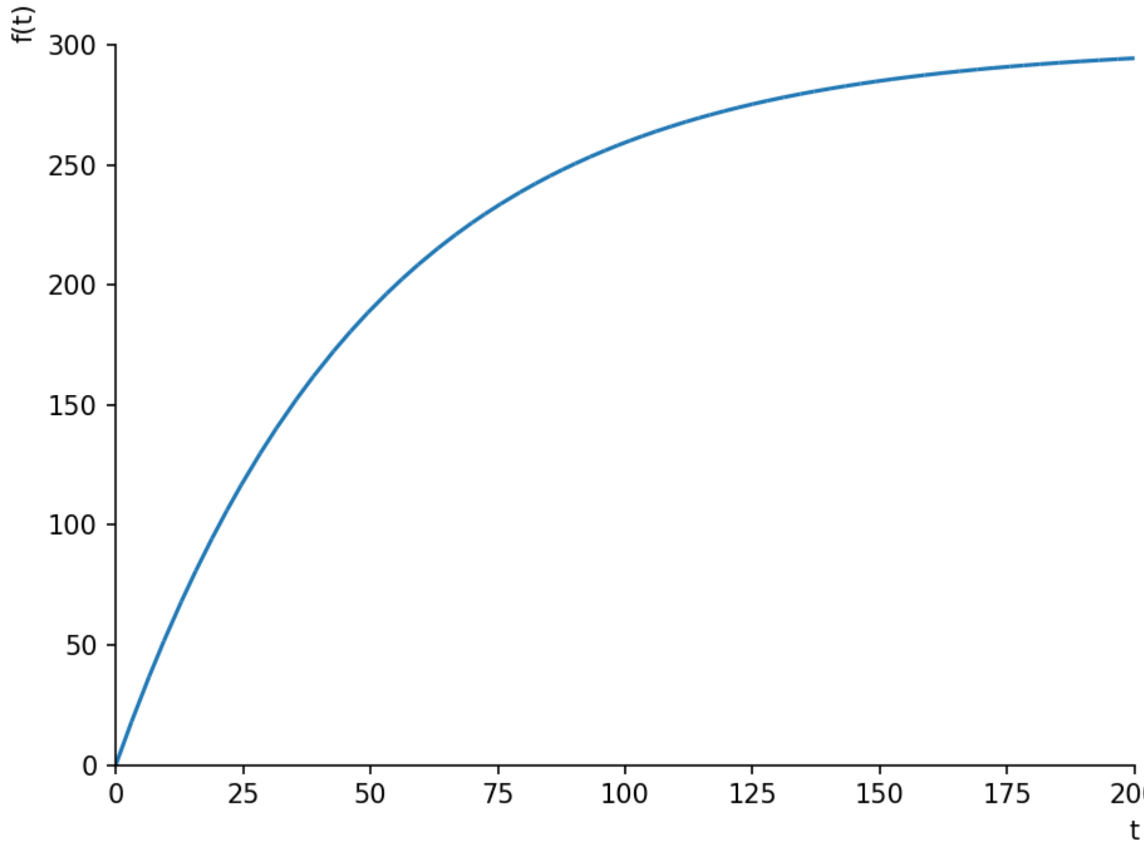
Example 2: IVP (based on rate in - rate out) is

y' = 2a - 2y/100, y(0)=0

In [13]:
```
matplotlib notebook
```

In [14]:
```
t=symbols('t')
a=symbols('a',positive=True)
y=Function('y')
deq=diff(y(t),t)-2*a+2*y(t)/100
ysoln=dsolve(deq,y(t),ics={y(0):0})
print('The solution is',ysoln.expand())
# Limit as t --> oo
yoft=ysoln.rhs #Reminder that rhs gives the right hand side of the equation
ylim=limit(yoft,t,oo)
print('As t-->oo, y approaches',ylim)
# Symbolic plot
yplot=yoft.subs(a,3)
plot(yplot,(t,0,200))
```

```
The solution is Eq(y(t), 100*a - 100*a*exp(-t/50))
As t-->oo, y approaches 100*a
```



Out[14]:
```
<sympy.plotting.plot.Plot at 0x99432d0>
```

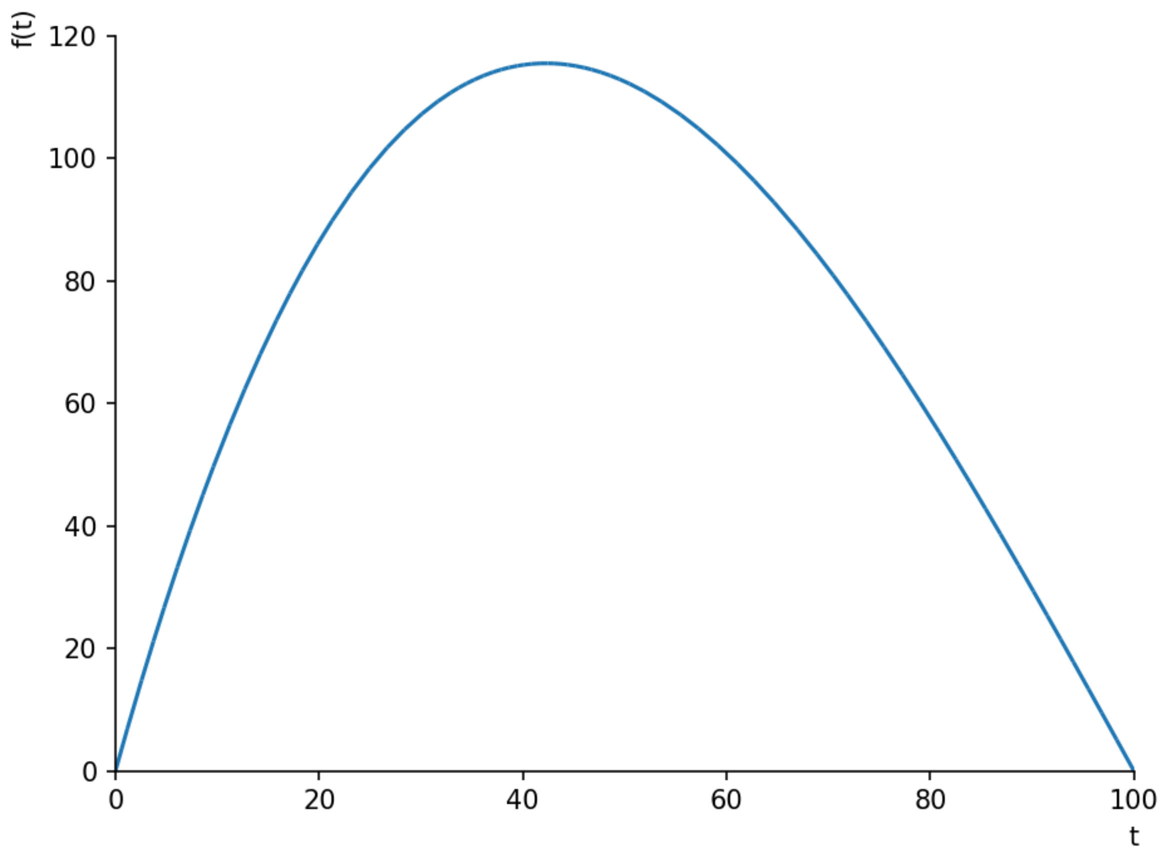
Example 3: Now the ODE is given by

y' = 6 - 3 * y/(100-t) , y(0)=0 (NOTE that solvent is decreasing by 1 L/min!).

In [6]:
```
matplotlib notebook
```

In [15]:
```
t=symbols('t')
y=Function('y')
deq=diff(y(t),t)-6+3*y(t)/(100-t)
ysoln=dsolve(deq,y(t),ics={y(0):0})
print('Solution is',ysoln.expand())
yoft=ysoln.rhs
# Symbolic plot
yplot=yoft.subs(a,3)
plot(yplot,(t,0,100))
# NOTICE that this model breaks down after 100 minutes (100 - t = 0)
```

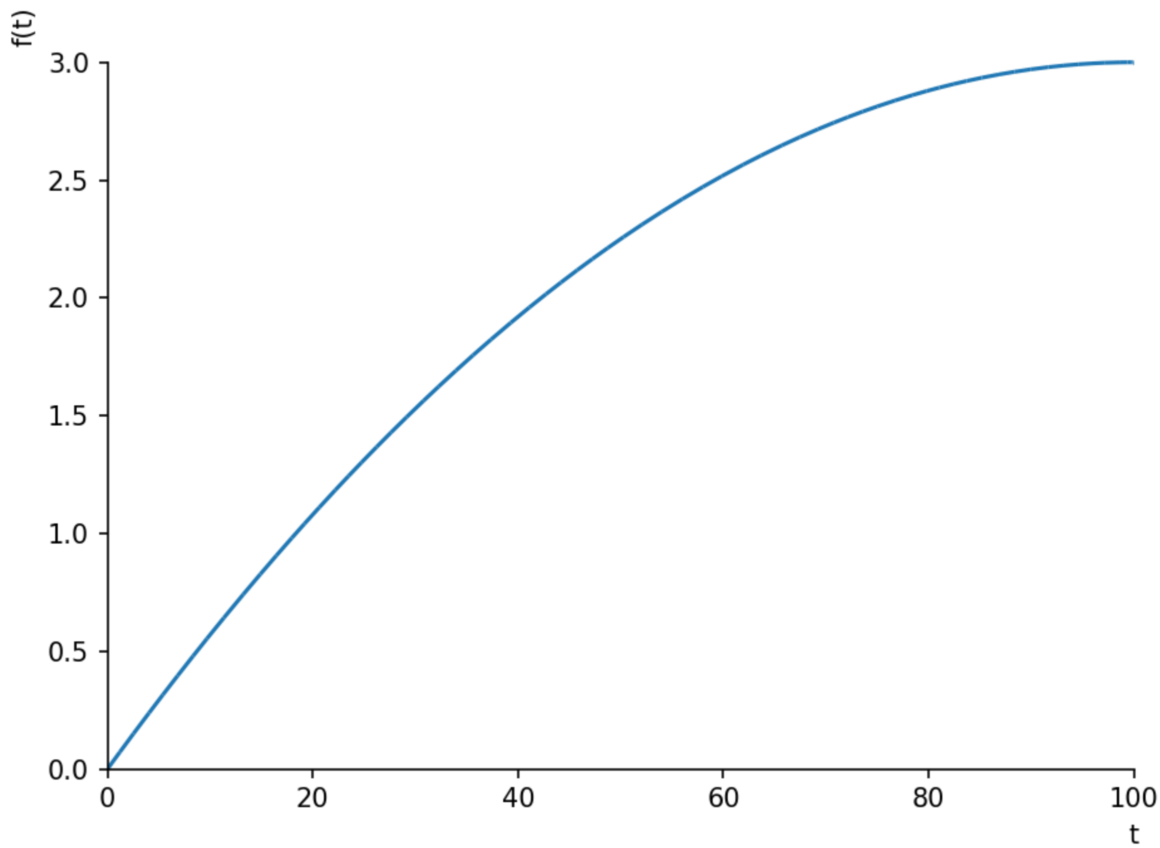
```
Solution is Eq(y(t), 3*t**3/10000 - 9*t**2/100 + 6*t)
```



Out[15]: ``<sympy.plotting.plot.Plot at 0x53df4f0>``

In [16]:
```
matplotlib notebook
```

In [17]:
```
# Plot the concentration over time: NOTE despite the solution approaching
 0, the concentration still approaches 3!
cplot=yplot/(100-t)
plot(cplot,(t,0,100))
```



Out[17]: <sympy.plotting.plot.Plot at 0x7d2f0d0>

In [ ]: