```
> #Title: FormalLimitAtInfinity
#Author: Gord Clement, July 2010
#Description: For a given function and epsilon value,
     this procedure calculates the limit of the function at

    #                  either positive or negative infinity, the minimum N value for the finite limit and
    displays an animation demonstrating
 #                  all possible N values
#Usage:
#Call: FormalLimitAtInfinity( function, variable, point, epsilon (optional) )
#function: function to use in animation
#variable: the variable the limit is taken with respect to

    #point: one of 'infinity' or '-infinity', indicating whether the limit is taken at positive or
    negative infinity
#epsilon: given epsilon value used to calculate delta, default=0.1

FormalLimitAtInfinity := proc(expr, var, pt, epsilon := 0.1)
#local variable declarations
local n, output, value, limval, yupper, ylower, xupper, xlower, xrange, i;
 #calculuate limit
limval := limit(expr, var = pt);
 #initialize N
 n := 0;
 if (not (type(evalf(limval), numeric))) then
          #Error Message
          output := "Error: Limit does not exist or is not finite at given point";
 elif (epsilon ≤ 0) then
          #Error Message
          output := "Error: epsilon value must be positve";
  #Limit taken at positive infinity
 elif (pt = ∞) then
           #Set y-bounds
           yupper := limval + epsilon;
          ylower := limval − epsilon;
          #solve all possible x-ranges satisfying the y-bounds
          xrange := solve(abs(expr − limval) < epsilon) ;
          #if just one x-range
          if (nops([xrange]) = 1) then
                #extract N value which will be the left end-point of the range
                n := op(1 , xrange);
                n := op(1, n);
           #if many x-ranges
          else
                #step through all possible ranges
                for i from 1 to nops([xrange]) do
                       #extract end-points
                      xupper := op(2, xrange[i]);
                      xupper := op(1, xupper);
                      #if the right end point is infinity, the left must be N
                      if (xupper = ∞) then
                             #extract N value
```

```
                        n := op(1 , xrange[i]);
                        n := op(1, n);
                    end if;
                end do;

        end if;
        #impose non-negativity condition on N
        if (n < 0) then
            n := 0;
        end if;
        #display animation
        plots[animate](plot, [[[var, expr, var=-n..5·n + 5], [t, yupper, t=-n..5·n + 5], [t,
    ylower, t=-n..5·n + 5], [n, t, t=limval − 2·epsilon..limval + 2·epsilon], [N, t, t=ylower
    ..yupper]], var=- n..5·n + 5, y=limval − 2·epsilon..limval + 2·epsilon, thickness=[2, 2,
    2, 2, 1], color=[black, red, red, blue, red], linestyle=[solid, solid, solid, dash, dash], labels
    =["", ""]], N=n..5·n + 5);
 #Limit is taken at negative infinity
elif (pt=-∞) then
        #set y-bounds
        yupper := limval + epsilon;
        ylower := limval − epsilon;
        #find all possible x-ranges
        xrange := solve(abs(expr − limval) < epsilon) ;
         #find N as before, this time if the left end point is -infinity, the right must be N
        if (nops([xrange]) = 1) then
            n := op(2, xrange);
            n := op(1, n);
        else
            for i from 1 to nops([xrange]) do
                xlower := op(1, xrange[i]);
                 if (xlower=-∞) then
                    n := op(2 , xrange[i]);
                    n := op(1, n);
                end if
            end do;
          end if;
          #impose non-positivity condition on N
          if (n > 0) then
            n := 0;
          end if;
          #display animation
          plots[animate](plot, [[[var, expr, var=5·n − 5.. -n], [t, yupper, t=5· n − 5..
    -n], [t, ylower, t=5· n − 5..- n], [n, t, t=limval − 2·epsilon..limval + 2·epsilon], [N, t, t
    =ylower..yupper]], var=5· n − 5..- n, y=limval − 2·epsilon..limval + 2·epsilon,
    thickness=[2, 2, 2, 2, 1], color=[black, red, red, blue, red], linestyle=[solid, solid, solid,
    dash, dash], labels=["", ""]], N=n..5·n -5);
 else
        #Error message
        output := "Error: Limit must be at ±∞";
end if;

 end proc:
```

> $FormalLimitAtInfinity\left(\frac{1}{x+10}, x, -\infty\right)$

>

>