```
> restart;
> In := (lambda, T) -> 2*Pi*c^2*h/lambda^5 * 1/(exp(h*c/(lambda*k*
  T))-1);
> In2:= (lambda,T) -> eval(In(lambda,T), {c=1,h=1,k=1});
> plot([In2(lambda,1), In2(lambda,1.2)], lambda=0..1.7);
> limit(In(lambda,T), lambda=0);
> limit(exp(-a*b*x), x=infinity) assuming a>0;
> limit(In(lambda,T), lambda=0,right) assuming h>0, c>0, k>0,T>0;
> assume(h>0, c>0, k>0, T>0);
> about(h);
> limit(In(lambda,T), lambda=0,right);
> In(lambda,T);
> unassign('h', 'c', 'k', 'T');
> about(h);
> dIn := diff(In(lambda,T), lambda);
> lambda[max] := solve(dIn = 0, lambda);
> with(ScientificConstants);
> GetConstant(e); GetConstant(c);
> evalf(eval(c, c=Constant(c)));
> lambda[max] := evalf(eval(lambda[max], {c=Constant(c), h=Constant
  (h), k=Constant(k)}));
> lambda[max] := 'lambda[max]';
> Rad := int(In(lambda,T), lambda=0..infinity);
> Rad;
> evalf(eval(Rad, {c=Constant(c), h=Constant(h), k=Constant(k)}));
> evalf(eval(Constant(c)));
> int(In(lambda,T), lambda=0..infinity) assuming h>0, c>0, k>0,
  T>0;
> assume(h>0, c>0, k>0, T>0);
> Rad := int(In(lambda,T), lambda=0..infinity);
> evalf(eval(Rad, {c=Constant(c), h=Constant(h), k=Constant(k)}));
> lprint(Rad);
> Rad2 := evalf(eval(lprint(Rad), {c=Constant(c), h=Constant(h), k=
  Constant(k)}));
> about(c);
> unassign('h', 'c', 'k', 'T');
> Rad;
> evalf(eval(2*Pi^5*k^4*T^4/(15*c^2*h^3), {c=Constant(c), h=
  Constant(h), k=Constant(k)}));
```