```
> # Prof. Dr. Serkan Dağ
  # ME 310 Numerical Methods
  # File 9.2
  # Lagrange Interpolating Polynomials
> restart :
  with(CurveFitting) :
  Digits := 16 :
> # Linear Interpolation
> # Generate the 1-st order polynomial
> f1L := PolynomialInterpolation([0.10377, 0.11144], [6.4147, 6.5453], v, form=Lagrange);
                        f1L := 17.0273794002607 v + 4.64776883963495                    (1)
> # Same result needs to be found by Newton form
> f1N := PolynomialInterpolation([0.10377, 0.11144], [6.4147, 6.5453], v, form=Newton);
                      f1N := 17.02737940026076 v + 4.647768839634941                    (2)
> subs(v=0.108, f1L);
                                  6.486725814863106                                     (3)
> # Quadratic Interpolation
> # Generate the 2-nd order polynomial
> f2L := expand( PolynomialInterpolation([0.10377, 0.11144, 0.1254], [6.4147, 6.5453, 6.7664], v, form
      =Lagrange) );
             f2L := -54.98245574384 v^2 + 28.86015370090 v + 4.0119446396729            (4)
> # Newton form
> f2N := expand( PolynomialInterpolation([0.10377, 0.11144, 0.1254], [6.4147, 6.5453, 6.7664], v, form
      =Newton) );
         f2N := -54.98245574383980 v^2 + 28.86015370089253 v + 4.011944639672877        (5)
> subs(v=0.108, f2L);
                                  6.487525875573950                                     (6)
> # Inverse interpolation to find v for s = 6.6 kj/(kgK)
> solve(f2L = 6.6, v);
                         0.1147707902952466, 0.4101267121982350                         (7)
>
```