```
> restart;
  my_cubic_spline := proc(xs, ys, curX)
  description "Cubic Spline Implementation";

  # Polynomials

  local P0 := x → a0·(x − xs[1])^3 + b0·(x − xs[1])^2 + c0·(x − xs[1]) + d0;
  local P1 := x → a1·(x − xs[2])^3 + b1·(x − xs[2])^2 + c1·(x − xs[2]) + d1;
  local P2 := x → a2·(x − xs[3])^3 + b2·(x − xs[3])^2 + c2·(x − xs[3]) + d2;
  local P3 := x → a3·(x − xs[4])^3 + b3·(x − xs[4])^2 + c3·(x − xs[4]) + d3;
  local P4 := x → a4·(x − xs[5])^3 + b4·(x − xs[5])^2 + c4·(x − xs[5]) + d4;
  local P5 := x → a5·(x − xs[6])^3 + b5·(x − xs[6])^2 + c5·(x − xs[6]) + d5;
  local P6 := x → a6·(x − xs[7])^3 + b6·(x − xs[7])^2 + c6·(x − xs[7]) + d6;
  local P7 := x → a7·(x − xs[8])^3 + b7·(x − xs[8])^2 + c7·(x − xs[8]) + d7;
  local P8 := x → a8·(x − xs[9])^3 + b8·(x − xs[9])^2 + c8·(x − xs[9]) + d8;
  local P9 := x → a9·(x − xs[10])^3 + b9·(x − xs[10])^2 + c9·(x − xs[10]) + d9;

  # First order diffs

  local dP0 := diff(P0(x), x);
  local dP1 := diff(P1(x), x);
  local dP2 := diff(P2(x), x);
  local dP3 := diff(P3(x), x);
  local dP4 := diff(P4(x), x);
  local dP5 := diff(P5(x), x);
  local dP6 := diff(P6(x), x);
  local dP7 := diff(P7(x), x);
  local dP8 := diff(P8(x), x);
  local dP9 := diff(P9(x), x);

  # Second order diffs

  local d2P0 := diff(dP0, x);
  local d2P1 := diff(dP1, x);
  local d2P2 := diff(dP2, x);
  local d2P3 := diff(dP3, x);
  local d2P4 := diff(dP4, x);
  local d2P5 := diff(dP5, x);
  local d2P6 := diff(dP6, x);
  local d2P7 := diff(dP7, x);
  local d2P8 := diff(dP8, x);
  local d2P9 := diff(dP9, x);

  # Smoothness

  local eq0 := subs(x = xs[2], dP0) = subs(x = xs[2], dP1);
  local eq1 := subs(x = xs[3], dP1) = subs(x = xs[3], dP2);
```

```
 local eq2 ≔ subs(x = xs[4], dP2) = subs(x = xs[4], dP3);
 local eq3 ≔ subs(x = xs[5], dP3) = subs(x = xs[5], dP4);
 local eq4 ≔ subs(x = xs[6], dP4) = subs(x = xs[6], dP5);
 local eq5 ≔ subs(x = xs[7], dP5) = subs(x = xs[7], dP6);
 local eq6 ≔ subs(x = xs[8], dP6) = subs(x = xs[8], dP7);
 local eq7 ≔ subs(x = xs[9], dP7) = subs(x = xs[9], dP8);
 local eq8 ≔ subs(x = xs[10], dP8) = subs(x = xs[10], dP9);

 local eq9 ≔ subs(x = xs[2], d2P0) = subs(x = xs[2], d2P1);
 local eq10 ≔ subs(x = xs[3], d2P1) = subs(x = xs[3], d2P2);
 local eq11 ≔ subs(x = xs[4], d2P2) = subs(x = xs[4], d2P3);
 local eq12 ≔ subs(x = xs[5], d2P3) = subs(x = xs[5], d2P4);
 local eq13 ≔ subs(x = xs[6], d2P4) = subs(x = xs[6], d2P5);
 local eq14 ≔ subs(x = xs[7], d2P5) = subs(x = xs[7], d2P6);
 local eq15 ≔ subs(x = xs[8], d2P6) = subs(x = xs[8], d2P7);
 local eq16 ≔ subs(x = xs[9], d2P7) = subs(x = xs[9], d2P8);
 local eq17 ≔ subs(x = xs[10], d2P8) = subs(x = xs[10], d2P9);

# Bounds

 local eq18  ≔ subs(x = xs[1], d2P0) = 0;
 local eq19  ≔ subs(x = xs[11], d2P9) = 0;

 #Basic equations

 local eq20 ≔ P0(xs[1]) = ys[1];
 local eq30 ≔ P0(xs[2]) = ys[2];

 local eq21 ≔ P1(xs[2]) = ys[2];
 local eq31 ≔ P1(xs[3]) = ys[3];

 local eq22 ≔ P2(xs[3]) = ys[3];
 local eq32 ≔ P2(xs[4]) = ys[4];

 local eq23 ≔ P3(xs[4]) = ys[4];
 local eq33 ≔ P3(xs[5]) = ys[5];

 local eq24 ≔ P4(xs[5]) = ys[5];
 local eq34 ≔ P4(xs[6]) = ys[6];

 local eq25 ≔ P5(xs[6]) = ys[6];
 local eq35 ≔ P5(xs[7]) = ys[7];

 local eq26 ≔ P6(xs[7]) = ys[7];
 local eq36 ≔ P6(xs[8]) = ys[8];

 local eq27 ≔ P7(xs[8]) = ys[8];
 local eq37 ≔ P7(xs[9]) = ys[9];
```

```
    local eq28 ≔ P8(xs[9]) = ys[9];
    local eq38 ≔ P8(xs[10]) = ys[10];

    local eq29 ≔ P9(xs[10]) = ys[10];
    local eq39 ≔ P9(xs[11]) = ys[11];

    # Final calcs

    local coefs ≔ solve({eq0, eq1, eq2, eq3, eq4, eq5, eq6, eq7, eq8, eq9, eq10, eq11, eq12, eq13,
        eq14, eq15, eq16, eq17, eq18, eq19, eq20, eq30, eq21, eq31, eq22, eq32, eq23, eq33, eq24,
        eq34, eq25, eq35, eq26, eq36, eq27, eq37, eq28, eq38, eq29, eq39}, {a0, a1, a2, a3, a4, a5,
        a6, a7, a8, a9, b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, d0,
        d1, d2, d3, d4, d5, d6, d7, d8, d9});

    local spline ≔ piecewise(0 ≤ curX and curX ≤ 0.1, P0(curX), 0.1 < curX and curX ≤ 0.2,
        P1(curX), 0.2 < curX and curX ≤ 0.3, P2(curX), 0.3 < curX and curX ≤ 0.4,
        P3(curX), 0.4 < curX and curX ≤ 0.5, P4(curX), 0.5 < curX and curX ≤ 0.6,
        P5(curX), 0.6 < curX and curX ≤ 0.7, P6(curX), 0.7 < curX and curX ≤ 0.8,
        P7(curX), 0.8 < curX and curX ≤ 0.9, P8(curX), 0.9 < curX and curX ≤ 1, P9(curX),
        0);

   return subs(coefs, spline);

    end proc
```

```
my_cubic_spline ≔ proc(xs, ys, curX)                                                        (1)
    local P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, dP0, dP1, dP2, dP3, dP4, dP5, dP6, dP7, dP8,
    dP9, d2P0, d2P1, d2P2, d2P3, d2P4, d2P5, d2P6, d2P7, d2P8, d2P9, eq0, eq1, eq2, eq3, eq4,
    eq5, eq6, eq7, eq8, eq9, eq10, eq11, eq12, eq13, eq14, eq15, eq16, eq17, eq18, eq19, eq20, eq30,
    eq21, eq31, eq22, eq32, eq23, eq33, eq24, eq34, eq25, eq35, eq26, eq36, eq27, eq37, eq28,
    eq38, eq29, eq39, coefs, spline;
    description "Cubic Spline Implementation";
    P0 ≔ x→a0 * (x − xs[1])^3 + b0 * (x − xs[1])^2 + c0 * (x − xs[1]) + d0;
    P1 ≔ x→a1 * (x − xs[2])^3 + b1 * (x − xs[2])^2 + c1 * (x − xs[2]) + d1;
    P2 ≔ x→a2 * (x − xs[3])^3 + b2 * (x − xs[3])^2 + c2 * (x − xs[3]) + d2;
    P3 ≔ x→a3 * (x − xs[4])^3 + b3 * (x − xs[4])^2 + c3 * (x − xs[4]) + d3;
    P4 ≔ x→a4 * (x − xs[5])^3 + b4 * (x − xs[5])^2 + c4 * (x − xs[5]) + d4;
    P5 ≔ x→a5 * (x − xs[6])^3 + b5 * (x − xs[6])^2 + c5 * (x − xs[6]) + d5;
    P6 ≔ x→a6 * (x − xs[7])^3 + b6 * (x − xs[7])^2 + c6 * (x − xs[7]) + d6;
    P7 ≔ x→a7 * (x − xs[8])^3 + b7 * (x − xs[8])^2 + c7 * (x − xs[8]) + d7;
    P8 ≔ x→a8 * (x − xs[9])^3 + b8 * (x − xs[9])^2 + c8 * (x − xs[9]) + d8;
    P9 ≔ x→a9 * (x − xs[10])^3 + b9 * (x − xs[10])^2 + c9 * (x − xs[10]) + d9;
    dP0 ≔ diff(P0(x), x);
```

```
dP1 := diff(P1(x), x);
dP2 := diff(P2(x), x);
dP3 := diff(P3(x), x);
dP4 := diff(P4(x), x);
dP5 := diff(P5(x), x);
dP6 := diff(P6(x), x);
dP7 := diff(P7(x), x);
dP8 := diff(P8(x), x);
dP9 := diff(P9(x), x);
d2P0 := diff(dP0, x);
d2P1 := diff(dP1, x);
d2P2 := diff(dP2, x);
d2P3 := diff(dP3, x);
d2P4 := diff(dP4, x);
d2P5 := diff(dP5, x);
d2P6 := diff(dP6, x);
d2P7 := diff(dP7, x);
d2P8 := diff(dP8, x);
d2P9 := diff(dP9, x);
eq0 := subs(x = xs[2], dP0) = subs(x = xs[2], dP1);
eq1 := subs(x = xs[3], dP1) = subs(x = xs[3], dP2);
eq2 := subs(x = xs[4], dP2) = subs(x = xs[4], dP3);
eq3 := subs(x = xs[5], dP3) = subs(x = xs[5], dP4);
eq4 := subs(x = xs[6], dP4) = subs(x = xs[6], dP5);
eq5 := subs(x = xs[7], dP5) = subs(x = xs[7], dP6);
eq6 := subs(x = xs[8], dP6) = subs(x = xs[8], dP7);
eq7 := subs(x = xs[9], dP7) = subs(x = xs[9], dP8);
eq8 := subs(x = xs[10], dP8) = subs(x = xs[10], dP9);
eq9 := subs(x = xs[2], d2P0) = subs(x = xs[2], d2P1);
eq10 := subs(x = xs[3], d2P1) = subs(x = xs[3], d2P2);
eq11 := subs(x = xs[4], d2P2) = subs(x = xs[4], d2P3);
eq12 := subs(x = xs[5], d2P3) = subs(x = xs[5], d2P4);
eq13 := subs(x = xs[6], d2P4) = subs(x = xs[6], d2P5);
eq14 := subs(x = xs[7], d2P5) = subs(x = xs[7], d2P6);
eq15 := subs(x = xs[8], d2P6) = subs(x = xs[8], d2P7);
eq16 := subs(x = xs[9], d2P7) = subs(x = xs[9], d2P8);
eq17 := subs(x = xs[10], d2P8) = subs(x = xs[10], d2P9);
eq18 := subs(x = xs[1], d2P0) = 0;
```

```
    eq19 := subs(x = xs[11], d2P9) = 0;
    eq20 := P0(xs[1]) = ys[1];
    eq30 := P0(xs[2]) = ys[2];
    eq21 := P1(xs[2]) = ys[2];
    eq31 := P1(xs[3]) = ys[3];
    eq22 := P2(xs[3]) = ys[3];
    eq32 := P2(xs[4]) = ys[4];
    eq23 := P3(xs[4]) = ys[4];
    eq33 := P3(xs[5]) = ys[5];
    eq24 := P4(xs[5]) = ys[5];
    eq34 := P4(xs[6]) = ys[6];
    eq25 := P5(xs[6]) = ys[6];
    eq35 := P5(xs[7]) = ys[7];
    eq26 := P6(xs[7]) = ys[7];
    eq36 := P6(xs[8]) = ys[8];
    eq27 := P7(xs[8]) = ys[8];
    eq37 := P7(xs[9]) = ys[9];
    eq28 := P8(xs[9]) = ys[9];
    eq38 := P8(xs[10]) = ys[10];
    eq29 := P9(xs[10]) = ys[10];
    eq39 := P9(xs[11]) = ys[11];
    coefs := solve({eq0, eq1, eq2, eq3, eq4, eq5, eq6, eq7, eq8, eq9, eq10, eq11, eq12, eq13, eq14,
    eq15, eq16, eq17, eq18, eq19, eq20, eq30, eq21, eq31, eq22, eq32, eq23, eq33, eq24, eq34,
    eq25, eq35, eq26, eq36, eq27, eq37, eq28, eq38, eq29, eq39}, {a0, a1, a2, a3, a4, a5, a6, a7,
    a8, a9, b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, d0, d1, d2, d3,
    d4, d5, d6, d7, d8, d9});
    spline := piecewise(0 <= curX and curX <= 0.1, P0(curX), 0.1 < curX and curX <= 0.2,
    P1(curX), 0.2 < curX and curX <= 0.3, P2(curX), 0.3 < curX and curX <= 0.4,
    P3(curX), 0.4 < curX and curX <= 0.5, P4(curX), 0.5 < curX and curX <= 0.6,
    P5(curX), 0.6 < curX and curX <= 0.7, P6(curX), 0.7 < curX and curX <= 0.8,
    P7(curX), 0.8 < curX and curX <= 0.9, P8(curX), 0.9 < curX and curX <= 1, P9(curX),
    0);
    return subs(coefs, spline)
end proc
```

```
> approximation_test := proc(u, f)
  description "Compares the values of the approximation (u) and the function (f)"

  local i := 0;
  local xs := [seq(i, i = 0 ..1, 0.05)];
```

```
    local u_ys := map(u, xs);
    local f_ys := map(f, xs);

     local d := [ ];

     for i from 1 to nops(xs) do
        d := [op(d), f_ys[i] − u_ys[i]];
     end do;

    return max(d);

    end proc
```

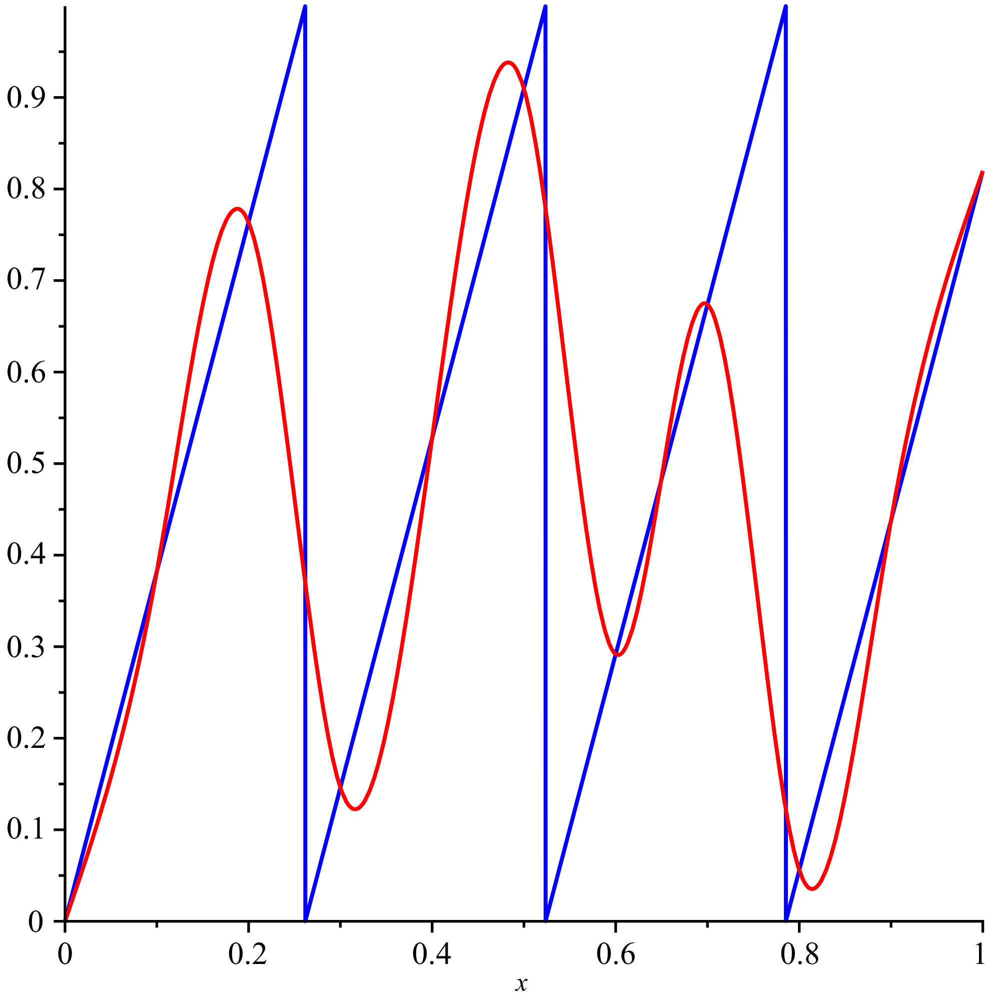
```
approximation_test := proc(u,f)                                        (2)
    local i, xs, u_ys, f_ys, d;
    description "Compares the values of the approximation (u) and the function (f)";
    i := 0;
    xs := [seq(i, i = 0 ..1, 0.05 ) ];
    u_ys := map(u, xs);
    f_ys := map(f, xs);
    d := [ ];
    for i to nops(xs) do d := [op(d), f_ys[i] − u_ys[i]] end do;
    return max(d)
end proc
```

```
> f := x → frac(12·x/π) ;

   A := [seq(i, i = 0 ..1, 0.1 ) ];
    B := [seq(f(i), i = 0 ..1, 0.1 ) ];

    approximation_test(x→my_cubic_spline(A, B, x), x→f(x));

    plot([f(x), my_cubic_spline(A, B, x) ], x = 0 ..1, color = [blue, red]);

    with(Student[NumericalAnalysis]);
    C := [ ];
   for i to 11 do
      C := [op(C), [A[i], B[i]]];
   end do;
   maple_spline := CubicSpline(C, independentvar = x, boundaryconditions = clamped(0, 1),
          bc_type = 'natural');
   Draw(maple_spline);
```

$$f := x \mapsto \mathrm{frac}\left(\frac{12\cdot x}{\pi}\right)$$

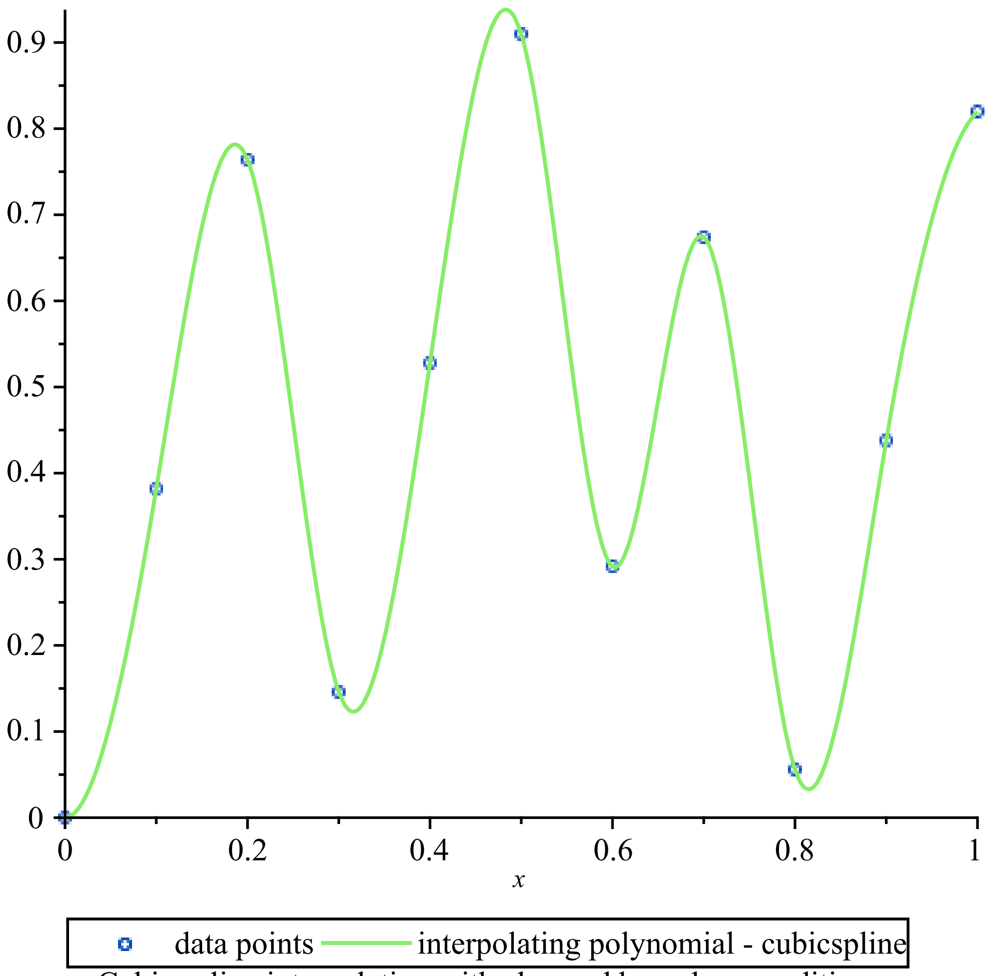
$$A := [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]$$

```
B := [0, 0.3819718633, 0.7639437268, 0.145915590, 0.527887454, 0.909859317, 0.291831180,
    0.673803044, 0.055774907, 0.437746770, 0.819718633]
                              0.4927758799
```



```
[AbsoluteError, AdamsBashforth, AdamsBashforthMoulton, AdamsMoulton, AdaptiveQuadrature,
    AddPoint, ApproximateExactUpperBound, ApproximateValue, BackSubstitution, BasisFunctions,
    Bisection, CubicSpline, DataPoints, Distance, DividedDifferenceTable, Draw, Euler, EulerTutor,
    ExactValue, FalsePosition, FixedPointIteration, ForwardSubstitution, Function,
    InitialValueProblem, InitialValueProblemTutor, Interpolant, InterpolantRemainderTerm,
    IsConvergent, IsMatrixShape, IterativeApproximate, IterativeFormula, IterativeFormulaTutor,
    LeadingPrincipalSubmatrix, LinearSolve, LinearSystem, MatrixConvergence,
    MatrixDecomposition, MatrixDecompositionTutor, ModifiedNewton, NevilleTable, Newton,
    NumberOfSignificantDigits, PolynomialInterpolation, Quadrature, RateOfConvergence,
    RelativeError, RemainderTerm, Roots, RungeKutta, Secant, SpectralRadius, Steffensen, Taylor,
    TaylorPolynomial, UpperBoundOfRemainderTerm, VectorLimit]
                              C := [ ]
                            C := [[0, 0]]
                    C := [[0, 0], [0.1, 0.3819718633]]
```

```
                        C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268]]
                 C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590]]
  C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590], [0.4, 0.527887454]]
C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590], [0.4, 0.527887454],
    [0.5, 0.909859317]]
C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590], [0.4, 0.527887454],
    [0.5, 0.909859317], [0.6, 0.291831180]]
C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590], [0.4, 0.527887454],
    [0.5, 0.909859317], [0.6, 0.291831180], [0.7, 0.673803044]]
C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590], [0.4, 0.527887454],
    [0.5, 0.909859317], [0.6, 0.291831180], [0.7, 0.673803044], [0.8, 0.055774907]]
C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590], [0.4, 0.527887454],
    [0.5, 0.909859317], [0.6, 0.291831180], [0.7, 0.673803044], [0.8, 0.055774907], [0.9,
    0.437746770]]
C := [[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3, 0.145915590], [0.4, 0.527887454],
    [0.5, 0.909859317], [0.6, 0.291831180], [0.7, 0.673803044], [0.8, 0.055774907], [0.9,
    0.437746770], [1.0, 0.819718633]]
maple_spline := POLYINTERP([[0, 0], [0.1, 0.3819718633], [0.2, 0.7639437268], [0.3,
    0.145915590], [0.4, 0.527887454], [0.5, 0.909859317], [0.6, 0.291831180], [0.7, 0.673803044],
    [0.8, 0.055774907], [0.9, 0.437746770], [1.0, 0.819718633]], independentvar = x,
    boundaryconditions = clamped(0, 1), bc_type = natural, INFO)
```



Cubic spline interpolation with clamped boundary conditions.

>