Find a circular Catmull-Rom spline. The parameters are the list of data points, and the variable to be used as the unknown in the polynomials. The result is a piecewise polynomial function (packaged up by the Maple piecewise function).

```
> circspline := proc (x, t)

    local n, a, b, c, d, poly, dpoly, lefti, righti, k,
          eqs, lefts, rights, pieces;

    # Find how many data points (and hence pieces) we have.

    n := nops(x);

    # Set up a polynomial and its deriviative in which the
    # coefficients are unknown.

    poly := a + b*t + c*t^2 + d*t^3;
    dpoly := diff(poly,t);

    # Find the polynomial for each piece, creating the input
    # for the piecewise function in the variable pieces.

    pieces := NULL;

    for k from 1 to n do

      # Set lefti to the index in xi of the data point at the
      # left end of this piece.

      if k=1 then
        lefti := n;
      else
        lefti := k-1;
      fi;

      # Set righti to the index in xi of the data point at the
      # right end of this piece.

      righti := k;

      # Set lefts to the slope at the left end of this piece.

      if lefti=1 then
        lefts := (x[righti]-x[n])*n/2;
      else
        lefts := (x[righti]-x[lefti-1])*n/2;
      fi;
```

```
    # Set righti to the slope at the right end of this piece.

    if righti=n then
      rights := (x[1]-x[lefti])*n/2;
    else
      rights := (x[righti+1]-x[lefti])*n/2;
    fi;

    # Set eqs to the set of equations that must be satisfied
    # by this piece.

    eqs := { subs(t=(k-1)/n,poly) = x[lefti],
             subs(t=k/n,poly) = x[righti],
             subs(t=(k-1)/n,dpoly) = lefts,
             subs(t=k/n,dpoly) = rights };

    # Solve these equations, substitute the values found
    # into the polynomial, and use this as the polynomial
    # for the range of t values for this piece.

    pieces := pieces,
                t<=k/n, subs(solve(eqs,{a,b,c,d}),poly);
  od;

  # Create the final piecewise polynomial function.

  piecewise(pieces);

end:
```

Find basis functions for a circular spline. The sequence of basis functions are computed using the circspline procedure, and saved in the global variable basis. The parameters are the number of data points and the unknown variable to use.

```
> circbasis := proc (n, t)
    global basis;
    local i;
    basis := NULL;
    for i from 1 to n do
      basis := basis, circspline([0$(i-1),1,0$(n-i)],t);
    od;
    NULL;
  end:
```

Find circular spline from the basis functions pre-computed by circbasis. The

parameter is the list of data points, which must be the same length as the number given in the previous call of circbasis.

```
> csb := proc (x)
    global basis;
    local i;
    simplify (sum (x[i]*basis[i], i=1..nops(x)));
  end:
>
```