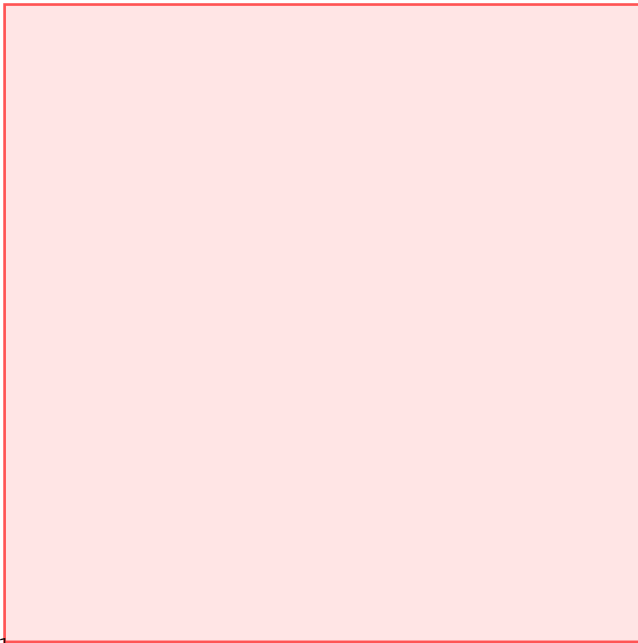```
Off[InterpolatingFunction::dmval];

DynamicModule[
  {
   pts = {{0, 0}, {0.4, 0}, {0.5, 0.6}, {0.6, 0}, {1, 0}},
   n, x, y
  },
  n = Length[pts];
  cpts = Table[{x[i], y[i]}, {i, n}];
  Evaluate[cpts] = pts;
  LocatorPane[
   Dynamic[cpts],
   Dynamic[
    c = BezierCurve[Evaluate[cpts], SplineDegree → 2];
    Graphics[c, PlotRange → {{0, 1}, {0, 1}}]
   ],
   {{0, 0}, {1, 1}},
   Appearance → Range[n]
  ]
]
```

```
DynamicModule[
 {n = 7, pts, x, y},
 pts = Table[{x[i], y[i]}, {i, n}];
 Evaluate[pts] = Table[{i, i} / (n + 1), {i, n}];
 LocatorPane[
  Dynamic[pts],
  Dynamic[
   r = BezierCurve[Evaluate[pts], SplineDegree → 3];
   Graphics[r, PlotRange → {{0, 1}, {0, 1}}]
  ],
  {{0, 0}, {1, 1}},
  Appearance → Range[n]
 ]
]
```
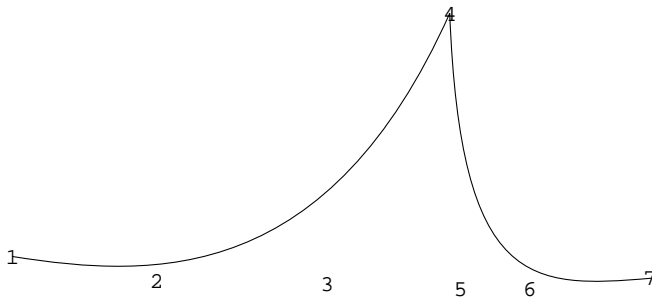
```
Dynamic[
 c3[t_] := Append[BezierFunction[First@c][t], 0];
 r1[t_] := Last[BezierFunction[First@r][t]];
 Graphics3D[{
   CapForm[None], JoinForm["Miter"], Opacity[0.5],
   Tube[
    Table[c3[t], {t, 0, 1, 0.05}],
    Table[r1[t], {t, 0, 1, 0.05}]
   ]
  }]
]
```
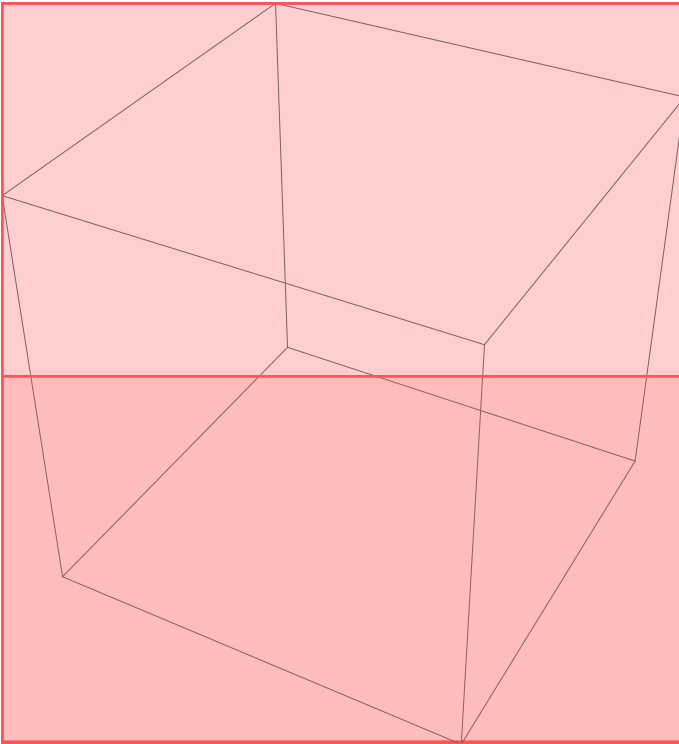

c3[0.3]

{0.3504, 0.15876, 0}

c // FullForm

BezierCurve[List[List[0, 0], List[0.4`, 0], List[0.5`, 0.6`], List[0.6`, 0], List[1, 0]],
 Rule[SplineDegree, 2]]

Length[c]

2

BezierFunction[{{0, 0}, {1, 1}, {3, 2}}, SplineDegree → 2]

BezierFunction[{{0., 1.}}, <>]

```
t = BezierFunction @@ c
```

BezierFunction::invdeg : Value of option SplineDegree–>2 should be a positive integer, or a list of positive integers. ≫

```
BezierFunction[{{0, 0}, {0.4, 0}, {0.5, 0.6}, {0.6, 0}, {1, 0}}, SplineDegree → 2]
```

```
t[1]
```

```
BezierFunction[{{0, 0}, {0.4, 0}, {0.5, 0.6}, {0.6, 0}, {1, 0}}, SplineDegree → 2][1]
```

```
BezierFunction @@ c
```

BezierFunction::invdeg : Value of option SplineDegree–>2 should be a positive integer, or a list of positive integers. ≫

```
BezierFunction[{{0, 0}, {0.4, 0}, {0.5, 0.6}, {0.6, 0}, {1, 0}}, SplineDegree → 2]
```

```
BezierFunction[
  {{0.`, 0.`}, {0.386`, 0.1`}, {0.5`, 0.6`}, {0.686`, 0.126`}, {1.`, 0.`}}, SplineDegree → 2]
```

BezierFunction::argt : BezierFunction called with 0 arguments; 1 or 2 arguments are expected. ≫

```
BezierFunction[]
```

```
Graphics3D[{
  CapForm[None], JoinForm["Miter"], Opacity[0.5],
  Tube[
   Table[{x, y, 0}, {t, 0, 0.75, 0.05}],
   Table[r, {t, 0, 0.75, 0.05}]
  ],
  Tube[
   Table[{x, y, 0}, {t, -0.75, 0, 0.05}],
   Table[r, {t, -0.75, 0, 0.05}]
  ]
 }]
```