```
1  using System;
2  using System.Windows.Forms;
3  using SldWorks;
4  using SwConst;
5  using System.Runtime.InteropServices;
6
7  namespace SW_Macro_FeatAnalysis
8  {
9      public partial class Form1 : Form
10     {
11         SldWorks.SldWorks swApp;
12
13         public Form1()
14         {
15             InitializeComponent();
16         }
17
18         // Verbindet mit SolidWorks
19         private void b_startSLDWorks_Click(object sender, EventArgs e)
20         {
21             try
22             {
23                 swApp = (SldWorks.SldWorks)Marshal.GetActiveObject
                     ("SldWorks.Application");
24             }
25             catch (Exception)
26             {
27                 swApp = new SldWorks.SldWorks();
28                 swApp.Visible = true;
29             }
30         }
31
32         // Durchläuft alle Feature des geladenenen Teils und sucht nach
             Kreis-Elementen, gibt diese in ListBox aus
33         private void b_analyze_Click_Click(object sender, EventArgs e)
34         {
35             //Variablen
36             ModelDoc2 swModel;
37             Feature swFeature, subFeature;
38             String FeatureName, FeatureType;
39             Sketch swSketch;
40             Object[] Segments;
41             SketchPoint cPoint;
42
43             // Bereinigen der ListBox
44             lb_output.Items.Clear();
45
46             // sldprt breits geöffnent, daher nur auslesen des bereits
                 geöffneten Teils
47             swModel = (ModelDoc2)swApp.ActiveDoc;
48             swFeature = (Feature)swModel.FirstFeature();
49
50             // Durchlaufen aller Features
```

```
51              while (swFeature != null)
52              {
53                  // Output des Featurenamen und Typs in ListBox
54                  lb_output.Items.Add(swFeature.Name + " " +
                      swFeature.GetTypeName());
55
56                  // Überprüfen, ob Feature vom Typ Schnitt
57                  // weiterhin, ob Name "Bohrbild"
58                  if (swFeature.GetTypeName() == "Cut" && swFeature.Name ==
                      "Bohrbild")
59                  {
60                      // Auslesen des ersten Sub-Features
61                      subFeature = (Feature)swFeature.GetFirstSubFeature();
62
63                      // Durchlaufen aller Sub-Features
64                      while (subFeature != null)
65                      {
66                          FeatureName = subFeature.Name;
67                          FeatureType = subFeature.GetTypeName();
68
69                          // Output des Sub-Feature Name und Typs in ListBox
70                          lb_output.Items.Add("\t" + FeatureName + "\t" +
                        FeatureType);
71
72                          // Überprüfen, ob es sich um Skizze handelt
                        (ProfilFeature)
73                          if (FeatureType == "ProfileFeature")
74                          {
75                              // Auslesen des spezifischen Features
76                              swSketch = (Sketch)
                            subFeature.GetSpecificFeature2();
77                              // Auslesen der Skizzenegmente
78                              Segments = (Object[])swSketch.GetSketchSegments
                            ();
79
80                              // Durchlaufe alle Sketch-Segmente
81                              foreach (SketchSegment sketchSeg in Segments)
82                              {
83                                  // Prüfen, ob aktuelles Sketch-Segment vom
                                Typ swSketchARC
84                                  if (sketchSeg.GetType() == (int)
                                swSketchSegments_e.swSketchARC)
85                                  {
86                                      // Umwandeln in Skizzenkreis
87                                      SketchArc arc = (SketchArc)sketchSeg;
88                                      // Auslesen Mittelpunkt
89                                      cPoint = (SketchPoint)
                                    arc.GetCenterPoint2();
90
91                                      // Ausgeben in Listbox
92                                      lb_output.Items.Add("\tKreis: X = " +
                                    cPoint.X + "\tY=" + cPoint.Y + "\tZ=" + cPoint.Z +
93                                          "\tR=" + arc.GetRadius());
```

```
                                }
                            }
                        }

                        // nächstes Sub-Feature
                        // wenn keines mehr vorhanden, liefert "null" zurück und bricht aus Schleife aus
                        subFeature = (Feature)subFeature.GetNextSubFeature();
                    }
                }
                // nächstes Feature
                swFeature = (Feature)swFeature.GetNextFeature();
            }
        }

        private void b_about_Click(object sender, EventArgs e)
        {
            MessageBox.Show("Kleines Makro, das eine Baugruppe analysiert und alle Feature ausgibt. Außerdem zeigt es die Koordinaten der Bohrungen an. ");
        }
    }
}
```