```c
#include "includetest5.h"

int Anzahl_der_Daten = 0,f_Anzahl_der_Verfeinerungen = 0,Anzahl_der_Verfeinerungen = 0,Anzahl_der_Dreiecke
= 0,
p4_p5_Punkte_pruefen = 0,Punkte_pruefen = 0,p5_p6_Punkte_pruefen = 0,p6_p4_Punkte_pruefen = 0,
p4_p9_Punkte_pruefen = 0,p5_p10_Punkte_pruefen = 0,p6_p11_Punkte_pruefen = 0,p5_p11_Punkte_pruefen = 0,
p6_p10_Punkte_pruefen,
p4_p10_Punkte_pruefen = 0,p5_p9_Punkte_pruefen = 0,p6_p9_Punkte_pruefen = 0,p4_p11_Punkte_pruefen = 0,
Anzahl = 0,Position = 0,Zaehler = 0,
f_Anzahl_der_Dreiecke[2],Dreiecke_pruefen[4];
double x = 0.0,y = 0.0,xA[4][3],yA[4][3];
point temp,p4,p5,p6,p7,p8,p9,p10,p11,p13,f_p[4][3],p3[4][3],p12[4][3],ps[4][3];

char Text[128];

const char *Anzahl_der_Daten_Textdatei = "Aufgabe 3 Anzahl der Daten5.txt",*Ausgabe_Textdatei = "Ausgabe
Aufgabe 3 test5.txt";

void f_p0_Koordinaten_setzen()
{
    xA[0][0] = 0.0;
    yA[0][0] = 0.0;
    xA[1][0] = 0.0;
    yA[1][0] = 0.0;
    xA[2][0] = 0.0;
    yA[2][0] = 0.0;
    xA[3][0] = 1.0;
    yA[3][0] = 0.0;
}

void f_p1_Koordinaten_setzen()
{
    xA[0][1] = 1.0;
    yA[0][1] = 0.0;
    xA[1][1] = 1.0;
    yA[1][1] = 0.0;
    xA[2][1] = 0.5;
    yA[2][1] = 1.0;
    xA[3][1] = 0.5;
    yA[3][1] = 1.0;
}

void f_p2_Koordinaten_setzen()
{
    xA[0][2] = 0.5;
    yA[0][2] = 1.0;
    xA[1][2] = 0.5;
    yA[1][2] = -1.0;
    xA[2][2] = 0.0;
    yA[2][2] = 0.5;
    xA[3][2] = 1.0;
    yA[3][2] = 0.5;
}

point set_point()
{
    temp.x = x;
    temp.y = y;

    return(temp);
}

void Standardwerte()
{
    f_Anzahl_der_Verfeinerungen =  1;

```

```
64      f_Anzahl_der_Dreiecke[0] = 1;
65      f_Anzahl_der_Dreiecke[1] = 4;
66
67      for(int i = 0;i < 3;i++)
68      {
69          switch(i)
70          {
71              case 0:f_p0_Koordinaten_setzen();
72                     break;
73              case 1:f_p1_Koordinaten_setzen();
74                     break;
75              case 2:f_p2_Koordinaten_setzen();
76                     break;
77          }
78      }
79      for(int i = 0;i < 4;i++)
80      {
81          for(int j = 0;j < 3;j++)
82          {
83              x = xA[i][j];
84              y = yA[i][j];
85              f_p[i][j] = set_point();
86          }
87      }
88  }
89
90  void Anzahl_der_Verfeinerungen_setzen()
91  {
92      Anzahl_der_Verfeinerungen = 0;
93  }
94
95  void korrekte_Anzahl_der_Verfeinerungen()
96  {
97      Anzahl_der_Verfeinerungen_setzen();
98      if(Anzahl_der_Verfeinerungen <= 0)
99          Anzahl_der_Verfeinerungen = f_Anzahl_der_Verfeinerungen;
100 }
101
102 void Anzahl_der_Dreiecke_setzen()
103 {
104     Anzahl_der_Dreiecke = 1;
105 }
106
107 void korrekte_Anzahl_der_Dreiecke()
108 {
109     Anzahl_der_Dreiecke_setzen();
110     if(Anzahl_der_Dreiecke < 1)
111         Anzahl_der_Dreiecke = f_Anzahl_der_Dreiecke[0];
112     if(Anzahl_der_Dreiecke > 4)
113         Anzahl_der_Dreiecke = f_Anzahl_der_Dreiecke[1];
114 }
115
116 void korrekte_Werte()
117 {
118     korrekte_Anzahl_der_Verfeinerungen();
119     korrekte_Anzahl_der_Dreiecke();
120 }
121
122 void null_Koordinaten_setzen()
123 {
124     xA[0][0] = 0.0;
125     yA[0][0] = 0.0;
126     xA[0][1] = 0.0;
127     yA[0][1] = 0.0;
128     xA[0][2] = 0.0;
129     yA[0][2] = 0.0;
```

```
}

void Punkte_setzen_1_Dreieck()
{
    null_Koordinaten_setzen();
    for(int i = 0;i < 3;i++)
    {
        x = xA[0][i];
        y = yA[0][i];
        p3[0][i] = set_point();
    }
}

void eins_Koordinaten_setzen()
{
    xA[1][0] = 0.0;
    yA[1][0] = 0.0;
    xA[1][1] = 0.0;
    yA[1][1] = 0.0;
    xA[1][2] = 0.0;
    yA[1][2] = 0.0;
}

void Punkte_setzen_2_Dreiecke()
{
    Punkte_setzen_1_Dreieck();
    eins_Koordinaten_setzen();
    for(int i = 0;i < 3;i++)
    {
        x = xA[1][i];
        y = yA[1][i];
        p3[1][i] = set_point();
    }
}

void zwei_Koordinaten_setzen()
{
    xA[2][0] = 0.0;
    yA[2][0] = 0.0;
    xA[2][1] = 0.0;
    yA[2][1] = 0.0;
    xA[2][2] = 0.0;
    yA[2][2] = 0.0;
}

void Punkte_setzen_3_Dreiecke()
{
    Punkte_setzen_2_Dreiecke();
    zwei_Koordinaten_setzen();
    for(int i = 0;i < 3;i++)
    {
        x = xA[2][i];
        y = yA[2][i];
        p3[2][i] = set_point();
    }
}

void drei_Koordinaten_setzen()
{
    xA[3][0] = 0.0;
    yA[3][0] = 0.0;
    xA[3][1] = 0.0;
    yA[3][1] = 0.0;
    xA[3][2] = 0.0;
    yA[3][2] = 0.0;
}
```

```
196
197 void Punkte_setzen_4_Dreiecke()
198 {
199     Punkte_setzen_3_Dreiecke();
200     drei_Koordinaten_setzen();
201     for(int i = 0;i < 3;i++)
202     {
203         x = xA[3][i];
204         y = yA[3][i];
205         p3[3][i] = set_point();
206     }
207 }
208
209 int Punkte_ueberpruefen()
210 {
211     Punkte_pruefen = 0;
212     if((p7.x != p8.x) || (p7.y != p8.y))
213         Punkte_pruefen = 1;
214
215     return(Punkte_pruefen);
216 }
217
218 void Dreiecke_ueberpruefen()
219 {
220     for(int i = 0;i < Anzahl_der_Dreiecke;i++)
221         Dreiecke_pruefen[i] = 0;
222     for(int i = 0;i < Anzahl_der_Dreiecke;i++)
223     {
224         p4 = p3[i][0];
225         p5 = p3[i][1];
226         p6 = p3[i][2];
227
228         p7 = p4;
229         p8 = p5;
230         p4_p5_Punkte_pruefen = Punkte_ueberpruefen();
231
232         p7 = p5;
233         p8 = p6;
234         p5_p6_Punkte_pruefen = Punkte_ueberpruefen();
235
236         p7 = p6;
237         p8 = p4;
238         p6_p4_Punkte_pruefen = Punkte_ueberpruefen();
239
240         if((p4_p5_Punkte_pruefen == 0) || (p5_p6_Punkte_pruefen == 0) || (p6_p4_Punkte_pruefen == 0))
241             Dreiecke_pruefen[i] = 1;
242     }
243     for(int i = 0;i < Anzahl_der_Dreiecke - 1;i++)
244     {
245         if(Dreiecke_pruefen[i] == 0)
246         {
247             p4 = p3[i][0];
248             p5 = p3[i][1];
249             p6 = p3[i][2];
250
251             for(int j = i + 1;j < Anzahl_der_Dreiecke;j++)
252             {
253                 if(Dreiecke_pruefen[j] == 0)
254                 {
255                     p9 = p3[j][0];
256                     p10 = p3[j][1];
257                     p11 = p3[j][2];
258
259                     p7 = p4;
260                     p8 = p9;
261                     p4_p9_Punkte_pruefen = Punkte_ueberpruefen();
```

```
262
263                     p7 = p5;
264                     p8 = p10;
265                     p5_p10_Punkte_pruefen = Punkte_ueberpruefen();
266
267                     p7 = p6;
268                     p8 = p11;
269                     p6_p11_Punkte_pruefen = Punkte_ueberpruefen();
270
271                     p7 = p5;
272                     p8 = p11;
273                     p5_p11_Punkte_pruefen = Punkte_ueberpruefen();
274
275                     p7 = p6;
276                     p8 = p10;
277                     p6_p10_Punkte_pruefen = Punkte_ueberpruefen();
278
279                     p7 = p4;
280                     p8 = p10;
281                     p4_p10_Punkte_pruefen = Punkte_ueberpruefen();
282
283                     p7 = p5;
284                     p8 = p9;
285                     p5_p9_Punkte_pruefen = Punkte_ueberpruefen();
286
287                     p7 = p6;
288                     p8 = p9;
289                     p6_p9_Punkte_pruefen = Punkte_ueberpruefen();
290
291                     p7 = p4;
292                     p8 = p11;
293                     p4_p11_Punkte_pruefen = Punkte_ueberpruefen();
294
295                     if(((p4_p9_Punkte_pruefen == 0) && (p5_p10_Punkte_pruefen == 0) && (
p6_p11_Punkte_pruefen == 0)) ||
296                        ((p4_p9_Punkte_pruefen == 0) && (p5_p11_Punkte_pruefen == 0) && (
p6_p10_Punkte_pruefen == 0)) ||
297                        ((p4_p10_Punkte_pruefen == 0) && (p5_p9_Punkte_pruefen == 0) && (
p6_p11_Punkte_pruefen == 0)) ||
298                        ((p4_p10_Punkte_pruefen == 0) && (p5_p11_Punkte_pruefen == 0) && (
p6_p9_Punkte_pruefen == 0)) ||
299                        ((p4_p11_Punkte_pruefen == 0) && (p5_p9_Punkte_pruefen == 0) && (
p6_p10_Punkte_pruefen == 0)) ||
300                        ((p4_p11_Punkte_pruefen == 0) && (p5_p10_Punkte_pruefen == 0) && (
p6_p9_Punkte_pruefen == 0)))
301                         Dreiecke_pruefen[j] =  1;
302                 }
303             }
304         }
305     }
306 }
307
308 void korrekte_Punkte()
309 {
310     switch(Anzahl_der_Dreiecke)
311     {
312         case 1:Punkte_setzen_1_Dreieck();
313                break;
314         case 2:Punkte_setzen_2_Dreiecke();
315                break;
316         case 3:Punkte_setzen_3_Dreiecke();
317                break;
318         case 4:Punkte_setzen_4_Dreiecke();
319                break;
320     }
321     Dreiecke_ueberpruefen();
```

```
322     for(int i = 0;i < Anzahl_der_Dreiecke;i++)
323     {
324         if(Dreiecke_pruefen[i] == 1)
325         {
326             for(int j = 0;j < 3;j++)
327                 p3[i][j] = f_p[i][j];
328         }
329     }
330 }
331
332 triangle *create_triangle(const point p0,const point p1,const point p2)
333 {
334     static triangle *first_triangle = nil;
335     triangle *nf = malloc(sizeof(triangle));
336
337     nf->pt[0] = p0;
338     nf->pt[1] = p1;
339     nf->pt[2] = p2;
340
341     nf->next = first_triangle;
342     first_triangle = nf;
343
344     return(first_triangle);
345 }
346
347 void p9_setzen()
348 {
349     x = (p7.x + p8.x) / 2;
350     y = (p7.y + p8.y) / 2;
351     p9 = set_point();
352 }
353
354 void p10_setzen()
355 {
356     x = (p7.x + p8.x) / 2;
357     y = (p7.y + p8.y) / 2;
358     p10 = set_point();
359 }
360
361 void p11_setzen()
362 {
363     x = (p7.x + p8.x) / 2;
364     y = (p7.y + p8.y) / 2;
365     p11 = set_point();
366 }
367
368 void ps_Dreieck_setzen()
369 {
370     ps[Zaehler][0] = p7;
371     ps[Zaehler][1] = p8;
372     ps[Zaehler][2] = p9;
373 }
374
375 void Punkte_setzen()
376 {
377     for(int i = 0;i < 4;i++)
378     {
379         printf("Position = %d\n",Position);
380         for(int j = 0;j < 3;j++)
381             p3[Position][j] = ps[i][j];
382         Position++;
383     }
384 }
385
386 void refine_element()
387 {
```

```
388      p4 = p12[Zaehler][0];
389      p5 = p12[Zaehler][1];
390      p6 = p12[Zaehler][2];
391
392      p7 = p4;
393      p8 = p5;
394      p9_setzen();
395
396      p7 = p5;
397      p8 = p6;
398      p10_setzen();
399
400      p7 = p6;
401      p8 = p4;
402      p11_setzen();
403
404      printf("p9 = (%f,%f)\n",p9.x,p9.y);
405      printf("p10 = (%f,%f)\n",p10.x,p10.y);
406      printf("p11 = (%f,%f)\n",p11.x,p11.y);
407
408      for(int i = 1;i <= 4;i++)
409      {
410          switch(i)
411          {
412              case 1:p7 = p9;
413                     p8 = p10;
414                     p13 = p11;
415                     break;
416              case 2:p7 = p4;
417                     p8 = p9;
418                     p13 = p11;
419                     break;
420              case 3:p7 = p9;
421                     p8 = p5;
422                     p13 = p10;
423                     break;
424              case 4:p7 = p10;
425                     p8 = p6;
426                     p13 = p11;
427                     break;
428          }
429          printf("i = %d\n",i);
430          printf("p7 = (%f,%f)\n",p7.x,p7.y);
431          printf("p8 = (%f,%f)\n",p8.x,p8.y);
432          printf("p13 = (%f,%f)\n",p13.x,p13.y);
433          Zaehler = i - 1;
434          ps_Dreieck_setzen();
435      }
436      for(int i = 0;i < 4;i++)
437      {
438          for(int j = 0;j < 3;j++)
439              printf("ps(%d) = (%f,%f)\n",i,ps[i][j].x,ps[i][j].y);
440      }
441      Punkte_setzen();
442  }
443
444  triangle *einhaengen(triangle *original)
445  {
446      triangle *y = original,*neu = nil;
447
448      neu = create_triangle(p3[Zaehler][0],p3[Zaehler][1],p3[Zaehler][2]);
449      neu->next = y;
450      y = neu;
451
452      return(neu);
453  }
```

```
454
455 triangle *refine(triangle *original)
456 {
457     triangle *y = original,*refine = nil;
458     Anzahl = Anzahl_der_Dreiecke;
459     Position = Anzahl;
460     Anzahl_der_Dreiecke = 5 * Anzahl_der_Dreiecke; // Anzahl_der_Dreiecke = Anzahl_der_Dreiecke + 4 *
Anzahl_der_Dreiecke = 5 * Anzahl_der_Dreiecke
461     for(int i = 0;i < Anzahl;i++)
462     {
463         for(int j = 0;j < 3;j++)
464         {
465             p3[i][j] = y->pt[j];
466             p12[i][j] = y->pt[j];
467         }
468         y = y->next;
469     }
470     for(int i = 0;i < Anzahl;i++)
471     {
472         Zaehler = i;
473         refine_element();
474     }
475     Dreiecke_ueberpruefen();
476
477     Anzahl = Anzahl_der_Dreiecke;
478     Anzahl_der_Dreiecke = 0;
479     for(int i = Anzahl - 1;i >= 0;i--)
480     {
481         printf("Dreiecke pruefen(%d) = %d\n",i,Dreiecke_pruefen[i]);
482         if(Dreiecke_pruefen[i] == 0)
483         {
484             Zaehler = i;
485             refine = einhaengen(refine);
486             Anzahl_der_Dreiecke++;
487         }
488     }
489
490     return(refine);
491 }
492
493 void print_all_triangles(triangle *first)
494 {
495   for (; first; first = first->next)
496   {
497     printf("(%f %f),(%f %f),(%f %f)\n", first->pt[0].x, first->pt[0].y, first->pt[1].x, first->pt[1].y,
first->pt[2].x, first->pt[2].y);
498     sprintf(Text,"(%f %f),(%f %f),(%f %f)\r\n", first->pt[0].x, first->pt[0].y, first->pt[1].x, first->pt[1
].y, first->pt[2].x, first->pt[2].y);
499     Anzahl_der_Daten = speichern_in_Datei(Anzahl_der_Daten,Ausgabe_Textdatei,Text);
500   }
501 }
502
503 int main()
504 {
505   triangle *anchor;
506
507   Anzahl_der_Daten = eingabe_int_aus_Textdatei(Anzahl_der_Daten_Textdatei);
508   Standardwerte();
509   korrekte_Werte();
510
511   korrekte_Punkte();
512   for(int i = Anzahl_der_Dreiecke - 1;i >= 0;i--)
513     anchor = create_triangle(p3[i][0],p3[i][1],p3[i][2]);
514   for(int i = 1;i <= Anzahl_der_Verfeinerungen;i++)
515     anchor = refine(anchor);
516   print_all_triangles(anchor);
```

```
517
518     sprintf(Text,"%d\r\n",Anzahl_der_Daten);
519     ausgabe_in_Textdatei(Anzahl_der_Daten_Textdatei,"w",Text);
520
521     return(0);
522   }
```