```cpp
#include "dreieck.h"

dreieck::dreieck()
{
    Eingabe_Dreieck.~eingabe();
    Punkt_Dreieck.~punkt();
    index = 0;
    Anzahl_der_Verfeinerungen = 0;
    Anzahl_der_Dreiecke = 0;
    f_Anzahl_der_Verfeinerungen = 0;
    p2_p3_Punkte_pruefen = 0;
    p3_p4_Punkte_pruefen = 0;
    p4_p2_Punkte_pruefen = 0;
    p2_p5_Punkte_pruefen = 0;
    p3_p6_Punkte_pruefen = 0;
    p4_p7_Punkte_pruefen = 0;
    p3_p7_Punkte_pruefen = 0;
    p4_p6_Punkte_pruefen = 0;
    p2_p6_Punkte_pruefen = 0;
    p3_p5_Punkte_pruefen = 0;
    p4_p5_Punkte_pruefen = 0;
    p2_p7_Punkte_pruefen = 0;
    Anzahl = 0;
    Position = 0;
    Zaehler = 0;
}

dreieck::~dreieck()
{
    //dtor
}

void dreieck::f_p0_Koordinaten_setzen()
{
    x[0][0] = 0.0;
    y[0][0] = 0.0;
    x[1][0] = 0.0;
    y[1][0] = 0.0;
    x[2][0] = 0.0;
    y[2][0] = 0.0;
    x[3][0] = 1.0;
    y[3][0] = 0.0;
}

void dreieck::f_p1_Koordinaten_setzen()
{
    x[0][1] = 1.0;
    y[0][1] = 0.0;
    x[1][1] = 1.0;
    y[1][1] = 0.0;
    x[2][1] = 0.5;
    y[2][1] = 1.0;
    x[3][1] = 0.5;
    y[3][1] = 1.0;
}

void dreieck::f_p2_Koordinaten_setzen()
{
    x[0][2] = 0.5;
    y[0][2] = 1.0;
    x[1][2] = 0.5;
    y[1][2] = -1.0;
    x[2][2] = 0.0;
    y[2][2] = 0.5;
    x[3][2] = 1.0;
    y[3][2] = 0.5;
```

```
}

void dreieck::Standardwerte()
{
    f_Anzahl_der_Verfeinerungen = 1;

    f_Anzahl_der_Dreiecke[0] = 1;
    f_Anzahl_der_Dreiecke[1] = 4;

    for(int i = 0;i < 3;i++)
    {
        switch(i)
        {
            case 0:f_p0_Koordinaten_setzen();
                   break;
            case 1:f_p1_Koordinaten_setzen();
                   break;
            case 2:f_p2_Koordinaten_setzen();
                   break;
        }
    }

    for(int i = 0;i < 4;i++)
    {
        for(int j = 0;j < 3;j++)
        {
            Punkt_Dreieck.x = x[i][j];
            Punkt_Dreieck.y = y[i][j];
            Punkt_Dreieck.set_point();
            f_p[i][j] = Punkt_Dreieck.temp;
        }
    }
}

void dreieck::Anzahl_der_Verfeinerungen_setzen()
{
    Anzahl_der_Verfeinerungen = 0;
}

void dreieck::korrekte_Anzahl_der_Verfeinerungen()
{
    Anzahl_der_Verfeinerungen_setzen();
    if(Anzahl_der_Verfeinerungen <= 0)
        Anzahl_der_Verfeinerungen = f_Anzahl_der_Verfeinerungen;
}

void dreieck::Anzahl_der_Dreiecke_setzen()
{
    Anzahl_der_Dreiecke = 0;
}

void dreieck::korrekte_Anzahl_der_Dreiecke()
{
    Anzahl_der_Dreiecke_setzen();
    if(Anzahl_der_Dreiecke < 1)
        Anzahl_der_Dreiecke = f_Anzahl_der_Dreiecke[0];
    if(Anzahl_der_Dreiecke > 4)
        Anzahl_der_Dreiecke = f_Anzahl_der_Dreiecke[1];
}

void dreieck::korrekte_Werte()
{
    korrekte_Anzahl_der_Verfeinerungen();
    korrekte_Anzahl_der_Dreiecke();
}

```

```
133 void dreieck::null_Koordinaten_setzen()
134 {
135     x[0][0] = 0.0;
136     y[0][0] = 0.0;
137     x[0][1] = 0.0;
138     y[0][1] = 0.0;
139     x[0][2] = 0.0;
140     y[0][2] = 0.0;
141 }
142 
143 void dreieck::Punkte_setzen_1_Dreieck()
144 {
145     null_Koordinaten_setzen();
146     for(int i = 0;i < 3;i++)
147     {
148         Punkt_Dreieck.x = x[0][i];
149         Punkt_Dreieck.y = y[0][i];
150         Punkt_Dreieck.set_point();
151         p1[0][i] = Punkt_Dreieck.temp;
152     }
153 }
154 
155 void dreieck::eins_Koordinaten_setzen()
156 {
157     x[1][0] = 0.0;
158     y[1][0] = 0.0;
159     x[1][1] = 0.0;
160     y[1][1] = 0.0;
161     x[1][2] = 0.0;
162     y[1][2] = 0.0;
163 }
164 
165 void dreieck::Punkte_setzen_2_Dreiecke()
166 {
167     Punkte_setzen_1_Dreieck();
168     eins_Koordinaten_setzen();
169     for(int i = 0;i < 3;i++)
170     {
171         Punkt_Dreieck.x = x[1][i];
172         Punkt_Dreieck.y = y[1][i];
173         Punkt_Dreieck.set_point();
174         p1[1][i] = Punkt_Dreieck.temp;
175     }
176 }
177 
178 void dreieck::zwei_Koordinaten_setzen()
179 {
180     x[2][0] = 0.0;
181     y[2][0] = 0.0;
182     x[2][1] = 0.0;
183     y[2][1] = 0.0;
184     x[2][2] = 0.0;
185     y[2][2] = 0.0;
186 }
187 
188 void dreieck::Punkte_setzen_3_Dreiecke()
189 {
190     Punkte_setzen_2_Dreiecke();
191     zwei_Koordinaten_setzen();
192     for(int i = 0;i < 3;i++)
193     {
194         Punkt_Dreieck.x = x[2][i];
195         Punkt_Dreieck.y = y[2][i];
196         Punkt_Dreieck.set_point();
197         p1[2][i] = Punkt_Dreieck.temp;
198     }
```

```
}

void dreieck::drei_Koordinaten_setzen()
{
    x[3][0] = 0.0;
    y[3][0] = 0.0;
    x[3][1] = 0.0;
    y[3][1] = 0.0;
    x[3][2] = 0.0;
    y[3][2] = 0.0;
}

void dreieck::Punkte_setzen_4_Dreiecke()
{
    Punkte_setzen_3_Dreiecke();
    drei_Koordinaten_setzen();
    for(int i = 0;i < 3;i++)
    {
        Punkt_Dreieck.x = x[3][i];
        Punkt_Dreieck.y = y[3][i];
        Punkt_Dreieck.set_point();
        p1[3][i] = Punkt_Dreieck.temp;
    }
}

void dreieck::Dreiecke_ueberpruefen()
{
    for(int i = 0;i < Anzahl_der_Dreiecke;i++)
        Dreiecke_pruefen[i] = 0;
    for(int i = 0;i < Anzahl_der_Dreiecke;i++)
    {
        p2 = p1[i][0];
        p3 = p1[i][1];
        p4 = p1[i][2];

        Punkt_Dreieck.temp = p2;
        Punkt_Dreieck.temp1 = p3;
        Punkt_Dreieck.Punkte_ueberpruefen();
        p2_p3_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;

        Punkt_Dreieck.temp = p3;
        Punkt_Dreieck.temp1 = p4;
        Punkt_Dreieck.Punkte_ueberpruefen();
        p3_p4_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;

        Punkt_Dreieck.temp = p4;
        Punkt_Dreieck.temp1 = p2;
        Punkt_Dreieck.Punkte_ueberpruefen();
        p4_p2_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;

        if((p2_p3_Punkte_pruefen == 0) || (p3_p4_Punkte_pruefen == 0) || (p4_p2_Punkte_pruefen == 0))
            Dreiecke_pruefen[i] = 1;
    }
    for(int i = 0;i < Anzahl_der_Dreiecke - 1;i++)
    {
        if(Dreiecke_pruefen[i] == 0)
        {
            p2 = p1[i][0];
            p3 = p1[i][1];
            p4 = p1[i][2];

            for(int j = i + 1;j < Anzahl_der_Dreiecke;j++)
            {
                if(Dreiecke_pruefen[j] == 0)
                {
                    p5 = p1[j][0];
```

```
265                     p6 = p1[j][1];
266                     p7 = p1[j][2];
267
268                     Punkt_Dreieck.temp = p2;
269                     Punkt_Dreieck.temp1 = p5;
270                     Punkt_Dreieck.Punkte_ueberpruefen();
271                     p2_p5_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
272
273                     Punkt_Dreieck.temp = p3;
274                     Punkt_Dreieck.temp1 = p6;
275                     Punkt_Dreieck.Punkte_ueberpruefen();
276                     p3_p6_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
277
278                     Punkt_Dreieck.temp = p4;
279                     Punkt_Dreieck.temp1 = p7;
280                     Punkt_Dreieck.Punkte_ueberpruefen();
281                     p4_p7_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
282
283                     Punkt_Dreieck.temp = p3;
284                     Punkt_Dreieck.temp1 = p7;
285                     Punkt_Dreieck.Punkte_ueberpruefen();
286                     p3_p7_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
287
288                     Punkt_Dreieck.temp = p4;
289                     Punkt_Dreieck.temp1 = p6;
290                     Punkt_Dreieck.Punkte_ueberpruefen();
291                     p4_p6_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
292
293                     Punkt_Dreieck.temp = p2;
294                     Punkt_Dreieck.temp1 = p6;
295                     Punkt_Dreieck.Punkte_ueberpruefen();
296                     p2_p6_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
297
298                     Punkt_Dreieck.temp = p3;
299                     Punkt_Dreieck.temp1 = p5;
300                     Punkt_Dreieck.Punkte_ueberpruefen();
301                     p3_p5_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
302
303                     Punkt_Dreieck.temp = p4;
304                     Punkt_Dreieck.temp1 = p5;
305                     Punkt_Dreieck.Punkte_ueberpruefen();
306                     p4_p5_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
307
308                     Punkt_Dreieck.temp = p2;
309                     Punkt_Dreieck.temp1 = p7;
310                     Punkt_Dreieck.Punkte_ueberpruefen();
311                     p2_p7_Punkte_pruefen = Punkt_Dreieck.Punkte_pruefen;
312
313                     if(((p2_p5_Punkte_pruefen == 0) && (p3_p6_Punkte_pruefen == 0) && (p4_p7_Punkte_pruefen
== 0)) ||
314                        ((p2_p5_Punkte_pruefen == 0) && (p3_p7_Punkte_pruefen == 0) && (p4_p6_Punkte_pruefen
== 0)) ||
315                        ((p2_p6_Punkte_pruefen == 0) && (p3_p5_Punkte_pruefen == 0) && (p4_p7_Punkte_pruefen
== 0)) ||
316                        ((p2_p6_Punkte_pruefen == 0) && (p3_p7_Punkte_pruefen == 0) && (p4_p5_Punkte_pruefen
== 0)) ||
317                        ((p2_p7_Punkte_pruefen == 0) && (p3_p5_Punkte_pruefen == 0) && (p4_p6_Punkte_pruefen
== 0)) ||
318                        ((p2_p7_Punkte_pruefen == 0) && (p3_p6_Punkte_pruefen == 0) && (p4_p5_Punkte_pruefen
== 0)))
319                         Dreiecke_pruefen[j] = 1;
320                 }
321             }
322         }
323     }
324 }
```

```
325
326 void dreieck::korrekte_Punkte()
327 {
328     switch(Anzahl_der_Dreiecke)
329     {
330         case 1:Punkte_setzen_1_Dreieck();
331                break;
332         case 2:Punkte_setzen_2_Dreiecke();
333                break;
334         case 3:Punkte_setzen_3_Dreiecke();
335                break;
336         case 4:Punkte_setzen_4_Dreiecke();
337                break;
338     }
339     Dreiecke_ueberpruefen();
340     for(int i = 0;i < Anzahl_der_Dreiecke;i++)
341     {
342         if(Dreiecke_pruefen[i] == 1)
343         {
344             for(int j = 0;j < 3;j++)
345                 p1[i][j] = f_p[i][j];
346         }
347     }
348 }
349
350 triangle_linked *dreieck::create_triangle(const point p0,const point p1,const point p2)
351 {
352     static triangle_linked *first_triangle = nil;
353     triangle_linked *nf = (triangle_linked*)malloc(sizeof(triangle_linked));
354
355     nf->pt[0] = p0;
356     nf->pt[1] = p1;
357     nf->pt[2] = p2;
358
359     nf->next = first_triangle;
360     first_triangle = nf;
361
362     return(first_triangle);
363 }
364
365 void dreieck::p5_setzen()
366 {
367     Punkt_Dreieck.x = (Punkt_Dreieck.temp.x + Punkt_Dreieck.temp1.x) / 2;
368     Punkt_Dreieck.y = (Punkt_Dreieck.temp.y + Punkt_Dreieck.temp1.y) / 2;
369     Punkt_Dreieck.set_point();
370     p5 = Punkt_Dreieck.temp;
371 }
372
373 void dreieck::p6_setzen()
374 {
375     Punkt_Dreieck.x = (Punkt_Dreieck.temp.x + Punkt_Dreieck.temp1.x) / 2;
376     Punkt_Dreieck.y = (Punkt_Dreieck.temp.y + Punkt_Dreieck.temp1.y) / 2;
377     Punkt_Dreieck.set_point();
378     p6 = Punkt_Dreieck.temp;
379 }
380
381 void dreieck::p7_setzen()
382 {
383     Punkt_Dreieck.x = (Punkt_Dreieck.temp.x + Punkt_Dreieck.temp1.x) / 2;
384     Punkt_Dreieck.y = (Punkt_Dreieck.temp.y + Punkt_Dreieck.temp1.y) / 2;
385     Punkt_Dreieck.set_point();
386     p7 = Punkt_Dreieck.temp;
387 }
388
389 void dreieck::ps_Dreieck_setzen()
390 {
```

```
391     printf("Zaehler = %d\n",Zaehler);
392     printf("temp = (%f,%f)\n",Punkt_Dreieck.temp.x,Punkt_Dreieck.temp.y);
393     printf("temp1 = (%f,%f)\n",Punkt_Dreieck.temp1.x,Punkt_Dreieck.temp1.y);
394     printf("temp2 = (%f,%f)\n",Punkt_Dreieck.temp2.x,Punkt_Dreieck.temp2.y);
395     ps[Zaehler][0] = Punkt_Dreieck.temp;
396     ps[Zaehler][1] = Punkt_Dreieck.temp1;
397     ps[Zaehler][2] = Punkt_Dreieck.temp2;
398     printf("ps = (%f,%f)\n",ps[Zaehler][0].x,ps[Zaehler][0].y);
399 }
400
401 void dreieck::Punkte_setzen()
402 {
403     printf("Position = %d\n",Position);
404     for(int l = 0;l < 4;l++)
405     {
406         for(int m = 0;m < 3;m++)
407             p1[Position][m] = ps[l][m];
408         Position++;
409     }
410 }
411
412 void dreieck::refine_element()
413 {
414     p2 = p8[Zaehler][0];
415     p3 = p8[Zaehler][1];
416     p4 = p8[Zaehler][2];
417
418     Punkt_Dreieck.temp = p2;
419     Punkt_Dreieck.temp1 = p3;
420     p5_setzen();
421     printf("p5 = (%f,%f)\n",p5.x,p5.y);
422
423     Punkt_Dreieck.temp = p3;
424     Punkt_Dreieck.temp1 = p4;
425     p6_setzen();
426     printf("p6 = (%f,%f)\n",p6.x,p6.y);
427
428     Punkt_Dreieck.temp = p4;
429     Punkt_Dreieck.temp1 = p2;
430     p7_setzen();
431     printf("p7 = (%f,%f)\n",p7.x,p7.y);
432
433     for(int k = 1;k <= 4;k++)
434     {
435         printf("k = %d\n",k);
436         switch(k)
437         {
438             case 1:Punkt_Dreieck.temp = p5;
439                    Punkt_Dreieck.temp1 = p6;
440                    Punkt_Dreieck.temp2 = p7;
441                    break;
442             case 2:Punkt_Dreieck.temp = p2;
443                    Punkt_Dreieck.temp1 = p5;
444                    Punkt_Dreieck.temp2 = p7;
445                    break;
446             case 3:Punkt_Dreieck.temp = p5;
447                    Punkt_Dreieck.temp1 = p3;
448                    Punkt_Dreieck.temp2 = p6;
449                    break;
450             case 4:Punkt_Dreieck.temp = p6;
451                    Punkt_Dreieck.temp1 = p4;
452                    Punkt_Dreieck.temp2 = p7;
453                    break;
454         }
455         printf("temp = (%f,%f)\n",Punkt_Dreieck.temp.x,Punkt_Dreieck.temp.y);
456         printf("temp1 = (%f,%f)\n",Punkt_Dreieck.temp1.x,Punkt_Dreieck.temp1.y);
```

```
457         printf("temp2 = (%f,%f)\n",Punkt_Dreieck.temp2.x,Punkt_Dreieck.temp2.y);
458         Zaehler = k - 1;
459         ps_Dreieck_setzen();
460     }
461     Punkte_setzen();
462 }
463
464 triangle_linked *dreieck::einhaengen(triangle_linked *original)
465 {
466     triangle_linked *y = original,*neu = nil;
467
468     neu = create_triangle(p1[Zaehler][0],p1[Zaehler][1],p1[Zaehler][2]);
469     neu->next = y;
470     y = neu;
471
472     return(neu);
473 }
474
475 triangle_linked *dreieck::refine(triangle_linked *original)
476 {
477     triangle_linked *y = original,*refine_d = nil;
478     Anzahl = Anzahl_der_Dreiecke;
479     Position = Anzahl;
480     printf("Anzahl = %d\n",Anzahl);
481     Anzahl_der_Dreiecke = 5 * Anzahl_der_Dreiecke;//Anzahl_der_Dreiecke = Anzahl_der_Dreiecke + 4 *
Anzahl_der_Dreiecke = 5 * Anzahl_der_Dreiecke;
482
483     for(int i = 0;i < Anzahl;i++)
484     {
485         for(int j = 0; j < 3;j++)
486         {
487             p1[i][j] = y->pt[j];
488             p8[i][j] = y->pt[j];
489         }
490         y = y->next;
491     }
492     for(int i = 0;i < Anzahl;i++)
493     {
494         Zaehler = i;
495         refine_element();
496     }
497     Dreiecke_ueberpruefen();
498
499     Anzahl = Anzahl_der_Dreiecke;
500     Anzahl_der_Dreiecke = 0;
501     for(int i = 0;i < Anzahl;i++)
502     {
503         if(Dreiecke_pruefen[i] == 0)
504         {
505             Zaehler = i;
506             refine_d = einhaengen(refine_d);
507             Anzahl_der_Dreiecke++;
508         }
509     }
510
511     return(refine_d);
512 }
513
514 void dreieck::print_all_triangles(triangle_linked *first,const char *Name_der_Textdatei)
515 {
516     for(;first;first = first->next)
517     {
518         sprintf(Text,"(%f %f),(%f %f),(%f %f)%s",first->pt[0].x,first->pt[0].y,first->pt[1].x,first->pt[1].
y,first->pt[2].x,first->pt[2].y,Eingabe_Dreieck.Ausgabe_Eingabe.Definition_Ausgabe.Schraegstrichn);
519         Eingabe_Dreieck.Ausgabe_Eingabe.ausgabe_Text(Text);
520         Eingabe_Dreieck.Ausgabe_Eingabe.speichern_in_Textdatei(Name_der_Textdatei,Text);
```

```
521       }
522   }
```