Kocaeli Üniversitesi
Bilgisayar Mühendisliği Bölümü
PROGRAMLAMA LAB. I- I. Proje
Minumum Çevreleyen Çember

MERTCAN KUŞCU 190202082@kocaeli.edu.tr
SİNAN TOPAL 190202035@kocaeli.edu.tr

Projenin Özeti

Programlama Laboratuvar I Projesi olarak bizden "Minimum Çevreleyen Çember" adındaki bir çalışmayı yapmamız beklenmektedir.

Biz proje için C Programlama Dilini ve CodeBlocks geliştirme ortamını seçtik.

Projede biz C Programlama Dilinde bulunan Graphics.h ve Math.h kütüphanelerini kullandık. Graphics.h kütüphanesi sayesinde arayüz tasarımını yaptık. Math.h kütüphanesi sayesinde ise çeşitli matematik işlemlerini yaptırdık.

Projemizde ilk olarak ekrana koordinat sistemini çizdirdik. Sonra belirttiğimiz dosyaya girilen n adet noktayı bu koordinat sisteminin üzerinde belirttik. Daha sonra Math.h kütüphanesi sayesinde matematiksel işlemlerle koordinat sisteminin üzerinde b-spline eğrisini çizdirdik. Daha sonra dosyadan alınan n adet noktaları içinde veya üstünde kapsayacak minimum çemberin merkezinin koordinatlarını, en uzak iki noktayı hesaplayarak bulduk. Çizilecek çemberin merkezine en uzak olan noktayı bulduktan sonra, merkezin bu noktaya olan uzunluğunu yarıçap olarak ekranımıza yansıttık. Elde ettiğimiz merkez koordinatları ve yarıçap ölçüsüne uygun olan çemberi koordinat sisteminin üzerinde gösterdik. En son olarak da merkezi ve yarıçapı çemberin üzerine yansıttık.

Giriş

Proje için C programlama dili ve CodeBlocks geliştirme ortamını kullandık.

C Programlama dili, yüksek seviyeli ve genel maksatlı olarak kullanılan bir yazılım dilidir. Günümüzde dahi C, hala en sık tercih edilen programlama dillerinden biri olarak kullanılmaktadır.

CodeBlocks özellikle C ve C++ gibi programlama dillerinde kodlama geliştirme yapmamıza imkân sağlayan, açık kaynak kodlu ve cross-platform bir IDE'dir.

Projemizde söylenen problem Minimum Çevreleyen Çember problemi olarak adlandırılmaktadır.

Projede bizden istenilen kullanıcı tarafından tamsayı koordinatlı 2 boyutlu bir düzlemde N nokta verildiğinde tüm noktaları içeren minimum çevreleyen yarıçaplı daireyi çizdirmemiz, verilen N noktanın en yakınından geçen eğriyi çizdirmemiz ve çizdirmiş olduğumuz dairenin ise yarıçapını ve merkezini hesaplamamız istenmektedir.

Yöntem

Projemizde yazdığımız kodu matematiksel işlemlerimizle paralel olarak götürmeye çalıştık. İlk olarak kullanıcının değerleri girebileceği bir dosya açtık, while döngüsü sayesinde programımızın değerleri bu dosyadan okumasını sağladık ve bu nokta değerlerini bir diziye atadık. Dosyadan okunan bu N adet nokta değerlerini for döngüsü sayesinde ekrana yazdırdık. Graphics.h kütüphanesi sayesinde ekranımıza düzgün bir koordinat sistemi çizdirdik. Dosyadan okunan N adet nokta değerlerini çizdirdiğimiz koordinat sisteminin üzerine for döngüsü sayesinde yansıttık. Çizdireceğimiz çemberin merkezini bulmak için dosyadan okunan en uzak iki noktayı for döngüsü ve Math.h kütüphanesi sayesinde bulduk ve bu iki noktanın ortasını merkezimiz olarak kabul ettik. Ardından bulduğumuz merkezin koordinatlarını ekrana yazdırdık. Merkeze en uzak noktayı yine aynı yöntemlerle hesaplayarak bulduk ve bu uzunluğu yarıçap olarak kabul ettik. Ardından bulduğumuz yarıçapı ekrana yazdırdık. Hesapladığımız merkez koordinatlarına ve yarıçap uzunluğuna göre çemberimizi koordinat sisteminin üzerine yansıttık. Daha sonra çeşitli kaynaklardan yardım alarak b-spline eğrisinin matematiksel işlemlerini öğrendik ve bunu kodumuza yansıttık. Ardından bunun sayesinde koordinat sistemimizde noktalara uygun şekilde olacak b-spline eğrisini çizdirmiş olduk. En sonda ise uygun değerlerle ekranda çemberin üzerine merkezi ve yarıçapı yansıttık.

B-Spline Matematiksel Yöntemleri

N, 1'den büyük bir doğal sayı olmak üzere; 1'den N'ye kadar olan doğal sayıların çarpımına N'nin ***faktöriyeli*** *denir. Örneğin 3!=3.2.1=6' dır. 0 faktöriyel ise her zaman 1 olarak kabul edilir.*

R ve N bir doğal sayı olsun. R ≤ N olmak üzere; N elemanlı bir kümenin R elemanlı her alt kümesine, bu N elemanın R'li bir ***kombinasyonu*** *denir. Örneğin*

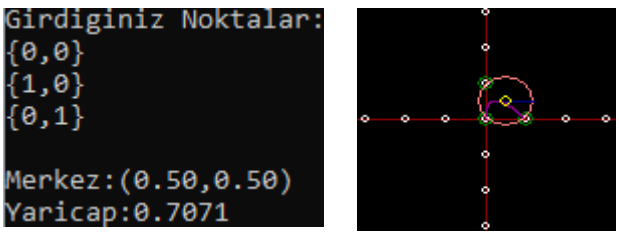
$\binom{9}{2} = \frac{9.8}{2.1} = 36$ *bulunur. Ayrıca kombinasyon(n,r)' yi şu şekilde de bulabiliriz;*

*(faktöriyel(n)/(faktöriyel(n-r)*faktöriyel(r)))*

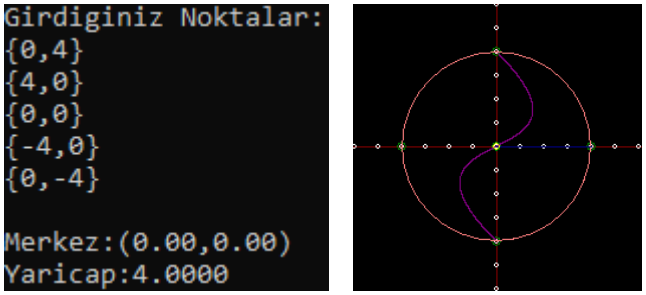
Bu iki yöntemi kullanarak da ;

```
for( t=0;t<=1;t=t+0.0001){

int x=0; int  y=0;

for(int i=0;i<=h;i++){

int ax[i]=noktalar[i][0]; int ay[i]=noktalar[i][1];

      x=x+(kombinasyon(h,i)*pow(1-t,h-i)*pow(t,i)*ax[i]);

      y=y+(kombinasyon(h,i)*pow(1-t,h-i)*pow(t,i)*ay[i]);

/*pow Math.h kütüphanesinde üslü alma işlemi için kullanılır. Örneğin pow(2,3)=2.2.2=8' dir.*/

    }

putpixel(300+x*20,240-y*20,5);

/*putpixel komutu ise içindeki değerlere göre piksel çizer. Yaptığımız döngü sayesinde bu bir eğri gibi gözükür.*/

  }
```

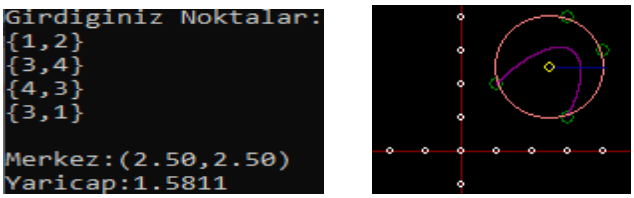
Deneysel Sonuçlar



Yukarıdaki noktaları girdiğimizde daire 0.7071 yarıçaplı ve merkez (0.50,0.50) ile çizildiğinde belirtilen tüm noktaların dairenin içinde veya üzerinde olduğu görülmektedir.



Yukarıdaki noktaları girdiğimizde daire 4.0000 yarıçaplı ve merkez (0.00,0.00) ile çizildiğinde belirtilen tüm noktaların dairenin içinde veya üzerinde olduğu görülmektedir.



Yukarıdaki noktaları girdiğimizde daire 1.5811 yarıçaplı ve merkez (2.50,2.50) ile çizildiğinde belirtilen tüm noktaların dairenin içinde veya üzerinde olduğu görülmektedir.

Sonuç

Projemizin, girdiğimiz bütün değerleri sorunsuz ve doğru şekilde çalıştığını görmüş olduk.

Bu proje sayesinde C dilinde çeşitli kütüphaneleri kullanmayı ve işlevlerini öğrenmiş olduk.

Graphics.h kütüphanesiyle ekrana ne tür şeyler çizdirebileceğimizi görmüş olduk. Math.h kütüphanesiyle de çeşitli matematiksel işlemleri nasıl kolaylıkla koda yansıtabileceğimizi öğrendik.

B-spline' ın ne olduğunu gördük, matematiksel işlemlerini öğrendik. Ekrana çizdirmeyi kodumuz sayesinde başarıyla sağladık.

Algoritma Açıklaması

1 *fct fonksiyonunu oluştur.*

2 *n ile integer değer tanımla.*

3 *n=0 ise sonuç 1'dir.*

4 *n=0 değil ise n*fct(n-1) işlemini uygula.*

5 *cmb fonksiyonunu oluştur.*

6 *n ve r ile integer değerler tanımla.*

7 *(fct(n)/(fct(n-r)*fct(r))) işlemini uygula.*

8 *main fonksiyonuna başla.*

9 *proje.txt dosyasından oku.*

10 *Dosyadan girilen nokta değerlerini al.*

11 *Dosyadan girilen değerleri ekrana yazdır.*

12 *En uzak iki noktayı hesapla.*

13 Çizilecek çemberin merkezini hesapla.

14 Merkezi ekrana yazdır.

15 Çizilecek çemberin yarıçapını hesapla.

16 Yarıçapı ekrana yaz.

17 Ekrana koordinat sistemini çiz.

18 Dosyadan alınan noktaları koordinat sisteminin üstünde göster.

19 Dosyadan alınan noktaların en yakınından geçen eğriyi ekrana fonksiyonlarla hesaplayıp çizdir.

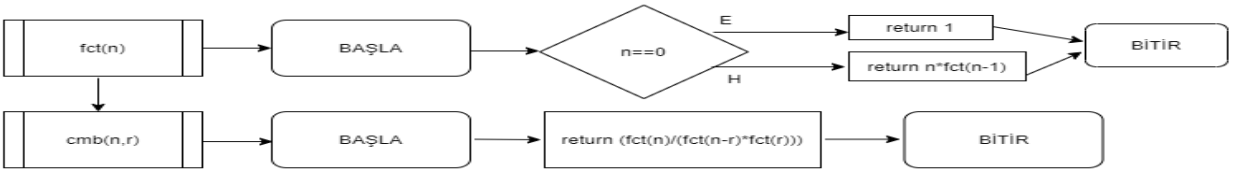
20 Hesaplanan merkez ve yarıçapa göre çemberi ekrana çizdir.

21 Çemberin üstünde merkezi ve yarıçapı göster.

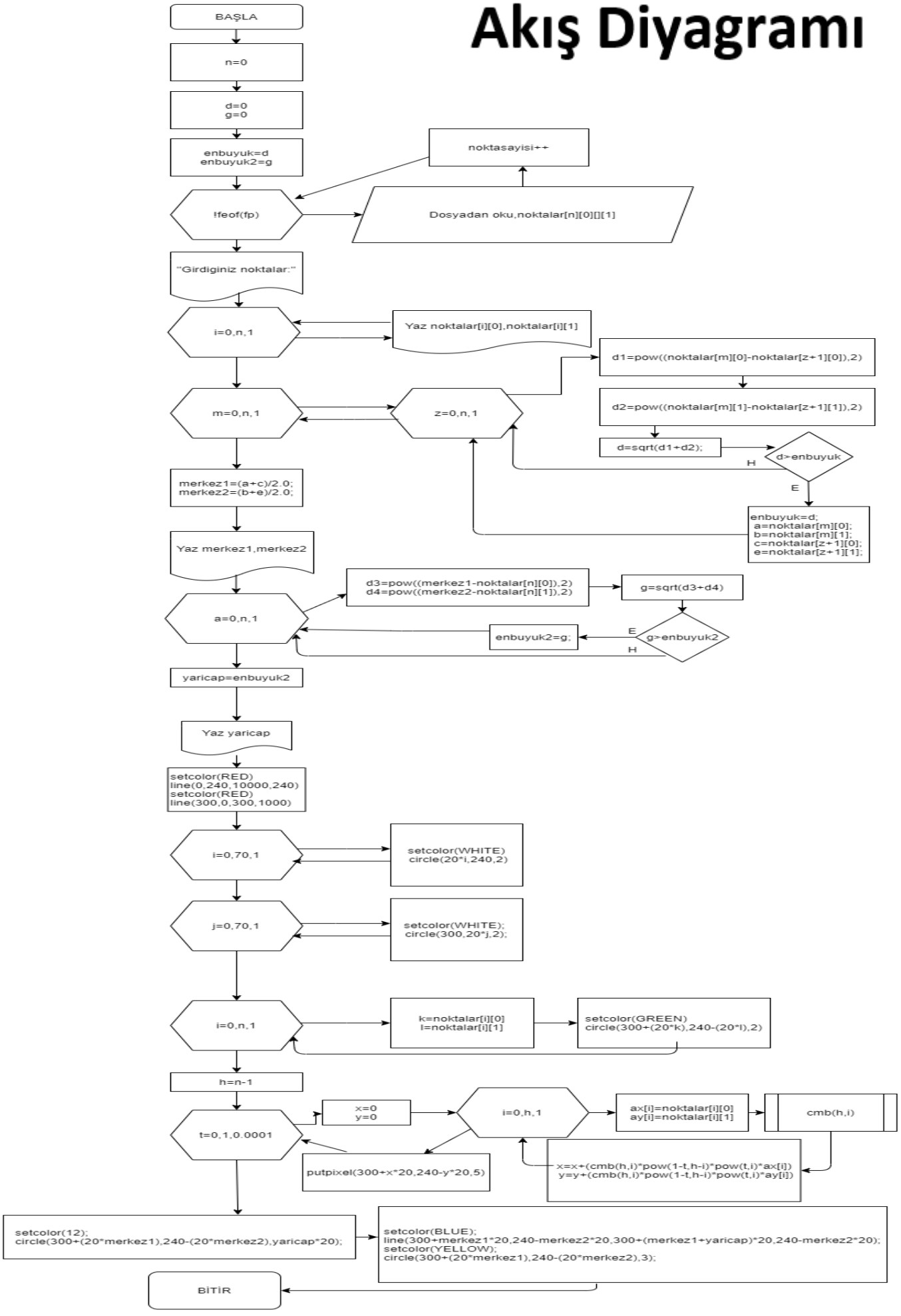
22 Bitir.

Algoritmanın Kaba Kodu

```
1  Fonksiyon fct(n);
2  Başla
3  if(n==0) return 1;
4  else  return n*fct(n-1);
5  Bitir
6  Fonksiyon cmb(n,r)
7  Başla
8  return (fct(n)/(fct(n-r)*fct(r)));
9  Bitir
10 Başla
11 Oku proje.txt;
12 0'dan n'ye kadar her bir i değeri için
13 Yaz noktalar[i][0],noktalar[i][1];
14 Döngüyü kır
15 0'dan n'ye kadar her bir m değeri için
16 0'dan n'ye kadar her bir z değeri için
17 d1=pow((noktalar[m][0]-noktalar[z+1][0]),2);
18 d2=pow((noktalar[m][1]-noktalar[z+1][1]),2);
19 d=sqrt(d1+d2);
20 if(d>enbuyuk) enbuyuk=d;
21 a=noktalar[m][0];  b=noktalar[m][1];
22 c=noktalar[z+1][0];   e=noktalar[z+1][1];
23 z döngüsünü kır
24 m döngüsünü kır
25 merkez1=(a+c)/2.0; merkez2=(b+e)/2.0;
26 Yaz merkez1,merkez2;
27 0'dan n'ye kadar her bir a değeri için
28 d3=pow((merkez1-noktalar[a][0]),2);
29 d4=pow((merkez2-noktalar[a][1]),2);
30 g=sqrt(d3+d4);
31 if(g>enbuyuk2)  enbuyuk2=g;
32 döngüyü kır
33 yaricap=enbuyuk2;
34 Yaz yaricap;
35 Oluştur initgraph(&sur,&gmode,"");
36 line(0,240,10000,240);
37 line(300,0,300,1000);
38 0'dan 70'e kadar her bir i değeri için
39 circle(20*i,240,2);
40 Döngüyü kır
41 0'dan 70'e kadar her bir j değeri için
42 circle(300,20*j,2);
43 Döngüyü kır
44 0'dan n'ye kadar her bir i değeri için;
45 k=noktalar[i][0];    l=noktalar[i][1];
46 circle(300+(20*k),240-(20*l),2);
47 Döngüyü kır
48 h=n-1;
49 0'dan 1'e kadar her t+0.0001 değeri için
50 0'dan h ye kadar her bir i değeri için
51 ax[i]=noktalar[i][0]; ay[i]=noktalar[i][1];
52 x=x+(cmb(h,i)*pow(1-t,h-i)*pow(t,i)*ax[i]);
53 y=y+(cmb(h,i)*pow(1-t,h-i)*pow(t,i)*ay[i]);
54 i döngüsünü kır
55 putpixel(300+x*20,240-y*20,5);
56 t döngüsünü kır
57 circle(300+(20*merkez1),240-(20*merkez2),yaricap*20);
58 line(300+merkez1*20,240-
merkez2*20,300+(merkez1+yaricap)*20,240-merkez2*20);
59 circle(300+(20*merkez1),240-(20*merkez2),3);
60 Bitir
```



Akış Diyagramı



Zaman Karmaşıklığı Hesabı (Big O)

Kodumuzda zaman karmaşıklığı hesabını yaparken döngüsüz normal satırları "1" olarak aldık. Döngülü satırlarda ise içindeki bilgilere göre zaman karmaşıklığı hesabını yaptık. Kodumuzun en baştan en sona ***sırayla*** *zaman karmaşıklığı hesabı şu şekildedir:*

// n elemanli bir islem icin n kez sorgu yapar.

//rekursifligin bittigi durumda calisan son cagri oldugu icin tek bir kez calişir ve islem biter: 1

// n-1 kez çalışır

//Bu fonksiyon toplam zaman karmasikligi= n+1+n-1= 2n'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= n'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= n'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= n^2'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= n'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 70'tir.

//Zaman Karmasikligi= 70'tir.

//Zaman Karmasikligi= n'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= n-1'dir.

*//Zaman Karmasikligi= 2*10.000*(n-1)=20.000*(n-1)'dir.*

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

//Zaman Karmasikligi= 1'dir.

*/*Zaman Karmaşıklığı= n^2+20.006n-19826*

BıgO Notasyonu O(n^2) olarak bulunur./*

Kaynakça

https://youtu.be/GM0kni4jdPY?t=210

Palme Yayınevi C ile Programlama Öğreniyorum

https://www.cs.mcgill.ca/~cs507/projects/1998/jacob/problem.html

https://www.geeksforgeeks.org/minimum-enclosing-circle-set-1/

https://app.diagrams.net/

https://www.youtube.com/watch?v=hqONlh0YTeA

https://technogezgin.com/pseudocode-nedir-sozde-kod/

http://bilgioloji.com/pages/yazilim/kod/program/algoritma/giris/sozde-kod-pseudocode-nedir/

http://gulden.kokturk.com/writingrules.pdf

http://www.ttbilgin.com/2018-2019-guz/bilmuh-giris/hafta4/algoritmalar-ve-akis-diyagramlari.pdf

https://www.elektrikport.com/teknik-kutuphane/yalanci-kod-nedir/14925#ad-image-0

https://acikders.ankara.edu.tr/pluginfile.php/19157/mod_resource/content/1/fzm205_2.pdf

https://web.cs.hacettepe.edu.tr/~maydos/Docs/c/flowcharts.pdf

https://e-bergi.com/y/algoritmalarda-karmasiklik/

MERTCAN KUŞCU 190202082@kocaeli.edu.tr

SİNAN TOPAL 190202035@kocaeli.edu.tr