T.C.

FIRAT ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

OTOMATİK DERS PROGRAMI ÇIKARMA

Hazırlayanlar

İbrahim Oktay BARUT

Levent SOLAKOĞLU

Tez Yöneticisi

Yrd.Doç. Dr. Hasan Hüseyin BALIK

LİSANS TEZİ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ BÖLÜMÜ

ELAZIĞ, 2005

T.C.

FIRAT ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

OTOMATİK DERS PROGRAMI ÇIKARMA

Hazırlayanlar

İbrahim Oktay BARUT

Levent SOLAKOĞLU

LİSANS TEZİ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ BÖLÜMÜ

Bu tez, ... tarihinde aşağıda belirtilen jüri tarafından oybirliği /oyçokluğu ile başarılı / başarısız olarak değerlendirilmiştir.

Danışman: Yrd. Doç. Dr. Hasan Hüseyin BALIK

Üye:

Üye:

TEŞEKKÜR

Bu lisans tezi çalışmasının hazırlanmasında hiçbir zaman desteğini esirgemeyen kıymetli hocam Yrd. Doç. Dr. Hasan Hüseyin BALIK 'a , çalışmanın içeriğindeki bir çok konuda bilgilerini benimle paylaşan sayın Hatice İNNECİ, Mehmet Akif BARIŞ, Gürkan ÇİÇEK ve Ömer ASLAN 'a teşekkür ederim.

İbrahim Oktay BARUT

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

ÖNSÖZ

Bilgisayar ve bilgisayar uygulamaları günlük hayatın her alanında önemli unsurlar olmaya başlayınca, daha önceden bir veya birden çok kişinin uzun zaman alan ve oldukça karmaşık işlemler sonucunda gerçekleştirdikleri bir çok işlem de, bilgisayar ortamında çözülmeye başlamıştır. Bu işlemler için genellikle bir paket program hazırlanarak kullanıcılara sunulmuş, kullanıcılar bu programları kullanarak hem çok hızlı hem de hatasız olarak işlemlerini gerçekleştirmiştir.

Okuldaki ders programının ayarlanması çok önemli bir konudur. Tahmin edileceği gibi otomatik ders programında bazı zorluklar, çakışmalar, kısıtlamalar, kriterler vardır ve bunlar dikkate alınarak yapılacak programlar oldukça karmaşıktır.

Problemlerinin çözümünde çeşitli yaklaşımlar kullanılmıştır. Kümeleme, Genetik Algorithm, Ard Arda Sıralama, Mantıksal Programlama, Yerel Arama ve Graph Coloring bunların başlıcalarıdır. Projemizde bu yaklaşımlardan biri olan Graph Coloring kullanılmıştır.

Üniversitelerdeki sınav programlarının hazırlanması da bu karmaşık ve hata oranı yüksek konuların başında gelmektedir. Bir çok üniversitede hala ciddi problemler çıkmakta ve her dönem sonunda bu sıkıntılar tekrarlanmaktadır. Bunun sonucu olarak sınavları öğrencinin aldığı derslere göre, çakıştırmadan dağıtacak bir programa ihtiyaç duyulmaktadır. Bu uygulama için Graph Coloring algoritması en kullanışlı algoritmalardan biridir.

Bu çalışmada, Graph Coloring algoritmalarından biri olan Welch ve Powel Algoritması kullanılmıştır. Bu algoritma genel olarak düğümlerin derecelerine dayanmaktadır. Bu yüzden dağıtımı yapılacak derslerin bir grafa aktarılması ve bu dersler arasındaki komşulukların belirlenmesi gerekmektedir.

Bu çalışmada dersler düğüm olarak kabul edilmiş, tüm dersleri ve bunların arasındaki komşulukları gösteren graf oluşturulmuştur. Düğümler arasındaki komşuluklar, bir öğrenci tarafından alınan derslere bakılarak kurulmuştur. Aynı öğrencinin aldığı dersler komşu ilan edilmiştir. Düğüm derecelerine bakılarak Welch ve Powel algoritması tüm düğümler renklendirilinceye kadar uygulanmıştır. Renklendirme yapılırken kullanılan renk sayısı, ders yerleştirme yapılırken ihtiyaç duyulan saat sayısını vermektedir.

Projede C++ Builder 6 kullanılmış olup Paradox veritabanı kullanılarak hazırlanmıştır.

ÖZET

Üniversitelerde yılda iki kez karşılaşılan dönem başlarındaki ders programlarının hazırlanması ve dönem sonlarındaki sınav programlarının yapılması oldukça zor ve karmaşıktır. Öğrencileri ve öğretim elemanlarını mağdur etmeyecek şekilde ders ve sınav programı hazırlama her üniversitede problem olmaktadır. Genellikle elle hazırlanmış ve bütün gayretlere rağmen sorunlar tam olarak giderilememiştir. Bu proje böyle bir sorunu ortadan kaldırmak için hazırlanmıştır.

Projede Graph Coloring Algorithm incelenecek ve projeye nasıl uygulandığı gösterilecektir.

1. GRAPH COLORING ALGORITHM:

Graf renklendirme, graf üzerinde birbirine komşu olan düğümlere farklı renk atama işlemidir. Amaç, en az sayıda renk kullanılarak tüm düğümlere komşularından farklı birer renk vermektir. Renklendirmede kullanılan renk sayısı kromatik sayı olarak adlandırılır.

Uygulamada, graf renklendirmenin kullanılacağı alanların başında, ilk akla gelen, harita üzerindeki bölgelerin renklendirilmesi olmasına karşın, graf renklendirme bilgisayar biliminde ve günlük yaşamdaki birçok problemin çözümünde kullanılan bir yaklaşımdır. Örneğin, sınırlı sayıda işlemcisi olan bir sistemde proseslerin işlemcileri kullanma zamanları ve sıralamasının belirlenmesinde, üniversitenin bir fakültesinde sınav saat ve günlerinin çakışmayacak bir şekilde yerleştirilmesinde çözüm sunar, bir de tabi ki bölge renklendirmede en az sayıda renk kullanılmasını sağlar.

Graf renklendirmede kullanılan algoritmalardan birisi Welch ve Powel'in önerdiği yöntemdir. Bu yöntem genel olarak düğümlerin derecelerine dayanmaktadır.

Welch ve Powel Algoritması: Bu algoritmanın davranışı adım adım aşağıdaki gibidir.

- Düğümler derecelerine göre büyükten küçüğe doğru sıralanır
- İlk renk birinci sıradaki düğüme atanır ve daha sonra aynı renk birbirlerine bitişik olmayacak biçimde diğer düğümlere verilir.
- Bir sonraki renge geçilir, bu renk sıradaki derecesi en yüksek olan düğüme atanır; ve sonra bu renk, daha önce renklendirilmemiş düğümlere birbirlerine bitişi olmayacak şekilde atanır. Üçüncü adım tüm düğümlere renk verilince sonlandırılır. [ÇÖLKESEN**,** R, Veri Yapıları ve Algoritmalar-2002]

2. PROGRAMIN GELİŞTİRİLMESİ

Programın geliştirilmesinde çeşitli verilerin tespiti gereklidir.Bunlar :

- Gün sayısı ve saat aralıkları: Hangi günler de ve hangi saat aralıklarına derslerin yerleştirileceği bilgisidir.
- Dersler: Veri tabanına hangi derslerin girileceği**,** bu derslerin kodu**,** adı, seviyesi, dersi alan öğrenci sayısıdır.
- Öğrenciler: Öğrencilerin numarası, adı, soyadı, sınıfı ve öğrencinin aldığı derslerdir.

Bu veriler doğrultusunda veritabanı tabloları oluşturulmuştur. Bize gerekli olan veritabanı tabloları:

-Ogrenci

-Ogrenciders

-Ders

dir.

Ders dağılımı yapılırken çeşitli kriterler vardır:

-Dersler homojen dağıtılmalıdır.

-Öğrencilerin derslerinin çakışmamasına özen gösterilmelidir.

3.PROJENİN HAZIRLANMASI :

Projeyi hazırlarken öncelikle veritabanı kullanmadan program içerisinden girilen sabit değerlerle graf renklendirme algoritmasının doğru çalışıp çalışmadığı test edildi. Bunun için program içerisinden aşağıdaki gibi beş öğrenci(Şekil-1) ve beş ders (Şekil-2) değer olarak girildi.

Table : :okhimba:ogrenci.DB

ogrenci	Ogr_no	Ogr_ad	Ogr_soyad	Ogr_sinif
1	1	Oktay	Barut	1,00
2	2	Melikþah	Özdemir	2,00
3	3	Fadlý	Tatar	3,00
4	4	Mehmet Levent	Aslan	4,00
5	5	Harun	Kanki	1,00

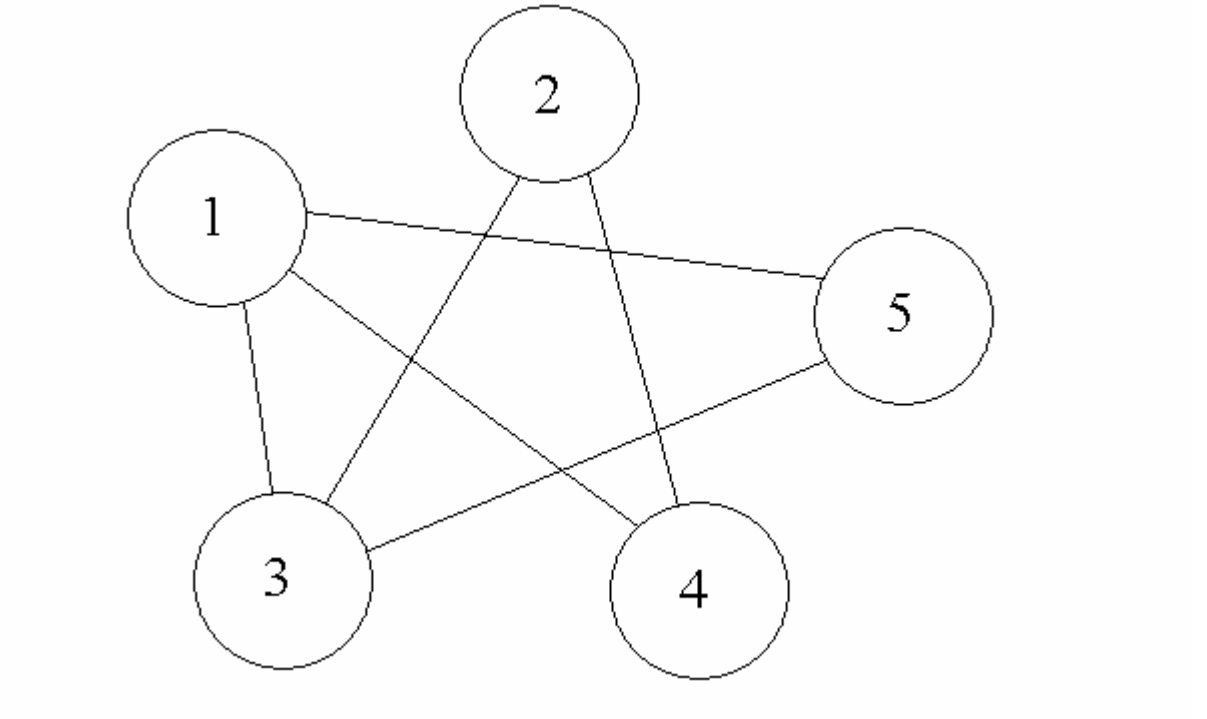
Şekil-1- Öğrenciler

Table : :okhimba:ders.DB

ders	Ders_kodu	Ders_adi	Hocasi	Sinif
1	1	Sayýsal Haberleþme	Hasan H. BALIK	1,00
2	2	Elektronik Tüm Devreler	Fikret ATA	2,00
3	3	Mikroiþlemciler	Melih C. ÝNCE	3,00
4	4	Otomatik Kontrol	Mustafa TÜRK	4,00
5	5	Ýç Tesisat Projesi	Muhsin T. GENCOÐLU	2,00

Şekil-2- Dersler

Veriler ile algoritmamız doğrultusunda komşuluklar çıkarılacak ve bu komşuluklara ilişkin graf elde edilecektir.Graf aşağıdaki gibi olacaktır(Şekil-3).Düğümler Ders Kodlarıdır.



Şekil-3- Komşuluklar

Yukarıdaki grafımızda komşuluklar aşikar olarak gözükmektedir.Ve Graf Coloring Algorithm gereği komşu olan hiçbir ders aynı saate konulamaz.Ve renklendirme bu koşul ile yapılabilir.

4.VERİTABANI VE TABLOLAR :

Şimdi veritabanı tablolarını ve fieldlerimizi tanıyalım :

Ogrenci tablosu :

Table : :okhimba:ogrenci.DB

ogrenci	Ogr_no	Ogr_ad	Ogr_soyad	Ogr_sinif
1	1	Oktay	Barut	1,00
2	2	Melikþah	Özdemir	2,00
3	3	Fadlý	Tatar	3,00
4	4	Mehmet Levent	Aslan	4,00
5	5	Harun	Kanki	1,00

Şekil-4- Öğrenci tablosu

Ders tablosu :

Table : :okhimba:ders.DB

ders	Ders_kodu	Ders_adi	Hocasi	Sinif
1	1	Sayýsal Haberleþme	Hasan H. BALIK	1,00
2	2	Elektronik Tüm Devreler	Fikret ATA	2,00
3	3	Mikroiþlemciler	Melih C. ÝNCE	3,00
4	4	Otomatik Kontrol	Mustafa TÜRK	4,00
5	5	Ýç Tesisat Projesi	Muhsin T. GENCOÐLU	2,00

Şekil-5- Ders tablosu

Ogrenciders tablosu :

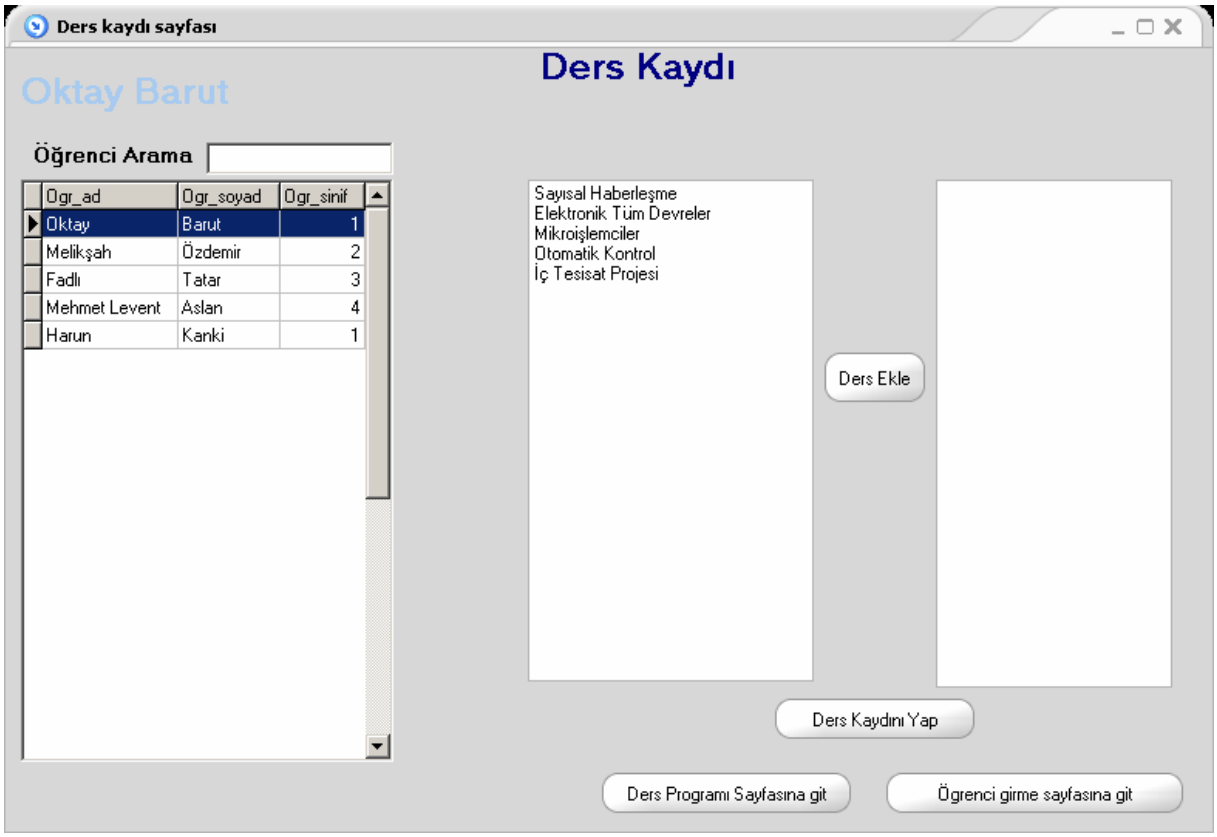
Table : :okhimba:ogrencid...

ogrenciders	Ogr_no	Ders_kodu
1	1	1
2	1	3
3	1	5
4	2	2
5	2	3
6	3	2
7	3	4
8	4	1
9	4	3
10	4	5
11	5	1
12	5	4

Şekil-6- Ogrenciders tablosu

5.OTOMATİK DERS YERLEŞTİRME YAZILIMI:

Otomatik ders yerleştirme yazılımın kullanılması ve kodlar bu kısımda anlatılacaktır.



Şekil-7- Ders kaydı sayfası

Yazılımı kurduktan sonra açılışta DBGrid1 kompanentinde öğrenciler ve sınıfları gözükmektedir. Ders kaydı da açılış ekranında mevcuttur. Bu sayede kendi veritabanımızı oluşturma imkanına da sahip olabiliyoruz. suiDBLookupListBox1 kompanentinde seçili öğrencinin ders kaydının yapılması için alabileceği dersler mevcuttur. Öğrenci seçildikten sonra aldığı dersler ya ders üzerine çift tıklanarak yada "ders ekle" butonuna basılarak dersler suiListBox1 içine atılır. Eğer seçili derslerden birini tekrar seçersek şu hata mesajıyla karşılaşırız(Şekil-8).

Şekil-8- Seçili derslerden biri tekrar seçilirse hata mesajı

"Ders Kaydını Yap" butonuna basılarak ders kaydı onaylanır.

Form2 öğrenci ve ders giriş sayfasıdır. Form1 üzerindeki

butonuna basılarak geçilir.

Öğrenci girme sayfasına git butonuna basılınca gelen formun şekli aşağıdaki gibidir (Şekil-9).

Şekil-9- Öğrenci ve der giriş sayfası

Bu formda Öğrencilerin Adı,Soyadı ve Sınıfı girilebiliyor. Editlere bilgiler girildikten sonra 

butonuna basılarak öğrenci kaydı onaylanır.

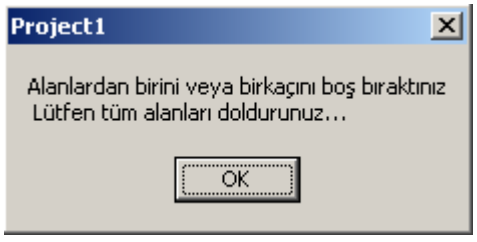
Ders ekleme kısmı yine bu formda yapılır. Ders bilgileri girildikten sonra 

butonuna basılarak ders kaydı onaylanır.

Eğer alanlardan birini yada bir kaçını boş bırakır ve 

veya 

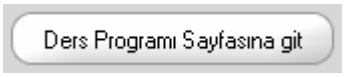
butonuna basarsanız karşınıza



hatası gelecektir.

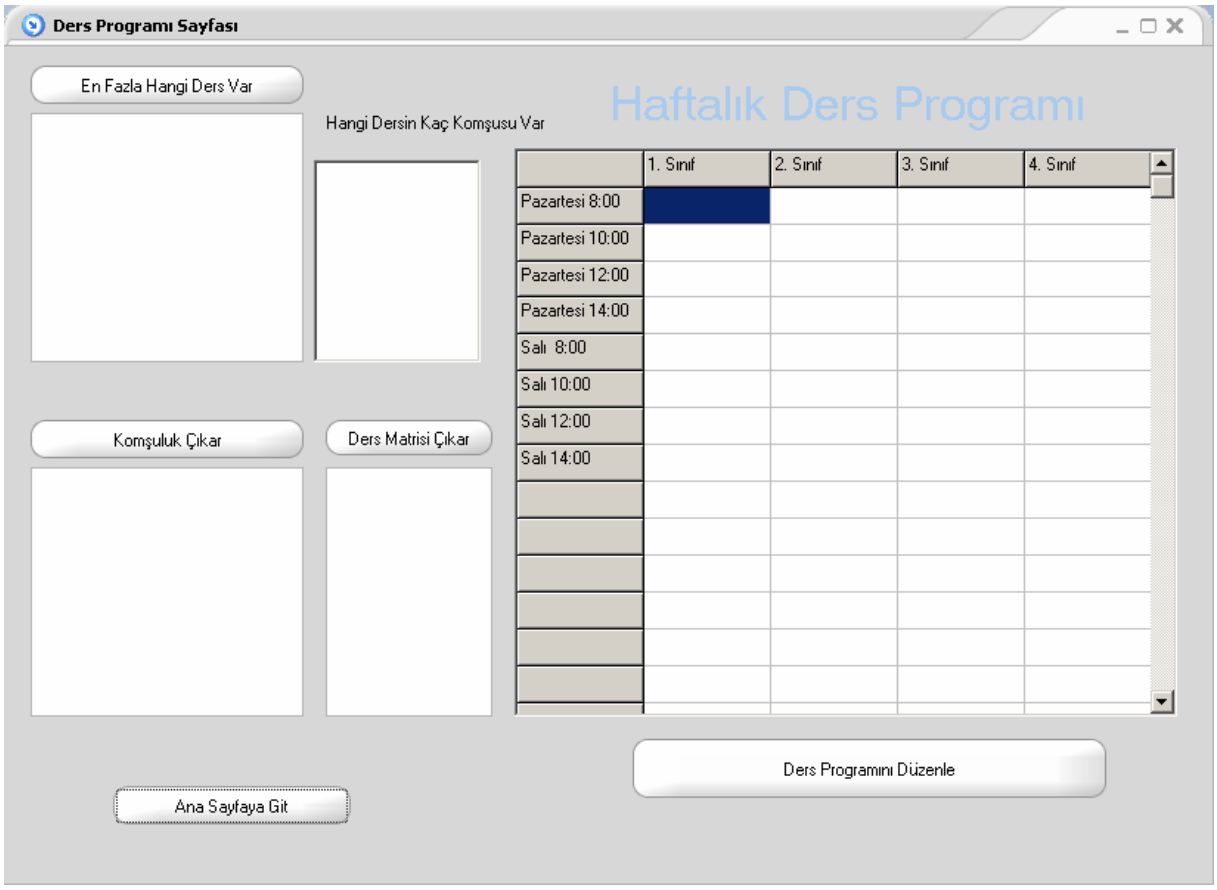
Yine form üzerinde bulunan 

butonuna tıklayarak ana sayfaya dönmüş olacağız.

Ana form üzerindeki 

butonuna basarak 3. formumuz olan 

na geçilir. Bu form projemiz için en önemli form olup ders yerleştirme işleminin gerçekleştirildiği formdur. Bu formu en ince ayrıntısına kadar incelememiz gerekir. Bu işleme başlarken Form3 ün görünümüne bakalım.Form3 aşağıdaki gibidir (Şekil-10).



Şekil-10- Ders programı sayfası

Bu çalışma akademik bir çalışma olduğu için Graph Coloring Algoritması için önemli bazı noktalar ayrı ayrı gösterildi. Bu önemli noktalar ListBox lar içerisinde gösterilerek nasıl çalıştığı daha iyi anlatılınabilinir.

Graph Coloring Algoritmasının temeli olan komşuluk çıkarma işlemi

butonuna basılarak gerçekleştirilir. Bu butona basıldığında bu işlemler icra edilir.

```
void Sirala(int dizi[100][2],int els)
{
  int i,j,t,i1,j1,t1;
  i=0;
  while(i<els-1)
  {
    j=i+1;
    while(j<els)
    {
      if(dizi[i][1]<dizi[j][1])
      {
        t=dizi[i][1];
        t1=dizi[i][0];
        dizi[i][1]=dizi[j][1];
        dizi[i][0]=dizi[j][0];
        dizi[j][1]=t;
        dizi[j][0]=t1;

      }
      j++;
    }
    i++;
  }

}
```

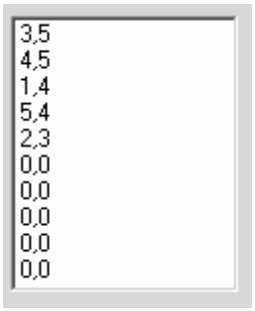
```
void sifirla(AnsiString dizi[100][100])
{
  int i,j;
  for(i=0;i<100;i++)
  {
    for(j=0;j<100;j++)
      dizi[i][j]='0';
  }
}
void __fastcall TForm3::suiButton3Click(TObject *Sender)
{
int i,k;
AnsiString temp_ders,ders_kodu[100];

   sifirla(komsuluk);
   Query1->Close();
   Query1->SQL->Clear();
   Query1->SQL->Add("select  ders_kodu from ders");
   Query1->Open();
   for(i=0;i<=Query1->RecordCount;i++)
   {
      komsuluk[i][0]=i+1;
   }
   for(i=0;i<Query1->RecordCount;i++)
   {
      temp_ders=Query1->FieldByName("ders_kodu")->AsString;
      //ders_ad[i]=Query1->FieldByName("ders_adi")->AsString;
      ders_kodu[i]=temp_ders;
      Query4->Close();
      Query4->SQL->Clear();
      //
      Query4->SQL->Add("select distinct ders_kodu from ogrenciders where ogr_no
in(select distinct ogr_no from ogrenciders where ders_kodu=\""+temp_ders+"\") and
ders_kodu!=\""+temp_ders+"\" ");
```

```
            Query4->Open();
            count[i][0]=i+1;
            count[i][1]=Query4->RecordCount+1;
            for(k=0;k<=Query4->RecordCount+1;k++)
            {
               komsuluk[i][k+1]=Query4->FieldByName("ders_kodu")->AsString;
               Query4->Next();

            }
               Query1->Next();

         }
         kayitsay=Query1->RecordCount;
         for(i=0;i<Query1->RecordCount;i++)
         {
         suiListBox2->Items->Add(komsuluk[i][0]);


         for(k=1;k<=count[i][1]-1;k++)
            {

               suiListBox2->Items->Strings[i]=suiListBox2->Items-
>Strings[i]+','+komsuluk[i][k];
            }
            }
          Sirala(count,99);
         for(i=0;i<10;i++)
          {
            ListBox1->Items->Add(count[i][0]);
            ListBox1->Items->Strings[i]=ListBox1->Items->Strings[i]+','+count[i][1];
          }
          suiButton3->Enabled=false;

      }
```

ve komşuluk çıkarma işlemi başarıyla tamamlanır.Komşuluk çıkarma işleminden sonra suiListBox2 nin ve butonun şekli aşağıdaki gibidir(Şekil-11).

Şekil-11- Komşuluk çıkarma

butonuna basıldığında icra edilen ikinci kısım ise kodda görüldüğü üzere hangi dersin kaç komşusu olduğu kısmıdır. Bu kısım algoritmamız da önemli yer tutar çünkü en fazla komşuluğa sahip dersten başlanarak yerleştirme yapılacaktır.ListBox1 de bu dizi görüntülenir ve aşağıdaki gibi olur(Şekil-12).



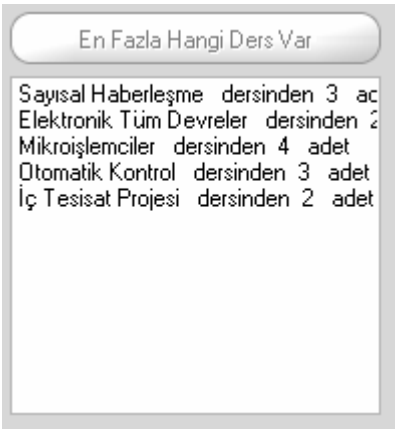
Şekil-12- Hangi dersin kaç komşusu var

Formdaki bir diğer görebileceğimiz şey hangi dersten kaç tane bulunduğu bilgisidir. Bunu görmek için 

butonuna tıklanarak sonuç görülebilir. Bu butonun altında icra edilen satırlar şunlardır.

```
void __fastcall TForm3::suiButton2Click(TObject *Sender)
{
int i;
String str;
    for(i=1;i<=Query2->RecordCount;i++)
    {
    Query1->Close();
    Query1->SQL->Clear();
    Query1->SQL->Add("select * from ogrenciders where
ders_kodu='"+IntToStr(i)+"'");
    Query1->Open();
    Query3->Close();
    Query3->SQL->Clear();
    Query3->SQL->Add("select * from ders where ders_kodu='"+IntToStr(i)+"'");
    Query3->Open();
    str=Query3->FieldByName("Ders_adi")->AsString+"   dersinden
"+IntToStr(Query1->RecordCount)+"   adet";
    suiListBox1->Items->Add(str);
    }
    suiButton2->Enabled=false;
}
```

İcra edilen bu satırlar sonrasında Form3 üzerinde bulunan suiListBox1'i şu şekilde görebiliriz(Şekil-13).



Şekil-13- En fazla hangi ders var

Projedeki en önemli kısım 

butonu altında icra edilen komutlardır. Bu buton altında daha önce bulunan komşuluklardan ve en fazla komşuluğa kim sahip gibi bilgiler yardımıyla bunların doğrultusunda en iyi şekilde ders programı çıkarılması amaçlanmıştır. Ve hiçbir öğrenciyi mağdur etmeden bunu başarabilmektedir.

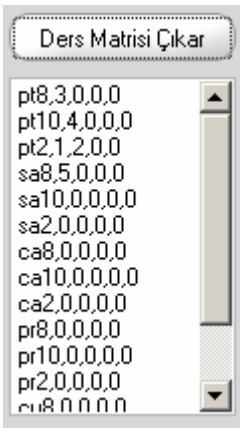
butonuna basıldığında şu satırlar icra edilir:

```
int komsumu(int deger,int indis)
{
int i,j,ara;
  for(i=1;renkler[indis][i]!='0';i++)
  {
    ara=StrToInt(renkler[indis][i]);
    for(j=0;komsuluk[ara][j]!=0;j++)
    {
      if(komsuluk[ara-1][j]==deger)
        return 0;
    }
  }
  return 1;
}
void yerlestir(int deger,int indis)
{
int i;
  for(i=1;renkler[indis][i]!=0;i++)
  { }
  renkler[indis][i]=deger;

}

void __fastcall TForm3::suiButton7Click(TObject *Sender)
{
bool tamam;
```

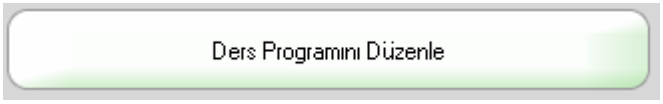
```
int i,k,deger;
  for(i=0;count[i][0]!=0;i++)
  {
    deger=count[i][0];
    k=0;
    tamam=true;
    while(tamam)
    {
      if(renkler[k][1]==0)
      {
        renkler[k][1]=deger;
        tamam=false;
      }
      else
      {
        if(komsumu(deger,k)!=0)
        {
          yerlestir(deger,k);
          tamam=false;
        }
        else
        {
          k++;
        }


      }
    }
  }

for(i=0;i<15;i++)
{
```

```
    suiListBox3->Items->Add(renkler[i][0]);
    suiListBox3->Items->Strings[i]=suiListBox3->Items->Strings[i]+','+renkler[i][1];
    suiListBox3->Items->Strings[i]=suiListBox3->Items->Strings[i]+','+renkler[i][2];
    suiListBox3->Items->Strings[i]=suiListBox3->Items->Strings[i]+','+renkler[i][3];
    suiListBox3->Items->Strings[i]=suiListBox3->Items->Strings[i]+','+renkler[i][4];

}
```

Ve sonuçta kullanıcı aşağıdaki bir görüntüyle karşılaşır(Şekil-14):



Şekil-14- Ders matrisi çıkar

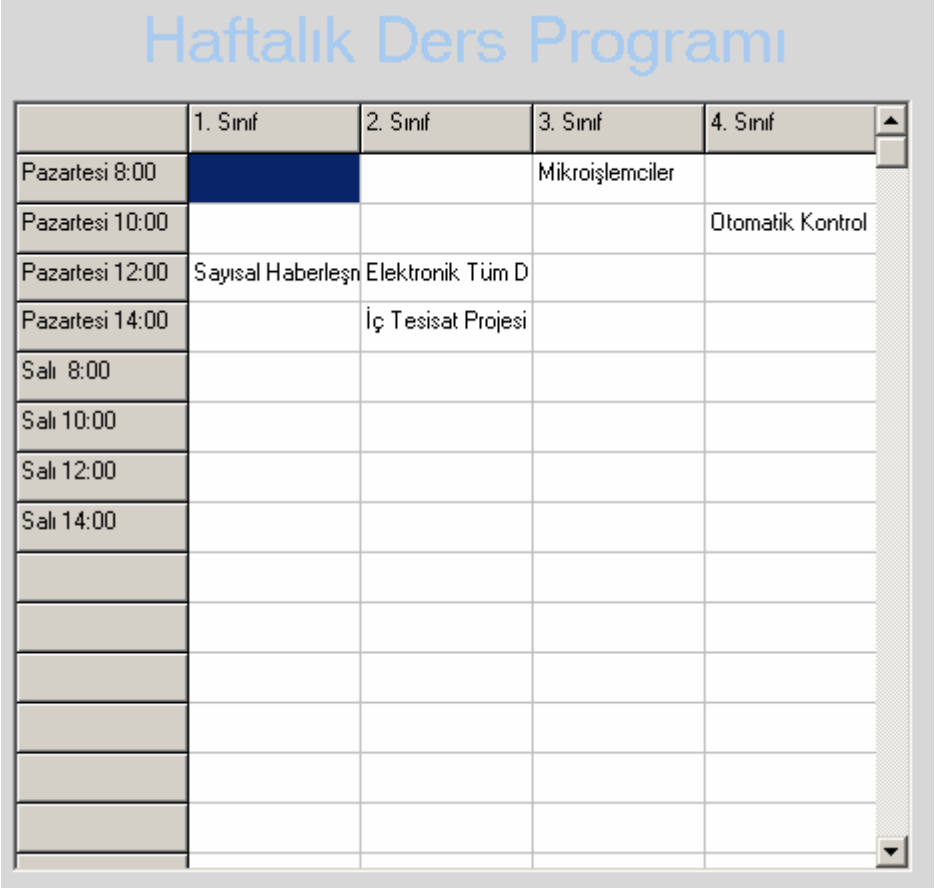
bu artık derslerin kodları ile yerleştirildiği son halidir. Bundan sonra yapılacak tek şey ders kodlarını çekip yerine isimlerini yazarak sınıfları da göz önüne alarak StringGrid1 içine yerleştirmektir. Bunu yapmak için 

butonuna basmak gerekir. Bu butona basıldığında şu satırlar icra edilir :

```
void __fastcall TForm3::suiButton8Click(TObject *Sender)
{
int i,j,sinif;
AnsiString ara;
```

```
for(i=0;i<10;i++)
{
  for(j=1;j<5;j++)
  {
    if(renkler[i][j]!=0)
    {
      Query6->SQL->Clear();
      ara=renkler[i][j];
      Query6->SQL->Add("Select * From ders where Ders_kodu=\""+ara+"\"");
      Query6->Close();
      Query6->Open();
      sinif=Query6->FieldByName("Sinif")->AsInteger;
      StringGrid1->Cells[sinif][i+1]=Query6->FieldByName("ders_adi")->AsString;

    }
   }

  }
}
```

ve ders programını StringGrid1 içine yerleştirmiş oluruz . Kullanıcı aşağıdaki gibi bir sonuçla karşılaşır (Şekil-15) :



Şekil-15- Ders programının yerleşik hali

SONUÇ

Sonuç olarak bu lisans tezinde üniversitelerin büyük bir sıkıntısı olan ders çakışmasını ortadan kaldıran bir projeyle uğraştık. Kullanımı basit ve işlevsel bir programdır.

KAYNAKLAR

[1] www.delphiturkiye.com

[2] ÇÖLKESEN, R, Veri Yapıları ve Algoritmalar-2002

[3] UYSAL, M, C ile Programlama-2000

[4] YANIK, M, Borland C++ Builder-2003