

YILDIZ TEKNİK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

Haftalık Ders Programı Optim.

Yüksek Lisans Tezi

Onsal Özdoğan

1991

209

86

T.C
YILDIZ UNİVERSİTESİ
FEN BİLİMLERİ ENSTİTUSU

10.000 TL

HAFTALIK DERS PROGRAMI
OPTİMİZASYONU

YÜKSEK LİSANS TEZİ

UNSAL ÖZDOĞRU

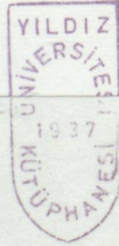
İSTANBUL - 1991

YILDIZ TEKNİK ÜNİVERSİTESİ
KÜTÜPHANE DOKÜMANTASYON
DAİRE BAŞKANLIĞI

R 209

86

Kot :.....
Alındığı Yer :...Fen. Bilimleri. Ens.....
Tarih :...18.03.1992.....
Fatura :.....-.....
Fiyatı :...20.000,-TL.....
Ayniyat No :...1/1.....
Kayıt No :...48203.....
UDC :...510.....
Ek :.....



YILDIZ TEKNİK ÜNİVERSİTESİ
KÜTÜPHANE DOKÜMANTASYON
DAİRE BAŞKANLIĞI

Kot R 209
: 86

Alındığı Yer : ... Fen. Bilimleri. Ens.

Tarih : ... 18.03.1992

Fatura : ... -

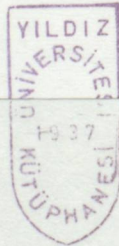
Fiyatı : ... 20.000,- TL

Ayniyat No : ... 1/1

Kayıt No : ... 48203

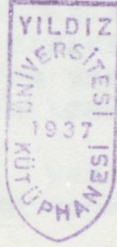
UDC : ... 510

Ek :



HAFTALIK DERS PROGRAMI

OPTİMİZASYONU



YÜKSEK LİSANS TEZİ

UNSAL ÖZDOĞRU

SUMMARY

TEŞEKKÜR

Ortak dayanışma, sabır, ilgi ve anlayışla gelen bir yaklaşım, eğitimdeki başarı için kaçınılmazdır.

Çalışmalarında önce kaynak araştırmaları için yardımcı olan, daha sonra düşünce ve fikirlerinden yararlandığım, dostça yaklaşımıyla yargının insanı yıpratıcı baskısını kırıp, çalışmamda ilerlememi sağlayan hocam Yrd. Doç. Dr. Ayşe Kuruüzüm'e teşekkürü bir borç bilirim.

Aynı zamanda, tez konumun seçiminde titizlikle davranan, gereken ilgiyi gösteren ve destek olan, mesleki ve akademik birikimini esirgemeyen hocam Prof. Dr. Behiç Çağal'a teşekkürlerimi sunarım.

Ortak çalışma ve anlayışın en güzel örneklerinden birini yılmadan sergileyen arkadaşlarım, özellikle bilgisayar programı aşamasındaki yardımları için Arş. Gör. Aydın Demiriz'e, elinden gelen yardımları için de Arş. Gör. Işım Genç ve Dilek Çelik'e teşekkür ederim.

Bana olan inancındaki yaklaşımıyla sağladığı manevi destekten dolayı anneme de sonsuz teşekkürlerimi sunarım.

SUMMARY

In this study is concerned with software of educational planning. It's seen that the applications have not several numbers relating with optimization of scheduling weekly courses, however some studies are closely met to my study. But, in general these are related with research allocation for faculty, such as assignment of faculty members to the courses or assignment of courses to the faculty members or spreading of examination days over an examination week for maximum success of students. This investigations are offered in the first chapter.

Toolbox package programme introduction we have been used and all knowledge about this study are given in the second chapter. Also, important some constraints and acceptance determined by decision maker to obtain optimum result from the prepared scheduling had been taken in this chapter. But, during input data into the computer, we saw that constraints weren't enough and in addition, appropriate some constraints and acceptance have been added for optimum outputs. Furthermore, variables-parameters and files in which are used the programme have been introduced in this chapter. Also in this chapter is explained the computer programme (software) prepared by using Toolbox package programme which two different assignment types define.

Computer programme and its sample outputs are given in the third chapter.

In the fourth chapter, utilities and deficiencies of our study, advantage and disadvantages of all constraints and acceptances that are used are explained.

İÇİNDEKİLER

ÖZET

SUMMARY

GİRİŞ

(1)

1. BÖLÜM	: Eğitim Planlama Konusunda Yapılan Çalışmalar	(2)
----------	--	-----

2. BÖLÜM	: Bilgisayar Programı için Yapılan Ön Hazırlıklar ve Programın tanıtımı	(10)
----------	---	------

2.1	: Kullanılan Paket Programın Tanıtımı	(10)
-----	---------------------------------------	------

2.2	: Kısıtlar ve Kabuller	(11)
-----	------------------------	------

2.3	: Değişken ve Parametreler	(12)
-----	----------------------------	------

2.4	: Programda Kullanılan Dosyalar	(13)
-----	---------------------------------	------

2.5	: Programın işleyişi	(14)
-----	----------------------	------

2.6	: Ek1 Turbo Access System (Toolbox) Erişim Modülleri	(17)
-----	---	------

3. BÖLÜM	: Bilgisayar Programı ve Örnek Çıktılar	(24)
----------	---	------

4. BÖLÜM	: Sonuç	(78)
----------	---------	------

ÖZGEÇMİŞ

KAYNAKLAR

Ö Z E T

Yapacağımız çalışmaya ışık tutması açısından, Eğitim Planlama Konusunda Yapılan Çalışmalar incelenmiş; Haftalık Ders Programı Tablolarının optimizasyonu ile ilgili bir çalışmaya rastlanmamakla birlikte, konuya yakın olan bazı incelemeler bulunmuştur. Bunlar, fakülte için kaynak tahsisatı, daha çok öğretim görevlilerinin derslere veya derslerin öğretim görevlilerine atanması veya öğrencilerin maksimum başarısı için, sınav günlerinin haftanın günleri üzerine dağılımı ile ilgilidir. Bu konu ile ilgili olan incelemeler birinci bölümde sunulmuştur.

İkinci bölümde ise, kullanmış olduğumuz TOOLBOX paket programın tanıtımı ve yapmış olduğumuz çalışmayla ilgili her türlü bilginin verildiği kısımdır. Bu bölümde öncelikle, hazırlanan tabloların optimum sonuç vermesi için karar verici tarafından belirlenen önemli bazı kısıt ve kabuller ele alınmıştır. Fakat, verilerin bilgisayara aktarılması sırasında bu kısıtların yeterli olmadığı görülmüş ve bunlara optimum sonuç için uygun görülen bazı kısıt ve kabuller de eklemiştir. Ayrıca, bu bölümde programda kullanılan değişken ve parametreler, dosyalar da tanıtılmıştır. Son kısımda ise, Toolbox paket programı kullanılarak yapmış olduğumuz bilgisayar programı ele alınmış; programdaki iki farklı atama şekli açıklanmıştır.

Bilgisayar Programı ve örnek çıktıları ise üçüncü bölümde verilmiştir.

Dördüncü ve son bölümde, yapmış olduğumuz çalışmanın getirdiği yararlar ve eksiklikleri, işlediğimiz tüm kısıt ve kabullerin avantajları ve dezavantajları anlatılmıştır.

Yapılan örnek çalışmada, bir örnek olarak, bir günde yapılacak maksimum ders sayısı ve haftaya dağılımı son derece önemlidir.

Yapılan örnek çalışmada, bir örnek olarak, bir karar vericisiyle, öğrencilerin sınav konularına ve sınavın mertebeye göre çıktıları, bilgisayarla optimum çözüm için programlanmıştır.

G İ R İ Ş

Eğitim ve Öğretim Sistemlerinde karşılaşılan en önemli problemlerden biri de öğrencilerin kendilerini aşan bir enerjiyle yoğun bir çalışma içinde bulunmalarındır. Bu yoğunluğu gidermese de yapılan bir takım sosyal faaliyetler ve yetenek dersleri onları, bu çalışma ortamının stresinden uzaklaştırmaktadır.

Bu çalışmada ele alınan konu, derslerin, öğrenci kapasitesini zorlamayacak bir biçimde haftanın günleri üzerine uygun bir biçimde dağılımını sağlamaktır.

Orta Öğretim de bu yerleştirme işlemi, Yüksek Öğretime oranla daha kolaydır. Çünkü, Yüksek Öğretimde bilhassa bazı derslerde, çeşitli sebeplerden dolayı, dersi alan öğrenci sayısında bir yığılma olmaktadır. Bu yığılma ise çakışmalara neden olduğundan, ya öğrencinin bir üstteki dersi almaması, ya da her iki dersi alıp, alttaki derse devam mecburiyetinin kaldırılması demektir. Şu açık bir gerçektir ki, ilk alıştırma devam zorunlu olduğundan, devam ettiği halde kalan öğrencinin devam etmeden bu dersi vermesi bazı istisnai durumlar gözönüne alınmayınca oldukça zordur. Bu nedenle öğrencinin bir dersi başarması her şeyden önce o derse devam etmesi ve dersi derste öğrenmesiyle mümkün olacaktır. Tabii bu arada öğrencinin başarısını etkileyen diğer durumlar da görmemezlikten gelinmez. Bir günde yapılan ders sayısının çokluğu, derslerin üstüste işlenmesi ve bunlara benzer sebepler başarı düşüşlerine neden olduğundan Haftalık Ders Programı oluşturulurken bütün bunlar gözönünde tutulmaktadır.

Bunların yanı sıra, bu tablolarda Öğretim elemanlarının randımanı ve yapacağı bilimsel çalışmalar için gerekli zaman açısından da, görevlendirilmiş olduğu derslerin, bir günde vereceği maksimum ders saati ve haftaya dağılımı son derece önemlidir.

Yapılan örnek çalışmada söz konusu etkin durumlar, bir karar vericiyle etkileşerek ortaya konulmuş ve mümkün mertebede ortaya çıkan kısıtlar bilgisayarla optimum çözüm için programlaştırılmıştır.

1 . B Ö L Ü M

EĞİTİM PLANLAMA KONUSUNDA YAPILAN ÇALIŞMALAR

Bir eğitim sistemi çeşitli düzeylerde, ulusal düzeyde, bir üniversite seviyesinde veya belirli bir üniversitenin özel bir bölümü veya disiplini düzeyinde tanımlanabilir. Her seviyede önemli planlama problemleri ortaya çıkmaktadır.

Eğitim Planlama (Educational Planning) konusundaki yayınlar genellikle fakülte veya kaynakların tahsisi (Resource Allocation) [1,2,3] veya öğretim üyelerinin derslere atanmasıyla ilgilidir [2,4].

Konuya yakın olduğu için sınavları veya dersleri planlayan, incelenmiş modeller şöyle özetlenebilir.

Mc Camley ve Plessner bir fakültenin, derslerinin atanmasının yapıldığı bir model geliştirmiştir [2,s:273]. Model bir LP (Lineer Programlama) modelidir; yani bir kısıtlar kümesi ile bu kısıtlara göre maksimize edilebilen bir amaç fonksiyonundan oluşmaktadır.

Bu modeldeki tablolarda bir ders, 3 kredi saattir. Tabloları oluşturmak için, önce dersler ilgili öğretim üyesine atanmaktadır. Böylece i. öğretim üyesi her bir dönemde 3 kredilik 3 dersin öğreniminden yada bütün bir akademik yıldaki 9 dersten sorumludur ($a_i=9$). Benzer biçimde j. seçenekte, üç sömestirin herbirinde özel olarak seçilmiş 3 saatlik 3 dersin önerilen bağımsız 2 bölümünden ibarettir ($b_j=6$ kredi saat). Yani öğretim üyeleri bir dönemde kendilerine atanmış olan 3 dersten herhangi ikisini de seçebileceklerdir. Eğer i. öğretim üyesi önerilen bu derslerin 2 sinide öğretmek için atanmışsa $x_{ij}=6$ olacaktır. Fakat, i. öğretim üyesi 3 dersi de bir sömestirde vermek istiyorsa bu bilgi tablolarda atama yapmak için kullanılabilecektir.

Modelde n öğretim üyesi ve m farklı atamadan oluşan bir bölüm tasarlanmış ve bu bölümle ilgili kısıtlar :

$$\sum_{j=1}^n x_{ij} \leq a_i \quad i=1,2,\dots,n \quad (1)$$

$$\sum_{j=1}^n x_{ij} = b_j \quad j=1,2,\dots,m \quad (2)$$

$$x_{ij} \geq 0 \quad i=1,2,\dots,n; j=1,2,\dots,m. \quad (3)$$

şeklinde ifade edilmiştir. (1) eşitsizliği, her bir öğretim

üyesine tahsis edilen tüm zamanın bu tahsis için kullanılabilen miktarı kesinlikle aşamayacağını göstermektedir. (2) eşitsizliği atamaların tümünün sağlandığını ifade etmektedir. (3) eşitsizliği pozitiflik koşulunu gerçeklemektedir. x_{ij} ler, j . seçeneğe atanmış i . öğretim üyesinin süresini temsil etmektedir. a_i ler, i . öğretim üyesinin bölümce kullanılmasına izin verdiği sürenin miktarı, b_j ler ise j . seçeneğin gerektirdiği öğretim süresini ifade etmektedir.

$$\sum_{i=1}^n a_i < \sum_{j=1}^m b_j \quad \text{ise uygun çözüm yoktur;}$$

$$\sum_{i=1}^n a_i \geq \sum_{j=1}^m b_j \quad \text{ise uygun çözüm vardır;}$$

$$\sum_{i=1}^n a_i > \sum_{j=1}^m b_j \quad \text{ise bölümde kadro fazlalığı vardır;}$$

$$\sum_{i=1}^n a_i = \sum_{j=1}^m b_j \quad \text{ise öğretim üyeleri ile verilen}$$

dersler arasında bir denge var demektir. Bu durumda kısıtlar "Transport Model" 'in kısıtları ile aynı formdadır; ve model bu tip problemlerin çözümünde kullanılan yöntemlerin herhangi birisiyle çözülebilir.

Modelde amaç fonksiyonu,

$$\sum_{i=1}^n \sum_{j=1}^m c_{ij} x_{ij}$$

şeklinde ifade edilmektedir. Burada c_{ij} , i . öğretim üyesinin j . dersi vermesi halinde elde edilen sonucun birim başına değeridir; x_{ij} ise i . öğretim üyesinin j . dersi vermesi için ayrılan süredir.

Böyle bir model elle çözülebilen küçük modeller içindir. Çözüm işlemi iki aşamadan oluşmaktadır. Birinci aşamada uygun bir çözüm elde edilmekte, ikinci aşamada ise optimal çözüme ulaşılmaktadır. İlk aşama uygunluk sağlanana kadar tekrar edilen tek bir adımdan oluşmaktadır. Modelde sıfırdan daha büyük olan a_i ve b_j lerin herbiri için

c_{1j} 'lerin en büyüğü seçilmektedir. Eğer bu eleman $c_{r=}$ ise, $x_{r=}$ kümesi, $b_{r=}$ ve $a_{r=}$ nin mevcut değerlerinden en büyüğüne eşittir. $a_{r=}$ ve $b_{r=}$ nin her ikisinden de $x_{r=}$ nin çıkarılmasıyla, yeni değerleri bulunur. a_{1j} ve b_{1j} ler sıfır oluncaya kadar bu adım sürekli tekrarlanır.

ikinci aşama esnasında, optimum çözüm elde edilinceye kadar çözüm sürekli geliştirilir. En iyi çözümün elde edilmesinde tek yol ayrılan bazı vektörlerin (x_{1j} lerden birisinin) sıfır olan bir seviyeden pozitif bir seviyeye arttırılmasıdır. İlk adım, arttırılacak faaliyet seviyesinin belirlenmesini içerir. Genellikle faaliyet seviyelerinin çoğu sıfır olacaktır; ve buna ek olarak verilen herhangi bir faaliyet seviyesini arttırmak için daima birkaç yol bulunacaktır [2,s:375].

Eğer x_{1j} , sıfırdan daha büyükse, dual değişkenler u_i (i. öğretim üyesinin verdiği hizmetin bir biriminin marjinal değeri) ve v_j (j. işlemin talep ettiği girdinin bir biriminin marjinal değeri) arasındaki

$$c_{1j} = u_i - v_j$$

ilişkisini sağlamaktır. Bu durum, u_i ve v_j için kolayca çözülebilen $n+m-1$ (veya daha az) denklem sisteminin kurulmasını sağlar. Sıfırdan daha büyük olan $c_{1j} - u_i + v_j$ için bütün x_{1j} 'ler faaliyet seviyesinin artması için birer adaydır. İlk artımda x_{1j} , en büyük $c_{1j} - u_i + v_j$ için tektir.

Bir sonraki adım, faaliyet seviyesinin ne kadar artacağını belirlemeyi içerir. Herhangi bir faaliyet seviyesinin (örneğin $x_{r=}$) artması için en azından iki faaliyet seviyesinin azalması ve hiç olmazsa bir faaliyet seviyesinin artması gereklidir. $x_{r=}$ faaliyetini artırmanın bir yolu,

$$\sum_{i=1}^n \Delta x_{1j} = 0 \quad j=1,2,\dots,m,$$

$$\sum_{j=1}^m \Delta x_{1j} = 0 \quad i=1,2,\dots,n,$$

$$\text{ve } \Delta x_{r=}=1, \Delta x_{1j} + \Delta x_{1j} \geq 0 \quad i=1,2,\dots,m \quad j=1,2,\dots,n$$

olmak koşuluyla, Tablo (3) de optimal çözüm ve onun dual matrix formunda verilmektedir.

$$\sum_{i=1}^n \sum_{j=1}^m \Delta x_{1j} C_{1j}$$

yi maksimize etmektir.

ikinci aşama adımları, birkaç nokta için

$$c_i, \leq u_i - v_j, \quad i=1,2,\dots,n, \quad j=1,2,\dots,m.$$

sağlanıncaya kadar tekrarlanır. Bu noktada elde edilen çözüm, optimal çözümdür.

Modelde, herbiri 9 kısma eğitim verecek veya, eğitim ve araştırma çalışmaları için yardım edecek 4 öğretim üyesi olan bir bölüm tasarlanmıştır. Bu bölümdeki 6 dersin birim saatleri 9,7,5,5,4 ve 3 olarak saptanmıştır. Aynı zamanda sırasıyla 2 ve 1 saat çalıştırma gerektiren iki küçük araştırma porjeside yaptıracaklardır. Günlük ders saati ve dersler arasında bir esneklik olduğu kabul edilmektedir.

Bu problem için gerekli bilgiler Tablo (1) de verilmektedir. Burada 32 faaliyet vardır. a_i 'ler pozitif olarak verilmiştir ve arzı ifade etmektedir. b_j 'ler ise negatif işaretlidir ve talebi ifade etmektedir. Tablodaki c_i değerleri bölüm başkanı tarafından belirlenmiştir.

TABLO (1)

Öğretim Üyesi	Dersler						Araştırma		Birim başına düşen a_i
	1	2	3	4	5	6	7	8	
1	10	7	5	9	15	4	6	3	9
2	6	8	4	3	7	5	12	2	9
3	7	7	6	11	10	9	5	1	9
4	5	6	2	5	3	8	7	6	9
b_j	9	7	5	5	4	3	2	1	36

Tablo (2) ve Tablo (3) de optimal çözüm ve onun duali matris formunda verilmiştir.

TABLO(2)

Öğretim üyesi (F ₁) ve Ders ve Proje(R ₃)		Atanan birimler										Sonuç
		X ₁₁	X ₁₅	X ₂₂	X ₂₇	X ₃₃	X ₃₄	X ₄₁	X ₄₃	X ₄₆	X ₄₈	
F ₁	1	0	0	0	0	0	0	0	0	0	0	5
F ₂	0	0	0	0	0	0	0	0	0	0	0	4
F ₃	0	0	0	0	0	1	1	0	0	0	0	7
F ₄	0	0	0	0	0	0	0	1	1	1	1	2
R ₁	-1	0	0	-1	0	0	0	-1	0	0	0	4
R ₂	0	0	0	0	0	0	0	0	0	0	0	5
R ₃	0	0	0	0	0	-1	-1	0	-1	0	0	4
R ₄	0	0	0	0	0	0	-1	0	0	0	0	1
R ₅	0	-1	0	0	0	0	0	0	0	0	0	3
R ₆	0	0	0	0	0	0	0	0	0	-1	0	1
R ₇	0	0	-1	0	0	0	0	0	0	0	0	
R ₈	0	0	0	0	0	0	0	0	0	0	-1	

$$= \begin{bmatrix} 9 \\ 9 \\ 9 \\ 9 \\ -9 \\ -7 \\ -5 \\ -5 \\ -4 \\ -3 \\ -2 \\ -1 \end{bmatrix}$$

TABLO (3)

$$\begin{bmatrix}
 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 \\
 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0
 \end{bmatrix}
 \begin{bmatrix}
 u1 \\
 u2 \\
 u3 \\
 u4 \\
 v1 \\
 v2 \\
 v3 \\
 v4 \\
 v5 \\
 v6 \\
 v7 \\
 v8
 \end{bmatrix}
 =
 \begin{bmatrix}
 10 \\
 15 \\
 8 \\
 12 \\
 6 \\
 11 \\
 5 \\
 2 \\
 8 \\
 6
 \end{bmatrix}
 \quad (4)$$

Son atamalar Tablo (4) de görüldüğü gibidir; ve amaç fonksiyonunun değeri ,

$$W = \sum_{i=1}^4 \sum_{j=1}^8 C_{ij} x_{ij} = 321$$

dir. Bu da optimal çözümdür.

TABLO(4)

Öğretim Üyesi	Atamalar Ders					Araştırma			a_i
	1	2	3	4	5	6	7	8	
1	5				4				9
2		7					2		9
3			4	5					9
4	4		1			3		1	9
b_j	9	7	5	5	4	3	2	1	36

Fakülte öğretim üyelerine derslerin atanmasını sağlayan bir diğer yaklaşım McClure ve Wells'in tamsayılı programlama modelidir [4,s:409-419] . McClure ve Wells 0-1 programlama tekniğini kullanarak her öğretim üyesine alternatif 10 program önermekte bunlar arasından kendine en uygun olanını seçmesini istemektedir. Söz konusu modelde kullanılan formüller aşağıdaki gibidir.

$$\max \sum_{i,j \in J_1} c(i,j) \cdot x(i,j) \quad (4)$$

Amaç fonksiyonu ,

$$\sum_{j \in J_1} x(i,j) = 1 \quad i=1,2,\dots,i^* \quad (5)$$

$$\sum_{i,j \in J_1} c(i,j,k) \cdot x(i,j) = d(k) \quad k=1,2,\dots,k^* \quad (6)$$

kısıtları altında optimize edilmektedir. Burada,

i^* : Öğretim üyelerinin sayısı;

k^* : verilen ders sayısı;

$d(k)$: verilen k dersin bölümlerinin sayısı;

J_1 : i öğretim üyesi için mümkün tabloların kümesi;

$x(i,j) < \begin{cases} 1, & \text{eğer } i \text{ nolu fakülte üyesi } j \text{ nolu programı} \\ & \text{alıyorsa,} \\ 0, & \text{diğer durumlarda;} \end{cases}$

$u(i,j)$: j nolu tablo için i öğretim üyesinin fayda ölçüsü;

$c(i,j,k)$: i nolu öğretim üyesi için, j tablosu içinde görülen k dersinin tekrar sayısı;

Model, bütün öğretim üyelerinin faydalarını, kendi atama tabloları için makzimizasyonunu amaçlıyor.

Burada iki kısıt söz konusudur :

1-) (5) nolu denklem, her öğretim üyesinin sadece bir ders planına atanmasını

2-) (3) denkleme ise, atanan tabloların herbir i için gerekli bölüm sayısını içermesini sağlamaktadır.

Bu modelde, bir öğretim üyesine bir dersten ziyade, bir programın atanması söz konusudur.

Ayrıca, program faydaları, ders faydalarının hesaplanması yerine öğretim üyeleri tarafından belirlenmektedir. Böylece derslerin yararları arasındaki etkileşimin anlaşılması daha kolaydır [4,s:411]. Herbir fakülte öğretim üyesinin ders planını veya uygun koşulları bozmamak kaydıyla bir kişinin uygun bütün derslerinin hepsinin kümesinden, mümkün tablolara karar verilir. Herbir fakülte öğretim üyesi için verilerin kabul edilebilir sayıları, ders saatleri önceden hazırlanmıştır; ve bu oluşan tablonun doğal yapısında vardır. Ders verileri ve öğretim planı kısıtlarının modelde açıkça belirtilmesine gerek yoktur.

Bu modeli tanımlamak için gerekli adımlar şunlardır. İlk olarak herbir öğretim üyesi aşağıdaki bilgileri içeren bir rapor verir :

- 1-) Gelecek sömestir için önerilen dersler;
- 2-) Öğretim üyelerinin genellikle ilgilendiği dersler;
- 3-) Öğretim tablosundaki standart maksimum ve minimum derslerim sayısı;
- 4-) Öğretim tablosu içindeki hazırlıkların standart maksimum sayısı ;

Bir öğretim üyesi belirli bir sömestir için ilgilendiği ders planında düzenli artış ve azalışları, öğretim saatlerini veya verilerin sayısını değiştirmeyi, derslerin listesinden ekleme veya çıkarma yapmayı isteyebilir. Bu değişiklikler yapıldığı zaman, öğretim üyesi isteklerini yönetimdeki görevli olan kişiye rapor eder; ve götürür. Böylece istekleri doğrultusundaki değişiklikler yapılarak ders atmaları sonuçlanabilir. Öğretim üyesi herhangi bir değişiklik yaparsa yeni bir tablo üretilmelidir. Bilgisayar kullanarak herbir fakülte üyesi için mümkün tabloları çabucak, zorlanmaksızın üretilebilir.

Bilgisayar programı yazılarak birkaç saniyede yüzlerce tablo üretilebilir. Bununla birlikte, oluşturulan çok fazla tablo bazen en iyi tablonun seçiminde güçlük yaratabilir. Bu problemde de bir fakülte öğretim üyesinin seçimiyle yani, derslerin sayısının sınırlanmasına karar verilir. Bu yöntem fakültenin büyüklüğüne bakılmaksızın etkin olabilir. Çünkü, öğretim üyeleri için oluşturulan planların kompozite durumu öğretim üyelerinin sayısının bir sonucu değil, bireysel olarak seçilen farklı derslerin sayısının bir sonucudur[4,s:411].

Bu modelde yerine getirilecek ikinci adım, herbir tabloya bir hizmetin atanabilmesi için mümkün tabloların listesini öğretim üyelerine vermektir. Geçmiş fayda değerleri her oluşturulan tablo için öğretim üyesine verilmektedir. Ancak modelde, öğretim üyeleri çeşitli planlar için faydaları belirlerken bazı küçük güçlüklerle karşılaşmaktadırlar. Fayda ölçüsü 1-100 arasında değişmektedir. Burada 100 en çok istenen plana karşılık gelmektedir.

2 . B Ö L Ü M

BİLGİSAYAR PROGRAMI İÇİN YAPILAN ÖN HAZIRLIKLAR

VE

PROGRAMIN TANITIMI

2.1 TURBO PASCAL ACCESS (T.P.A) :

Turbo Pascal Access (T.P.A) kullanmak son derece kolaydır. Eğer Turbo Pascal (T.P) programı yazmayı biliyorsanız, T.P.A programları içindeki işleyişi bilmeksizin kayıt yapabilir, kayıtlara çok çabuk erişebilir, indexleyebilir veya silebilirsiniz.

T.P.A kullanmanın 3 önemli faydası vardır:

- 1-) Kolayca dosya yaratıp, kayıt yapabilir;
- 2-) Kayıtlarınızı otomatikman indexleyebilir (indexleme işlemi yapmadan);
- 3-) Yeni kayıt ekleyebilir veya çıkarabilirsiniz.

Örneğin, bir kaydı 300 karakter (byte) içeren ve en az 1000 kişinin kayıtlarının saklanabileceği bir program yazmak T.P de hiç zor değildir. Eğer kayıtlar sıralı erişimli bir dosya kullanılarak yapılmışsa 300x1000 yani 300 KB lık bir DATA dosyasında herhangi bir kayda erişim çok kolaydır; ancak pek etkin olmayan bir biçimde yapılır. Bu basit fakat zaman açısından uygun olmayan bir yöntemdir.

Diğer bir yaklaşım ise, bütün kayıtlar için bir index dosyası yaratmaktır. Kayıtları alfabetik sıralamaya daha sonra yeni kayıt girildiğinde ise disket üzerinde sıralamaya almaktır. Böyle yapılan bir programda. verilen ismin aranması oldukça etkindir ve aşağı yukarı IBM PC 'de 5 sn'de gerçekleşir. Bilgilerin sıralanması ve iyi bir erişim programı yazılımının güçlüğü yanı sıra bu yaklaşım kayıtlara ulaşmak için iyi bir yol sağlamaz. Örneğin, ismi Smith ve Smithy olan iki kayıt ardarda disk(et) üzerinde sıralı olarak bulunuyorsa, ismi Smithson olan bir başka kayıt eklendiğinde, alfabetik sıralamada bulunan iki kayıt arasına herhangi bir işlem yapmaksızın yerleştirilmesi mümkün değildir.

Daha iyi bir yaklaşım, belirli bir sayıda bir bloğun kayıtlar için ayrılmasıdır. Bu durumda A için belirli sayıda kayıt, B için belirli sayıda kayıt, ... ayrılır. Fakat bu yönteminde sakıncası aşıkardır. Ayrılan yerlerin dolması durumunda bir başka kayıt söz konusu olmayacaktır.

Disk(et) üzerinde kayıtların organizasyonu görüldüğünden çok daha zor bir problemdir. İşte T.P.A da kayıtları B+TREE yapısıyla saklar ve dolayısıyla erişim çok kolaydır.

T.P.A da kayıtlara uygun bir biçimde erişebilmek için karışık bir biçimde bulunan kayıtları, kümeleyerek tutar; her an herhangi bir kaydı yapabilir, silebilir, değiştirebilirsiniz. Bütün bu işlemlerden sonra disket üzerindeki tablolar otomatik olarak güncelleşir. Bu ise kayıt erişim zamanının azalması ve belleğin uygun bir biçimde kullanılması demektir. 300 KB lık bir DATA dosyası içindeki bir kayda T.P.A 'la KEY'ler yardımıyla, bir IBM PC 2 sn gibi bir sürede erişilmektedir.

Turbo Access (TACCESS.PAS) programının içindeki procedure'ler ve işlevleri EK'de verilmiştir.

2.2 KISITLAR ve KABULLER :

Bir Haftalık Ders Programı oluşturulurken, bu tabloların meydana gelmesinde istenen bazı zorunlu şartlar, kesinlikle uyulması gereken bazı kurallar ve kendiliğinden esnek olan bazı koşulların ilk adımda gerçekleştirilmesi, bu tabloların optimum sonuç vermesinde son derece önemlidir. Kendiliğinden ortaya çıkan bu sınırlamalarla ve daha önceden belirlenen bazı verilerin otomatik olarak Haftalık Ders Programına konması bu yüzden bir zorunluluktur. Bütün bu bilgiler gözönüne alındığında, Kısıtlar ve Kabuller adı altında topladığımız bazı şartlar ortaya çıkar.

Kısıtlar ve Kabullerin bir kısmı bir karar vericiyle karşılıklı etkileşerek belirlenmiştir; bir kısmı ise doğal olarak Eğitim ve Öğretim sisteminin iyi bir biçimde işleyebilmesi açısından, bu işle ilgili kuruluşlarda uyulması gereken bir zorunluluk olarak ortaya çıkmıştır.

Bunları birkaç madde altında toplayalım :

KABULLER :

- 1-) 4. ve 3. sınıf öğrencilerinin haftanın bir günü dersi olmamalı
- 2-) Bütün sınıfların haftanın bir günü bölüm tarafından istenen günü boş olmalı
- 3-) Öğretim elemanlarının isteklerine bağlı olarak, haftanın istenen gün ve saatine ilgili ders atanmalı;
- 4-) Türkçe , Tarih , Yabancı Dil dersleri ilgili sömestirde daha önceden belirlenen bir saatte yapıldığından bu dersler baştan atanmalı.(Eğer bu derslerde belirli bir gün ve saatte atanmak

istenmiyorsa, bu derslere sömestirine göre numara (KEY) verilerek atamalar arasında işleme konulabilir.)

KISITLAR :

- 1-) Bir günlük ders saati sayısı 8 den fazla olmamalı;
- 2-) Dersler mümkün olduğunca sabah saatlerine atanmalı;
- 3-) Bir ders 3 saatten fazla ise 2 ' şerli dağılım yapılmalı; (Öğretim görevlisi eğer istiyorsa 3 saat üstüste yapabilir.)
- 4-) Üst sınıftaki derslerle, alttaki ilgili dersler minimum sayıda çakışmalı
- 5-) Bir öğretim üyesi aynı dersi farklı günlerde aynı saatte yapamaz.
- 6-) Bir günde 3 branş dersinden fazlası yapılamaz.

Yukarıda maddeler halinde yazılmış olan kısıt ve kabuller başlangıç bilgileri olarak ele alınmış ve işlenmiştir. Fakat, ders programının optimum sonuç vermesi için bunlara bazı kısıtlar da eklenmiştir; ve bunlar, programın anlatım kısmında ileride sunulacaktır.

2.3 DEĞİŞKEN ve PARAMETRELER :

Drec,Prec	: Kayıt Değişkeni
ProgDos,DersDos	: Data Dosyası
ProgInx,DersInx	: Index Dosyası
Drecnum,Precnum	: Kayıt Numarası
K	: Data Dosyasında atama yapılmamış ders sayısı
Ds	: Bir dersin toplam,Teorik Uygulama sayısı
Snf,Sınıf,Sf,Nf	: Sınıf değişkenleri
Gun	: İstenen (işlenen) gün
Saat	: İstenen (işlenen) saat
Ogr, Hoca	: Öğretim görevlisi
Dno	: Ders Numarası
Dadı	: Ders adı
Tu	: Teorik Uygulama
Td11	: Tarih ,Türkçe ve Yabancı Dil
G	: (M)atematik, (F)izik, (K)imya , (I)statistik bölümlerine ait kodlar
Bıl	: Güz (GUZ) ve Bahar (BAH)
Yıl	: İçinde bulunduğumuz yıl
Der_Say	: Bir günde yapılan ders sayısını taşıyan dizi (max = 3)
De,Dy	: Atanmayan dersleri taşıyan diziler
Ss,Sy	: Atanmayan derslerin, her birinin toplam ders saatini taşıyan diziler
Bl[h]	: H değişkenine bağlı olarak, bölümlerin atanmış olduğu dizi
Gn[1]	: I. değişkene bağlı olarak her günün atanmış olduğu dizi

Dprog[Sınıf, Gun, Saat] :

Haftalık Ders Programı tablolarında her bilgi, üç boyutlu bir dizi gibi depolanmışsa da , bu aslında 5 tane matris olarak düşünülebilir. Her matrise, günlere karşılık gelen değerlerine göre bir numara verilmiştir. Bu durum , atamaların daha kolay anlaşılmasını sağlamaktadır. Çünkü, geriye kalan form 9x4 boyutlu bir matris olmaktadır. Her sütun, daha önceden belirlenen GÜN'ün , sınıfına karşılık gelmektedir; ve satırlarındaki bilgiler ise, belirlenen GÜN'ün ilgili sınıfına ait o gün yapılan dersleri göstermektedir.

2.4 PROGRAMDA KULLANILAN DOSYALAR :

Örnek programda, bir Ders dosyası birde Program dosyası oluşturulmuştur. Ders dosyasında ilgili bölümün sömestirine (GÜZ ve BAHAR) ait ders bilgilerinde, her derse bir numara (bu numara KEY olarak kullanılmıştır) verilmiştir. Ders Dosyasında, Ders adı, Öğretim Görevlisi ve Teorik-Uygulama sayısı her derse ait bir kayıt olarak tutulmuştur.

Programda örnek olarak alınan fakültenin Matematik, Fizik, Kimya, İstatistik olmak üzere 4 bölümü vardır. Bu 4 bölümün Güz ve Bahar yarıyılı olmak üzere iki sömestiri olduğundan, herhangi bir bölümün bilgilerine erişebilmek için, bu bölüme ait kodun (M, F, K, I) ve sömestirin girilmesiyle erişilebilecektir. Örneğin, Matematik bölümünün Bahar yarıyılına ait derslerine ait bir bilgiye erişilecekse MBAHDERS.DAT dosyasının , Haftalık Ders Programına ait bir bilgiye erişilecekse MBAHPROG.DAT dosyasının açılması gerekir. Bu dosyalar açılırken aynı zamanda bunlara ait sırasıyla MBAHDERS.INX ve MBAHPROG.INX index dosyaları da aynı anda açılmaktadır.

Şimdi Ders ve Program dosyaları aşağıda verilmiştir :

DERS DOSYALARI :

MGUZDERS.DAT
MBAHDERS.DAT
FGUZDERS.DAT
FBAHDERS.DAT
KGUZDERS.DAT
KBAHDERS.DAT
IGUZDERS.DAT
IBAHBERS.DAT

PROGRAM DOSYALARI :

MGUZPROG.DAT
MBAHPROG.DAT
FGUZPROG.DAT
FBAHPROG.DAT
KGUZPROG.DAT
KBAHPROG.DAT
IGUZPROG.DAT
IBAHPROG.DAT

Bu dosyalar üzerinde kayıt, gözlem, değişiklik, iptal, listenin ekran veya printer çıkışı yapılabilmektedir. Yukarıda açılan herbir dosyaya ait bir de index dosyası yaratılmıştır. Data dosyasındaki istenen bilgiye index dosyası yardımıyla çok kolay bir biçimde erişilmekte bu da KEY' ler vasıtasıyla olmaktadır. Ders ve Haftalık Ders Programına ait index dosyaları aşağıdaki gibidir .

DERS INDEX DOSYALARI :

MGUZDERS.INX
MBAHDERS.INX
FGUZDERS.INX
FBAHDERS.INX
KGUZDERS.INX
KBAHDERS.INX
IGUZDERS.INX
IBAHBERS.INX

PROGRAM INDEX DOSYALARI :

MGUZPROG.INX
MBAHPROG.INX
FGUZPROG.INX
FBAHPROG.INX
KGUZPROG.INX
KBAHPROG.INX
IGUZPROG.INX
IBAHPROG.INX

2.5 PROGRAMIN İŞLEYİŞİ :

Haftalık Ders Programı tabloları oluşturmak için, gerekli olan en önemli bilgi, atanacak derslerdir. Bunun için Ders İşlemleri Menüsünden bu derslerin, Bölüm ve Sömostirlerine göre dosyalanması (kayıtlarının yapılması) gerekir. Ancak bu dosyalanma işlemi bittikten sonra, atamalar yapılabilir.

Atamalara geçmeden önce, Haftalık Ders Programı tablolarının optimum sonuç vermesi için, daha önceden sunulan bazı kabul ve kısıtların dosyalanma işleminin yapılması gerektiğini KISITLAR ve KABULLER bölümünde belirtmiştik. Fakat bu, yine de kullanıcının isteğine bağlı olması istendiğinden, karar verici tarafından belirlenen veriler bir MENU (Haftalık Ders Programı Ön Bilgi Girişi) altında toplanmıştır. Eğer bu MENU altında toplanan Kısıt ve Kabuller (biri, birkaçı veya hepsi) atanmak istenmiyorsa doğrudan Ders Atamaları seçilebilir.

Haftalık Ders Programı Ön Bilgi Girişinde ilk adım olarak Yabancı Dil, Türkçe ve Tarih dersleri, daha önceden belirlenen gün ve saatlere atanır. Benzeri olarak, ikinci adımda, eğer isteniyorsa haftanın yarım yada tam gün boş kalması belirlenen gününe gerekli atamalar yapılabilir. Öğretim Görevlilerinin isteklerine bağlı atamalarda ise, eğer dersi yapmak istediği gün boşsa 2' şerli dağılımla veya 3 saat üstüste atamalarını yapılabilir.

Yukarıda açıklanan MENU altında yapılan işlemler bittikten sonra (eğer isteniyorsa yapılmayabilir) atamaların yapıldığı en önemli kısma geçilmektedir :

Hazırlanan örnek programda, atamalar 2 farklı şekilde yapılmaktadır. Bunlardan ilkinde, her sınıf için atanmayan dersi kadar farklı tablo oluşmaktadır. Oluşan bu tablolarda karar verici kişinin, O sınıfa göre uygun atanan tablolardan herhangi birisini seçmesi istenir. Karar verici kişi, uygun tabloyu seçene kadar daha önceden belirlemiş olduğu uygun olabilecek tablolardan istediğini ekranda belirledikten sonra, karar vermiş olduğu tabloyu dosyaya kaydetmesi için gerekli komutu verir. Her sınıf için bu işlem aynı biçimde tekrarlanır.

Yalnız burada dikkat edilmesi gereken bir durum şöyle ifade edilebilir : Özellikle 1. ve 4. sınıf derslerinin birbiriyle çakışması öğrenciler açısından herhangi bir problem yaratmadığı için, bu sınıflara ait atamaların önce yapılıp daha sonra bunlara bağlı olarak 2. ve 3. sınıfın atamalarının yapılması gerekir veya benzeri olarak önce 2. ve 3. sınıfın derslerinin atanmasından sonra 1. ve 4. sınıf derslerinin atanması yapılmalıdır.

Bu yöntemde derslerin atanması yapılırken daha önceden karar verici tarafından belirlenen kısıtların yanında, hazırlanan tabloların optimum sonuç vermesi için bazı kısıtlar da ilave edilmiştir.

Bunlardan en önemlilerinden biri, aynı Öğretim Görevlisinin farklı sınıflara aynı saatte ders yapamayacağıdır. Bir ikinci kısıtlama ise ders saati sayısı 4 den fazla olan derslerde bu dersin 3 gün üstüste yapılmamasıdır. Üçüncü kısıtlama olarak öğle saatlerine herhangi bir ders konmamasına da bilhassa dikkat edilmiştir. Bir diğer kısıtlama ise özellikle çakışmaların minimuma indirilmesi için atanacak dersin, eğer varsa bir önceki ve bir sonraki sınıfın boş olan hücrelerine atanmasına dikkat edilmiş veya en azından 2 saatinden bir tanesinin çakışmasına izin verilmiştir. Böyle atamaların olmaması için gerekli alt programlar hazırlanmıştır; fakat uygulanması istenen daha önceki kısıtlarda olduğu gibi, bu kısıtlarda da, kısıtlamanın getirdiği uygun düşmeyen şartlardan dolayı, bazı derslerin atanması yapılamamakta, bazı derslerinde ders saati atamalarında eksiklik olmaktadır. Kısaca bu durum, tamamıyla uygun şartı bulamadığı için atamanın gerçekleştirilememesinden başka bir şey değildir. Böyle durumlarda ortaya çıkan dengesizliği kaldırabilmek için, Haftalık Ders Programı Menüsünden, atanmayan dersleri veya saatleri, karar vericinin yada Öğretim Görevlisinin istediği yada uygun gördüğü bir güne istenen biçimde atanması yapılabilir.

Aynı Kısıtlar ve Kabuller esas alınıp, bir diğer yöntem olarak, kullanıcının isteğine bağlı olmaksızın, her sınıfın atanmayan dersi kadar farklı tablo 1. sınıfın derslerinden başlayarak atanmaktadır. Örneğin, herhangi bir bölüme ait 1. sınıfın 4 dersi, 2. sınıfın 5 dersi, 3. sınıfın 3 dersi ve 4. sınıfın da 5 dersi atanmamış olsun. Bu durumda

$4+5+3+5 = 17$ farklı tablo oluşacaktır. Karar verici oluşan bu tablolar arasından en uygununu seçebilir.

Kullanılan ikinci yöntemde, aslında $4 \times 5 \times 3 \times 5 = 300$ farklı tablo üretilebilir. Fakat bu uygun tablonun seçimini zorlaştıracığı gibi, karar vericinin de çok fazla zamanını alacaktır.

Örneği genelleştirecek olursak $(A+B+C+D)$ farklı tablonun arasından birkaçının optimuma yakın sonuçlar vermesi yapılan programı bu şekilde formüle etmeyi daha uygun göstermiştir.

Üzerinde durulması gereken bir konu da şudur:

Kısıtların ortaya çıkardığı sınırlamalardan dolayı atanması hiç yapılmayan derslerle, 1 veya 2 saati hiç atanmayan derslerin Haftalık Ders Programına ne şekilde yerleştirileceğidir. Bu duruma biraz olsun çözüm getirmesi açısından Haftalık Ders Programı üzerinde gerekli değişiklik, iptal ve gözlemin yapılabileceği alt programlar HAFTALIK DERS PROGRAMI menüsü başlığı altına toplanmıştır.

```

Procedure OpenIndex(var IdxF      : IndexFile);
Procedure AddRec(var   DatF      : DataFile;
                  RecNum      : Longint);
Procedure FindRec(var   DatF      : DataFile;
                  RecNum      : Longint);
Procedure DeleteRec(var   DatF      : DataFile;
                  RecNum      : Longint);
Procedure AddKey(var   IdxF      : IndexFile;
                  DataRecNum      : Longint;
                  ProcKey      : Longint);
Procedure FindKey(var   IdxF      : IndexFile;
                  DataRecNum      : Longint;
                  ProcKey      : Longint);
Procedure NextKey(var   IdxF      : IndexFile;
                  DataRecNum      : Longint;
                  ProcKey      : Longint);
Procedure PrevKey(var   IdxF      : IndexFile;
                  DataRecNum      : Longint;
                  ProcKey      : Longint);
Procedure SearchKey(var   IdxF      : IndexFile;
                  DataRecNum      : Longint;
                  ProcKey      : Longint);

```


2.6 EK 1:TURBO ACCESS SYSTEM (TOOLBOX) ERİŞİM MODÜLLERİ :

```

Procedure OpenFile(var  DatF      : DataFile;
                     FName      : FileName;
                     RecLen     : Word);

```

```

Procedure OpenIndex(var IdxF      : IndexFile;
                      FName      : FileName;
                      KeyLen,Dup : Byte);

```

```

Procedure MakeFile(var  DatF      : DataFile;
                     FName      : FileName;
                     RecLen     : Word);

```

```

Procedure MakeIndex(var IdxF      : IndexFile;
                     FName      : FileName;
                     KeyLen,Dup : Byte);

```

```

Procedure CloseFile(var DatF      : DataFile);

```

```

Procedure CloseIndex(var IdxF      : IndexFile);

```

```

Procedure Addrec(var  DatF      : DataFile;
                  RecNum      ,
                  Buffer      : Longint );

```

```

Procedure PutRec(var  Datf      : DataFile;
                  RecNum      ,
                  Buffer      : Longint );

```

```

Procedure DeleteRec(var DatF      : DataFile;
                    RecNum      : Longint);

```

```

Procedure AddKey(var  IdxF      : IndexFile;
                  DataRecNum ,
                  ProcKey     : Longint );

```

```

Procedure FindKey(var  IdxF      : IndexFile;
                  DataRecNum ,
                  ProcKey     : Longint );

```

```

Procedure NextKey(var  IdxF      : IndexFile;
                  DataRecNum ,
                  ProcKey     : Longint );

```

```

Procedure Prevkey(var  IdxF      : IndexFile;
                  DataRecNum ,
                  ProcKey     : Longint );

```

```

Procedure SearchKey(var IdxF      : IndexFile;
                    DataRecNum ,
                    ProcKey     : Longint );

```



```

Procedure DeleteKey(var IdxF      : IndexFile;
                    DataRecNum : LongInt;
                    ProcKey      );

```

```

Procedure ClearKey(var IdxF      : IndexFile);

```

```

Procedure EraseFile(var DatF      : DataFile);

```

```

Procedure EraseIndex(var IdxF      : IndexFile);

```

```

Procedure FlushFile(var DatF      : DataFile);

```

```

Procedure FlushIndex(var IdxF      : IndexFile);

```

```

Function FileLen(var DatF      : DataFile): LongInt;

```

Turbo Access System içinde kullanılan erişim modüllerinin isimleri, kullanılan parametreleri ile beraber yukarıdadır. Öncelikle, kullanılan parametrelerin anlamlarını açıklamaya çalışalım:

DatF : DataFile tipi değişken ismidir. Var başlığı altında tanımlanır.

FName : Dosya ismi değişkenidir. Var başlığı altında tanımlanır.

RecLen : Kayıt (Record) uzunluğudur. Genellikle SizeOf deyimi yardımıyla verilir.

IdxF : IndexFile tipi değişken ismidir. Var başlığı altında tanımlanır.

KeyLen : Anahtar bilginin uzunluğudur. Genellikle SizeOf deyimi yardımıyla verilir.

Dup : Const başlığı altında tanımlanabilen Byte tipi bir değişkendir. Kayıtların tekrarlı yada tekrarsız olacağını bildirir.

Recnum : Kayıt numarası değişkenidir. Var başlığı altında tanımlanan LongInt tipi değişkenidir.

Buffer : Record tipi değişken ismidir. Var başlığı altında tanımlanır.

DataRecNum : Kayıt numarası değişkenidir. Var başlığı altında LongInt tipte tanımlanır. Bu değişken isminin Drecnum değişken ismi ile aynı olmasında hiçbir sakınca yoktur.

ProcKey : Anahtar değişken ismidir. Var başlığı altında String tipte tanımlanmalıdır.

T.A. SYSTEM İÇİNDEKİ MODÜLLERİN GENEL KULLANIMLARI

Yukarıdaki bütün erişim modülleri TACCESS.PAS programı içinde tanımlanmıştır. Öncelikle şu bilinmelidir. Yukarıdaki her bir procedure genel parametrelerin dışında işlemin olumlu yada olumsuz bittiğini bildiren tek bir değişken kullanılır. Bu değişken TACCESS.PAS içinde kullanılmış olan OK isimli BOOLEAN tipi değişkendir. Eğer bu değişken TRUE değerine sahip ise işlemin olumlu, aksi halde olumsuz bittiğini bildirir.

İçinde Turbo Access System erişim modüllerinin kullanıldığı bir program derlenirken TACCESS.PAS programının hatasız derlenmesi için aşağıdaki sabitlerin TACCESS.PAS içinde tanımlanması gerekir.

Genelde bu sabitlerin değerleri SETCONST.PAS isimli bir program ile belirlenir. Ancak bu program kullanılmadan da bu sabitleri belirlemek mümkündür.

MaxDataRecSize : Açılacak dosyalar içinde kayıt uzunluğu en büyük olanın uzunluğudur.

Min = 14 Max = 65535 olabilir.

MaxKeyLen : Kullanılacak anahtar değişkenlerin karakter sayısı en büyük olanının uzunluğudur.

PageStackSize : Hafızadaki Max. PageSize sayısıdır.

Min = 3 Max = 254 olabilir.

PageSize : Bir sayfadaki max anahtar bilgi sayısıdır.

Min = 4 Max = 254 olabilir.

Order : PageSize/2 olarak hesaplanır.

MaxHeight : Index dosyasındaki dal sayısıdır.

Şimdi yukarıdaki erişim modüllerini açıklamaya çalışalım.

Kayıt Yapısı :

Type

Dersrec = Record

Dsay1 : Longint ;

Ddersno : String[2];

Ddersadı : String[15];

Ogrgor : String[25];

Teouyg : String[3];


```

Const
    MaxDataRecSize = 45;
    MaxKeyLen      = 3;
    PageSize       = 24;
    PageStackSize  = 10;
    Order          = 12;
    MaxHeight      = 4;

    Tekrarsızkayıt = 0;
    Tekrarlıkayıt  = 1;

Var
    Drec      : Dersrec;
    Dersdos   : DataFile;
    DersInx   : IndexFile;
    Drecnum   : LongInt;
    Dno       : String[2];
    Dadı      : String[15];
    Ogr       : String[25];
    Tu        : String[3];

```

OpenFile : DATA dosyasını açmak için kullanılır.

```
Openfile(Dersdos, 'Ders.Dat', Sizeof(Drec));
```

Eğer OK = TRUE ise dosya açılmıştır. Aksi halde 2 durum söz konusudur :

- 1-) Dosya disk üzerinde mevcut değildir.
- 2-) Kayıt uzunluğu ile MaxDataRecSize uyumsuzluğu vardır.

Ancak bu uyumsuzluk genellikle kayıt uzunluğunun MaxDataRecSize değerinden küçük oluşundan ileri gelir.

OpenIndex : INDEX dosyasını açmak için kullanılır.

```
OpenIndex(DersInx, 'Ders.Inx', Sizeof(Drec.Ddersno)-1,
    Tekrarsızkayıt);
```

Eğer OK = TRUE ise dosya açılmıştır. Aksi halde anahtar uzunluğu ile MaxKeyLen uyumsuzluğu vardır. Bu yine anahtar uzunluğunun MaxKeyLen değerinden küçük oluşundan ileri gelir.

MakeFile : DATA dosyasını yaratmak için kullanılır.

```
MakeFile(Dersdos, 'Ders.dat', Sizeof(Drec));
```

Eğer OK = TRUE ise dosya açılmıştır. Aksi halde kayıt uzunluğu ile MaxDataRecSize uyumsuzluğu vardır. Bu yine kayıt uzunluğunun MaxDataRecSize uzunluğundan küçük oluşundan gelir.

Turbo Access System'in getirdiği bir zorunluluk olarak her DATA dosyasının bir INDEX dosyası olmak zorundadır.

MakeIndex : INDEX dosyası yaratmak için kullanılır.

```
MakeIndex(DersInx, 'ders.inx', Sizeof(Drec.Ddersno)-1,
Tekrarsızkayıt);
```

Eğer OK = TRUE ise dosya yaratılmıştır. Aksi halde anahtar bilginin uzunluğu ile MaxKeyLen uyumsuzluğu vardır. Bu yine anahtar uzunluğunun MaxKeyLen değerinden küçük oluşundan ileri gelir.

CloseFile : DATA dosyasını kapamak için kullanılır.

```
CloseFile(DersDos);
```

CloseIndex : INDEX dosyasını kapamak için kullanılır.

```
Close(dersinx);
```

Kapama işlemlerinin sonucunda genellikle hata oluşmaz.

AddRec : DATA dosyasına kayıt ilave etmek için kullanılır.

```
AddRec(DersDos, Drecnum, Drec);
```

Bu işlemden önce Drec.Dsayı = 0 alınmalıdır. Bu işlem sonucunda yapılan kayıdın kayıt numarası Drecnum isimli değişkene gönderilir. Bu numara daha sonra indexleri kaydederken kullanılır.

GetRec : DATA dosyasından bir kaydı belleğe getirmek için kullanılır.

```
GetRec(DersDos, Drecnum, Drec);
```

Bu procedürü kullanmadan önce Drecnum değişkeni istenilen kayıdın numarasına sahip olacak şekilde değer almalıdır. Bu işlemden sonra Drec.Dsayı değişkeni kontrol edilmelidir. Eğer sıfırdan farklı ise kayıt iptal edilmiştir.

PutRec : Uzerinde değişiklik yapılan kaydı DATA dosyasına koymak için kullanılır.

```
PutRec(DersDos, Drecnum, Drec);
```

DeleteRec : DATA dosyasından bir kaydı silmek için kullanılır.

```
DeleteRec(DersDos, Drecnum);
```

Bu işlem ile Turbo Access System Drec.Dsayı değişkenine sıfırdan farklı bir değer göndererek dosyaya kaydeder.

AddKey : INDEX dosyasına bir anahtar bilgi ilave etmek için kullanılır.

AddKey(DersInx,Drecnum,Dno);

Bir DATA dosyasına ait eğer iki anahtar bilgi kullanılacaksa aynı anda işlem gerçekleştirilmelidir.

Bu procedure kullanılmadan önce AddRec procedure kullanılmalıdır.

FindKey : INDEX dosyasından bir kayda ait anahtar bilgiyi direkt olarak bulmak için kullanılır.

FindKey(DersInx,Drecnum,Dno);

Bu işlem sonucunda anahtar bilgi ya bulunur yada bulunmaz. Eğer bulunursa OK = TRUE değerine sahip olur. Aksi halde OK = FALSE olacaktır.

Tekrarlı INDEX dosyalarında, tekrar eden anahtar bilgilerden daima ilk kaydedilene ulaşılır.

NextKey : Eğer anahtar bilgi değişkeni boş ise daima ilk, aksi halde bir sonraki bilgiyi hafızaya getirir. Ancak NextKey procedure 'ü dosya sonuna gelene kadar işletilebilir. Dosya sonuna geldikten sonra bir defa daha NextKey işletilirse dosya başına geçilir. Ancak OK değişkeni ile dosya sonuna gelinip gelinmediği kontrol edilebilir.

NextKey(DersDos,Drecnum,Dno);

Eğer OK = FALSE ise dosya sonudur.

PrevKey : Eğer anahtar bilgi değişkeni boş ise son, aksi halde bir önceki anahtar bilgiyi hafızaya getirir. Ancak PrevKey porcedure' ü dosya başına gelene kadar işletilebilir. Dosya başına geldikten sonra bir defa daha PrevKey işletilirse dosya sonuna geçilir. Ancak OK değişkeni ile dosya başına gelinip gelinmediği kontrol edilebilir.

PrevKey(DersInx,Drecnum,Dno);

Eğer OK = FALSE ise dosya başıdır.

SearchKey : INDEX dosyasında baştan herhangi bir anahtar bilgiyi bulmak için kullanılır. Eğer verilen kısım ile başlayan anahtar bilgi yoksa bir sonraki anahtar bilgiye ulaşılır.

SearchKey(DersInx,Drecnum,Dno);

Eğer OK = FALSE ise istenilen anahtar bilgi bulunamamıştır.

DeleteKey : İNDEX dosyasından herhangi bir anahtar bilgiyi silmek için kullanılır.

DeleteKey(DersInx,Drecnum,Dno);

Değişen bir anahtar bilgiyi İNDEX dosyasına kayıt etmek için, ilk önce eski anahtar bilgi silinir; yerine yeni anahtar bilgi eklenir.

FindKey, NextKey, PrevKey, SearchKey, DeleteKey procedurelerin kullanılması sonucu elde edilen anahtar bilgilere ait kayıtların, kayıt numaraları Drecnum isimli değişkene gönderilir.

ClearKey : Herhangi bir İNDEX dosyasına ait, hafızada bulunan tüm anahtar bilgileri temizler.

ClearKey(DersInx);

FlushFile, FlushIndex : DATA ve İNDEX dosyalarına yapılacak kayıtların bu dosyalara anında kaydedilmesini sağlar; yani hafızadaki tampon bölgenin dolması beklenmez. Ancak bu procedure' ler AddRec ve AddKey procedure'lerinden sonra kullanılmalıdır.

FlushFile(DersDos);
FlushIndex(DersInx);

EraseFile : DATA dosyasına disk(et) üzerinden silmek için kullanılır.

EraseFile(DersDos);

EraseIndex : İNDEX dosyasını disk(et) üzerinden silmek için kullanılır.

EraseIndex(DersInx);

FileLen : Bir DATA dosyasındaki kayıt sayısını verir.

FileLen(DersDos);

3. B Ö L U M

BİLGİSAYAR PROGRAMI ve ÖRNEK ÇIKIŞLAR

[illegible]


```

NORMAL; DEKRAN; XY(6,40); CIZGI; DNO:=' ';
REPEAT
DEV:   XY(6,40); READLN(DNO);
      IF DNO='*' THEN EXIT;
      UNTIL LENGTH(DNO)=2;
      FINDKEY(DERSINX,DRECNUM,DNO);
      IF OK THEN BEGIN
        XY(6,45); REVERSE;
        WRITELN(' BU KAYIT MEVCUT [ ENTER ] '); NORMAL;
        EH:=READKEY; XY(6,40);
        WRITE(' ');
      GOTO DEV;   END
      ELSE BEGIN
        DREC.DDERSNO:=DNO;
        XY(8,40); READLN(DREC.DDERSADI);
        XY(10,40); READLN(DREC.OGRGOR);
        XY(12,40); READLN(DREC.TEOUYG);
        REVERSE;
        ONAY;
        NORMAL;
        IF UPCASE(EH)='H' THEN BEGIN
          REPEAT
            ARA_ISLEM; IF DNO='*' THEN EXIT;
            REVERSE; XY(16,9); BOSLUK; ONAY; NORMAL;
            UNTIL UPCASE(EH)='E';
          END;
          DREC.DSAYI:=0;
          ADDREC(DERSDOS,DRECNUM,DREC);
          ADDKEY(DERSINX,DRECNUM,DREC.DDERSNO); END;
          XY(16,9); BOSLUK;
          REVERSE; DEVAM; NORMAL;
        UNTIL UPCASE(EH)='H';
        DERSDKAPA;
        END;

PROCEDURE DGOZLEM;
BEGIN
CLS;
  BOLUM_SOR;
  SOMESTIR_SOR;
  BOLUM;
  DDOSAC;
REPEAT
CLS;
  CERCEVE(1,1,15,74);
  XY(1,4); WRITE(BL[H], ' BÖLÜMÜ');
  DEKRAN;
  REPEAT
    XY(1,47); REVERSE;
    WRITE(' &&& DERS NO <*> ILE MENU &&&' ); NORMAL;
    XY(6,40); READLN(DNO);
    IF DNO='*' THEN EXIT;
  UNTIL LENGTH(DNO)=2;
  FINDKEY(DERSINX,DRECNUM,DNO);
  IF OK THEN BEGIN
    GETREC(DERSDOS,DRECNUM,DREC);
    XY(8,40); WRITE(DREC.DDERSADI);

```



```

DERSDKAPA: XY(10,40); WRITE(DREC.OGRGOR);
END: XY(12,40); WRITE(DREC.TEOUYG);
END
PROCEDURE ELSE BEGIN
BEGIN XY(6,40); REVERSE;
CLS; WRITE(' BU NO ILE HERHANGI BIR KAYIT YOK ');
NORMAL;
END;
REVERSE; DEVAM; NORMAL
UNTIL UPCASE(EH)='H';
DERSDKAPA;
END;

PROCEDURE DDEGIS;
BEGIN
CLS;
BOLUM_SOR;
SOMESTIR_SOR;
BOLUM;
DDOSAC;
REPEAT
CLS;
CERCEVE(1,1,18,74);XY(1,4); WRITE(BL[H],' BÖLÜMÜ');
DEKRAN;
XY(1,47); REVERSE;
WRITE('DERS NO <*> ICIN M E N U'); NORMAL;
REPEAT
XY(6,40); CIZGI; XY(6,40); READLN(DNO);
IF DNO='*' THEN EXIT;
UNTIL LENGTH(DNO)=2;
FINDKEY(DERSINX,DRECNUM,DNO);
IF NOT OK THEN BEGIN
XY(6,45); REVERSE;
WRITE('B U K A Y I T Y O K');
NORMAL; END
ELSE BEGIN
GETREC(DERSDOS,DRECNUM,DREC);
XY(8,40); WRITE(DREC.DDERSADI);
XY(10,40); WRITE(DREC.OGRGOR);
XY(12,40); WRITE(DREC.TEOUYG);
XY(16,16); REVERSE;
WRITE('DEĞİŞİKLİK İÇİN HERHANGİ BİR TUŞA BASINIZ !!!');
NORMAL; EH:=READKEY;
REPEAT
ARA_ISLEM;
XY(16,10); BOSLUK;
REVERSE; ONAY; NORMAL;
UNTIL UPCASE(EH)='E';
PUTREC(DERSDOS,DRECNUM,DREC); END;
IF (DNO<>DREC.DDERSNO) AND (LENGTH(DNO)<>0) THEN
BEGIN
DELETEKEY(DERSINX,DRECNUM,DREC.DDERSNO);
ADDKEY(DERSINX,DRECNUM,DREC.DDERSNO);
END;
XY(16,9); BOSLUK;
REVERSE; DEVAM; NORMAL;
UNTIL UPCASE(EH)='H';

```



```

DERSDKAPA;
END;

PROCEDURE DIPTAL;
BEGIN
CLS;
  BOLUM_SOR;
  SOMESTIR_SOR;
  BOLUM;
  DDOSAC;
REPEAT
  CLS;
  CERCEVE(1,1,18,74);
  XY(1,4); WRITE(BL[H], ' BÖLÜMÜ');
  XY(1,2); WRITE(BL[H], ' BÖLÜMÜ');
DEKRAN;
  XY(1,35); REVERSE;
  WRITE('PROGRAMDAN CIKMAK ICIN NUMARA YERINE <*>');
  NORMAL;
  REPEAT
  XY(6,40); CIZGI; XY(6,40); READLN(DNO);
  IF DNO='*' THEN EXIT;
  UNTIL LENGTH(DNO)=2;
  FINDKEY(DERSINX,DRECNUM,DNO);
  IF NOT OK THEN BEGIN
  XY(6,40);REVERSE;WRITE('B U K A Y I T Y O K');
  NORMAL; END
  ELSE BEGIN
  GETREC(DERSDOS,DRECNUM,DREC);
  XY(8,40); WRITE(DREC.DDERSADI);
  XY(10,40); WRITE(DREC.OGRGOR);
  XY(12,40); WRITE(DREC.TEOUYG);
  XY(23,17); REVERSE;
  WRITE('IPTAL ETMEK ISTIYOR MUSUNUZ [E/H] ');
  NORMAL;
  EH:=READKEY;
  IF UPCASE(EH)='E' THEN BEGIN
  DELETEKEY(DERSINX,DRECNUM,DNO);
  DELETETREC(DERSDOS,DRECNUM);

  XY(22,4); BOSLUK;
  XY(15,24); HIGHVIDEO; WRITE('#####');
  XY(16,24); HIGHVIDEO; WRITE('# GEREKLİ İPTAL YAPILDI #');
  XY(17,24); HIGHVIDEO; WRITE('#####');
  END;
  END;
  REVERSE; DEVAM; NORMAL;
  UNTIL UPCASE(EH)='H';
  DERSDKAPA;
END;

PROCEDURE E_LIS;
LABEL DBA;
BEGIN
CLS;
L:=FILELEN(DERSDOS)-1;
CLS; DEBASLIK;

```



```

XY(24, FOR I:=1 TO L DO
WRITE BEGIN
CLS; GETREC(DERSDOS,I,DREC);
IF DREC.DSAYI=0 THEN
BEGIN
XY(T+5,4); WRITE(DREC.DDERSNO);
XY(T+5,20); WRITE(DREC.DDERSADI);
UNTIL XY(T+5,44); WRITE(DREC.TEOUYG);
XY(T+5,59); WRITE(DREC.OGRGOR);
T:=T+1;
END IF T=17 THEN
BEGIN
XY(24,20); REVERSE;
WRITE(' *** BIR TUSA BASINIZ *** '); NORMAL;
EH:=READKEY; CLS; T:=1;
DEBASLIK; GOTO DBA;
END;
END;
DBA: END;
END;

PROCEDURE P_LIS;
BEGIN
CLS;
L:=FILELEN(DERSDOS)-1;
DPBASLIK;
WRITELN(LST,' ');
FOR I:=1 TO L DO
BEGIN
GETREC(DERSDOS,I,DREC);
IF DREC.DSAYI=0 THEN
BEGIN
WRITE(LST,' ',DREC.DDERSNO:2,DREC.DDERSADI:22);
WRITELN(LST,DREC.TEOUYG:15,DREC.OGRGOR:33);
END;
END;
END;

PROCEDURE LISTELEME;
BEGIN
CLS;
BOLUM_SOR;
SOMESTIR_SOR;
BOLUM;
DDOSAC;
IF FILELEN(DERSDOS)-1=0 THEN BEGIN
KAYIT_YOK; EXIT; END
ELSE
BEGIN
REPEAT
T:=1; CLS; XY(10,5);
WRITE('TERCİHİNİZ : EKRAN (E) PRINTER (P)');
REPEAT
EH:=READKEY;
UNTIL EH IN['E','P'];
IF UPCASE(EH)='E' THEN E_LIS
ELSE P_LIS;

```



```

XY(24,20); REVERSE;
WRITE(' *** BİR TUŞA BASINIZ *** '); NORMAL; EH:=READKEY;
CLS; XY(10,10);
WRITE('LİSTEYİ YENİDEN GÖRMEK İSTER MİSİNİZ [E/H] ');
  REPEAT
    EH:=READKEY;
    UNTIL EH IN ['E','H'];
  UNTIL UPCASE(EH)='H';
  END;
  DERSDKAPA;
END;

PROCEDURE BOSLUK_ATA;
BEGIN
  FOR SINIF:=1 TO 4 DO
    FOR GUN:=1 TO 5 DO
      FOR SAAT:=1 TO 9 DO BEGIN
        DPROG[SINIF,GUN,SAAT]:='';
      END;
    END;
  END;

PROCEDURE TABLO;
LABEL TEK;
BEGIN
  CLS;
  IF BIL='BAH' THEN BIL:='BAHAR';
  XY(2,21); REVERSE;
  WRITE(' Y I L D I Z   Ü N İ V E R S İ T E S İ ');
  NORMAL;
  XY(3,29); WRITE('FEN EDEBİYAT FAKÜLTESİ');XY(4,15);
  WRITE(YIL,' ',BIL,' YARIYILI HAFTALIK DERS PROGRAMI');
  XY(4,62); WRITE(BL[H],' BÖLÜMÜ');
  CERCEVE(5,1,25,79);
  FOR J:=1 TO 9 DO BEGIN
    FOR I:=7 TO 78 DO BEGIN
      XY(5+J*2,I); WRITE('D');
    END;
  END;
  J:=0;
  FOR I:=1 TO 19 DO BEGIN
    TEK:  XY(5+I,15+J); WRITE('3');
          XY(6,21+J); WRITE(T,' .YY');
          J:=J+16; T:=T+2;
          IF J<63 THEN GOTO TEK; T:=1;
          J:=0;
        END;
        XY(6,7); WRITE('SAATLER');
        FOR I:=1 TO 19 DO BEGIN
          XY(5+I,6); WRITE('3'); END;
          S:=0;
          FOR I:=9 TO 17 DO BEGIN
            XY(8+S,7); WRITE(I,'-',I,'.50');
            S:=S+2; END;
            S:=0; J:=0; XY(16,31); REVERSE;
  WRITE(' Ö Ğ L E   T A T İ L İ '); NORMAL;
  END;
  PROCEDURE YIL_SOR;

```



```

BEGIN
  CLS;
  XY(22,16);
  WRITE(' *** YILI GİRİNİZ [YYYY-YYYY] *** ');
  READLN(YIL);
  CLS;
  END;

PROCEDURE E_TAB_GUN_YAZ;
BEGIN
  CLS;
  T:=0; Z:=0;
  IF BIL='GUZ' THEN T:=1 ELSE T:=2;
  FOR GUN:=1 TO 5 DO
    BEGIN
      CLS;
      TABLO;
      L:=LENGTH(GN[GUN]); B:=0; S:=0;
      FOR I:=1 TO L DO
        BEGIN
          G:=COPY(GN[GUN],I,1);
          XY(5+I*2,3); WRITE(G); T 1 L 1;
        END;
      END;
  FOR SINIF:=1 TO 4 DO
    BEGIN
      FOR SAAT:=1 TO 9 DO
        BEGIN
          XY(8+S,16+B);
          IF DPROG[SINIF,GUN,SAAT]<>' ' THEN
            WRITE(DPROG[SINIF,GUN,SAAT]);
            S:=S+2;
          END;
          S:=0;
          B:=B+16;
        END;
      IF GUN<5 THEN BEGIN
        XY(1,5); REVERSE;
        WRITE(' NOT: BİR SONRAKİ GÜN İÇİN [ENTER] ');
        NORMAL;
        EH:=READKEY; XY(16,17); BOSLUK; END;
        END;
        EH:=READKEY; THEN BEGIN
      END;

PROCEDURE P_TAB_GUN_YAZ;
LABEL 1;
BEGIN
  IF DS<>100 THEN BEGIN
    BOSLUK_ATA;
    DOS_BIL_OKU_ATA;
    END;

    J:=0;
  FOR GUN:=1 TO 5 DO BEGIN
    CLS; BOS;
    Z:=0;
    IF BIL='BAH' THEN BIL:='BAHAR';

```



```

WRITELN(LST,' ');
WRITELN(LST,' Y I L D I Z   U N İ V E R S İ T E S İ ');
WRITELN(LST,' ');
WRITE(LST,' FEN EDEBİYAT F. ');
WRITE(LST,YIL,' ',BIL,' YY. HAFTALIK DERS PROGRAMI ');
WRITELN(LST,BL[H],' BÖLÜMÜ');
WRITELN(LST,' '); WRITE(LST,'Z');
CIZ; WRITELN(LST,'G');
WRITE(LST,'3 3SAATLER 3 ');
FOR I:=1 TO 3 DO BEGIN
    WRITE(LST,(2*I-1),'. YY 3 ');
    END;
    WRITELN(LST,'7.YY 3');
    J:=1; DNO:='09'; S:=9;
FOR SAAT:=1 TO 9 DO BEGIN
    CIZ1;
    G:=COPY(GN[GUN],J,1); IF G<>' ' THEN DE[J]:=G;
    WRITE(LST,'3',DE[J],'3',DNO,'-',DNO,'.50');
    IF SAAT=5 THEN
        BEGIN
            WRITE(LST,'3 ');
            WRITELN(LST,'Ö Ğ L E T A T İ L İ ');
            GOTO 1; END;
        FOR SINIF:= 1 TO 4 DO
            BEGIN
                WRITE(LST,'3',DPROG[SINIF,GUN,SAAT]:16);
                END;
                WRITELN(LST,'3');
1:      S:=S+1; STR(S,DNO); J:=J+1
        END;
        WRITE(LST,'U'); CIZ; WRITELN(LST,'Y');
    END;
END;

PROCEDURE PROGRAMDOS_KAYIT;
BEGIN
    CLS;
    XY(15,15); REVERSE;
    WRITE('*** * DOSYAYA BU BİLGİLERİ KAYDEDELİM Mİ * ***');
    NORMAL;
    EH:=READKEY;
    IF UPCASE(EH)='E' THEN BEGIN
        CLS;
        XY(2,5); WRITE('DERS KODU :');
        XY(3,5); WRITE('=====');
        XY(2,40); WRITE('DERS ADI :');
        XY(3,40); WRITE('=====');
        FOR SINIF:=1 TO 4 DO
            FOR GUN:=1 TO 5 DO
                FOR SAAT:=1 TO 9 DO BEGIN
                    IF DPROG[SINIF,GUN,SAAT]<>' ' THEN
                        BEGIN
                            STR(100*SINIF+10*GUN+SAAT,TU);
                            XY(2,25); WRITE(' '); XY(2,25); WRITE(TU);
                            XY(2,55); BOSLUK; XY(2,55);
                            WRITE(DPROG[SINIF,GUN,SAAT]);
                            FINDKEY(PROGINX,PRECNUM,TU);
                        END;
                END;
            END;
        END;
    END;
END;

```



```

IF OK THEN BEGIN
    XY(13,25); REVERSE;
    WRITE('BU KAYIT MEVCUT');
    NORMAL;
    XY(15,25); WRITE('BIR SONRAKI DERS iCin');
    XY(15,47); WRITE('[ENTER]'); EH:=READKEY;
    XY(13,25); BOSLUK;
    XY(15,25); BOSLUK; END
ELSE BEGIN
    PREC.PROGNO:=TU;
    PREC.PROGADI:=DPROG[SINIF,GUN,SAAT];
    PREC.PSAYI:=0;
    ADDREC(PROGDOS,PRECNUM,PREC);
    ADDKEY(PROGINX,PRECNUM,PREC.PROGNO);
    XY(15,25); WRITE('BIR SONRAKI DERS iCin');
    XY(15,47); WRITE('[ENTER]'); EH:=READKEY;
    END;
END;
END;
END;

```

```

PROCEDURE BOS_SAAATLER;
BEGIN
    CERCEVE(1,60,15,78); K:=0;
    B:=0;
    XY(2,62); WRITE(GN[GUN], ' GUNU');
    XY(4,62); REVERSE; WRITE(' BOŞ SAAATLER '); NORMAL;
    XY(5,62); WRITE('=====');
    FOR SAAT:=1 TO 9 DO

```

```

        BEGIN
            IF (SAAT<5) OR (SAAT>5) THEN
                BEGIN
                    STR(100*SINIF+10*GUN+SAAT,TU);
                    FINDKEY(PROGINX,PRECNUM,TU);
                    IF NOT OK THEN BEGIN
                        XY(6+K,62); WRITE(SAAT, ' .SAAT');
                        K:=K+1;
                    END;
                END;
            END;
        END;
END;

```

```

PROCEDURE B_BIRAK;
BEGIN
    XY(2,62); WRITE(' ');
    FOR I:=1 TO 8 DO BEGIN
        XY(5+I,62); WRITE(' ');
    END;
END;

```

```

PROCEDURE OGR_GOR_ISTEK;
LABEL 101,100,110;
BEGIN
    BOSLUK_ATA;
    CLS;
    BOLUM_SOR;

```



```

SOMESTIR_SOR;
BOLUM;
DDOSAC;
PDOSAC;
REPEAT
  CLS;
  J:=0;
  CERCEVE(1,5,24,60);
  XY(2,7); REVERSE; WRITE(' ',BL[H],' BÖLÜMÜ');
  NORMAL;
  XY(2,33);REVERSE;WRITE('DERS NO <*> İÇİN ÇIKIŞ');
  NORMAL;
  XY(4,19); WRITE('ÖĞRETİM GÖREVLİLERİ İSTEKLERİ');
  XY(5,19); WRITE('=====');
  XY(6,7); REVERSE; WRITE('DERS NO      : '); NORMAL;
  REPEAT
    XY(6,19); READLN(DNO);
    IF DNO='*' THEN GOTO 110;
  UNTIL LENGTH(DNO)=2;
  FINDKEY(DERSINX,DRECNUM,DNO);
  IF NOT OK THEN
    BEGIN
      XY(6,18); REVERSE;
      WRITE(' *** BU NO İLE KAYIT YOKTUR *** '); NORMAL;
    END
  ELSE BEGIN
    GETREC(DERSDOS,DRECNUM,DREC);
    VAL(COPY(DREC.TEOUYG,1,1),A,Z);
    VAL(COPY(DREC.TEOUYG,3,1),B,Z);
    DS:=A+B;
    VAL(COPY(DNO,1,1),SINIF,Z);
    CASE SINIF OF
      1,2: SINIF:=1;
      3,4: SINIF:=2;
      5,6: SINIF:=3;
      7,8: SINIF:=4;
    END;
    XY(8,6); WRITE('DERS ADI      : '); WRITE(DREC.DDERSADI);
    XY(8,46); WRITE('TEO/UYG      : '); WRITE(DREC.TEOUYG);
    Z:=5; GUNLER;
100:  XY(25,5); BOSLUK; WRITE('          ');
    B_BIRAK;
    XY(20,7); WRITE('HANGİ GÜN          : ');
    XY(20,33);
    REPEAT
      READLN(GUN);
    UNTIL GUN IN[1,2,3,4,5];
    BOS_SAAATLER;
    IF K=0 THEN BEGIN
      XY(7,62); REVERSE; WRITE(' BUGÜN BOŞ '); NORMAL;
      XY(9,62); REVERSE; WRITE(' SAAT YOK '); NORMAL;
      XY(11,62); REVERSE; WRITE(' E N T E R '); NORMAL;
      EH:=READKEY;
      B_BIRAK;
      GOTO 100; END;
    XY(25,5); WRITE('BOŞ SAATLER SİZE UYGUN DEĞİLSE ');
    WRITE('BİR BAŞKA GÜNÜ DE GÖRMEK İSTER MİSİNİZ ');

```



```

REPEAT
EH:= READKEY;
UNTIL EH IN['E','H'];
IF UPCASE(EH)='E' THEN GOTO 100;
101:
XY(25,5); BOSLUK; WRITE(' ');
XY(21,7); WRITE('HANGİ SAAT [1..4]-[6..9]: ');
XY(21,33); READLN(SAAT);
UNTIL SAAT IN[1,2,3,4,5,6,7,8,9];
IF SAAT=5 THEN BEGIN
XY(21,34); WRITE(' SAAT ÖĞLE TATİLİ [ENTER]');
EH:=READKEY;
XY(21,36); WRITE(' ');
GOTO 101;
END;
STR(100*SINIF+10*GUN+SAAT,TU);
FINDKEY(PROGINX,PRECNUM,TU);
IF OK THEN BEGIN
XY(25,5);REVERSE;WRITE('BU SAAT DOLUDUR <ENTER>');
NORMAL; EH:=READKEY;
XY(25,5); BOSLUK;
GOTO 101; END;
XY(25,5); WRITE('BU SAATİ ONAYLIYOR MUSUNUZ');
REPEAT
EH:=READKEY;
UNTIL EH IN['E','H'];
IF UPCASE(EH)='H' THEN GOTO 101;
XY(25,5); BOSLUK;
XY(22,7); WRITE('KAÇ SAAT ÜSTÜSTE : ');
XY(22,33); READLN(S);
FOR I:=SAAT TO (SAAT+S-1) DO
BEGIN
DPROG[SINIF,GUN,I]:=DREC.DDERSADI;
END;
B_BIRAK;
J:=J+S;
XY(23,7);WRITE(DS,'DERSİN ',J,' TANESİ İŞLENDİ');
IF J<DS THEN GOTO 100;
J:=0; XY(25,5);
WRITE('* KAYIT İÇİN BİR TUŞA BASINIZ *** ');
EH:=READKEY;
PROGRAMDOS_KAYIT;
END;
XY(24,20); WRITE('< > İÇİN YENİ DERS');
REVERSE; XY(24,22); WRITE('R E T U R N'); NORMAL;
XY(24,50); EH:=READKEY;
UNTIL EH<>#13;
110: Z:=0;
PROGDKAPA;
DERSDKAPA;
END;

PROCEDURE BASLIK_TRH_DIL;
LABEL YENI1,YENI2;
BEGIN
CLS; Z:=0;
CERCEVE(1,5,23,61);

```



```

XY(2,23); REVERSE;
WRITE(' ',BL[H], ' BÖLÜMÜ '); NORMAL;
XY(3,12); REVERSE;
WRITE(TDIL, ' DERSİ İÇİN SINIFLARIN YERLEŞTİRİLMESİ');
NORMAL; XY(4,12);
WRITE('=====');
GUNLER;
XY(15,23); REVERSE; WRITE(TDIL, ' D E R S İ N İ');
NORMAL;
XY(17,10);WRITE(SINIF, '.SINIF İÇİN HANGİ GÜN:');
YENI1: XY(17,41); READ(GUN);
IF (GUN<1) OR (GUN>5) THEN GOTO YENI1;
XY(19,10); WRITE('HANGİ SAAT [1..4]-[6..9] ');
YENI2: XY(19,41); READ(SAAT);
IF (SAAT<1) OR (SAAT>8) THEN GOTO YENI2;
IF SAAT=5 THEN BEGIN
XY(19,43); WRITE('ÖĞLE TATİLİ [ENTER]');
EH:=READKEY;
XY(19,41);WRITE(' ');
GOTO YENI2;
END;
END;

PROCEDURE SABIT;
LABEL SAATT,SAATY,88,89,SA;
BEGIN
BOSLUK_ATA;
CLS;
BOLUM_SOR;
SOMESTIR_SOR;
BOLUM;
PDOSAC;
TDIL:='TARİH';
FOR SINIF:=1 TO 4 DO
BEGIN
IF (BIL='BAH') AND (SINIF>1) THEN GOTO 88;
SAATT: BASLIK_TRH_DIL;
XY(24,15); REVERSE;
WRITE(' BU BİLGİLERİ ONAYLIYOR MUSUNUZ [E/H] ');
NORMAL; EH_READ;
IF UPCASE(EH)='H' THEN GOTO SAATT;
FOR I:=SAAT TO SAAT+1 DO
DPROG[SINIF,GUN,I]:='TARİH';
END;
88: XY(24,15); BOSLUK;
XY(24,28);WRITE('* *YABANCI DİL DERSİ İÇİN* *');
XY(25,28); WRITE('* ****BİR TUŞA BASINIZ**** *');
EH:=READKEY;
TDIL:='Y.DİL';
FOR SINIF:=1 TO 2 DO
BEGIN
IF (BIL='BAH') AND (SINIF>1) THEN GOTO 89;
J:=1;
SAATY: BASLIK_TRH_DIL;
XY(21,10); WRITE('KAC SAAT USTUSTE ');
XY(21,41); READLN(S);
XY(24,15); REVERSE;

```



```

        WRITE(' BU BİLGİLERİ ONAYLIYOR MUSUNUZ [E/H] ');
NORMAL; EH_READ;
IF UPCASE(EH)='H' THEN GOTO SAATY;
FOR I:=SAAT TO (SAAT+S-1) DO
    DPROG[SINIF,GUN,I]:='Y.DİL';
J:=J+S;
IF J<(8/SINIF) THEN GOTO SAATY;
    END;
89:     XY(24,15); BOSLUK;
        S:=0; J:=0;
        XY(25,30);WRITE('* * HERHANGİ BİR TUŞA BASINIZ * *');
        EH:=READKEY; CLS;
IF BIL='BAH' THEN BEGIN
CLS; XY(14,10);
WRITE('TÜRKÇE DERSİNİN YERLEŞTİRİLMESİNİ İSTİYOR MUSUNUZ');
XY(16,23); WRITE('İSTİYORSANIZ [E] AKŞI HALDE [H]');
EH:=READKEY;
REPEAT
    UNTIL EH IN['E','H'];
    IF EH='E' THEN BEGIN
        TDIL:='TÜRKÇE'; SINIF:=1; BASLIK_TRH_DIL;
SA: XY(21,10); WRITE('KAC SAAT ÜSTÜSTE : '); READLN(S);
        XY(24,15); REVERSE;
        WRITE('BU BİLGİLERİ ONAYLIYOR MUSUNUZ [E/H] ');
        REPEAT
            EH:=READKEY;
            UNTIL EH IN['E','H'];
            IF UPCASE(EH)='H' THEN GOTO SA;
            FOR I:=SAAT TO (SAAT+S-1) DO
                DPROG[SINIF,GUN,I]:='TÜRKÇE';
        END;
    END;
NORMAL;
YIL_SOR;
CLS; XY(10,5);
WRITE('# ATAMALARINIZI GÖRMEK İSTİYOR MUSUNUZ [E/H] #');
REPEAT
    EH:=READKEY;
    UNTIL EH IN['E','H'];
IF UPCASE(EH)='E' THEN E_TAB_GUN_YAZ;
PROGRAMDOS_KAYIT;
PROGDKAPA;
END;

PROCEDURE GUN_BOSALT;
LABEL 1001,1002;
BEGIN
    BOSLUK_ATA;
    BOLUM_SOR;
    SOMESTIR_SOR;
    BOLUM;
    PDOSAC;
1001:  CLS;
        CERCEVE(1,7,20,63); Z:=1;
        XY(2,19);REVERSE;WRITE(BL[H],' BÖLÜMÜ ');NORMAL;
        XY(4,17); REVERSE;
        WRITE('BELİRLİ GÜNLERİN BOŞ BIRAKILMASI');NORMAL;

```



```

GUNLER;
XY(16,10); WRITE('HANGİ SINIF [1-4] : ');
XY(16,32); READ(SINIF);
IF (SINIF<1) OR (SINIF>4) THEN GOTO 1001;
XY(17,10); WRITE('HANGİ GÜN [1-5] : '); XY(17,32);
READ(GUN);
XY(18,10); WRITE('HANGİ SAAT [1-9] : ');
XY(18,32); READ(SAAT);
XY(19,10); WRITE('KAÇ SAAT USTUŞTE : ');
XY(19,32); READ(S);
XY(21,7); REVERSE;
WRITE(' BU SINIF BİLGİSİ ONAY [E \ H] '); NORMAL;
EH:=READKEY;
IF UPCASE(EH)='H' THEN GOTO 1001;
FOR I:=SAAT TO (SAAT+S-1) DO
  IF I<>5 THEN DPROG[SINIF,GUN,I]:='*****';
  XY(21,7); REVERSE;
  WRITE(' BİR SONRAKİ SINIF İÇİN [E \ H] '); NORMAL;
  EH_READ;
  IF UPCASE(EH)='E' THEN GOTO 1001;
1002: CLS;
XY(1,18); REVERSE;
WRITE('BELİRLİ GÜNLERİN BOŞ BIRAKILMASI'); NORMAL;
CERCEVE(2,3,18,68);
XY(4,20); REVERSE; WRITE(,BL[H], 'BÖLÜMÜ'); NORMAL;
GUNLER; XY(16,6);
REVERSE; WRITE(' * * * BÜTÜN SINIFLARIN BOŞ KALMASINI ');
WRITE('İSTEDİĞİNİZ GÜN * * * '); NORMAL;
XY(16,65); READ(GUN);
XY(16,4); BOSLUK;
REVERSE; ONAY; NORMAL;
IF UPCASE(EH)='E' THEN BEGIN
  FOR SINIF:=1 TO 4 DO
    FOR SAAT:=6 TO 9 DO
      DPROG[SINIF,GUN,SAAT]:='*****'; END
  ELSE GOTO 1002;
YIL_SOR;
CLS; XY(10,5);
WRITE('# ATAMALARINIZI GÖRMEK İSTİYOR MUSUNUZ [E/H] #');
REPEAT
  EH:=READKEY;
  UNTIL EH IN['E','H'];
  IF UPCASE(EH)='E' THEN E_TAB_GUN_YAZ;
  PROGRAMDOS_KAYIT;
  PROGDKAPA;
END;

PROCEDURE SOMESTR_SIL;
BEGIN
CLS;
CERCEVE(5,5,18,75);
XY(7,10); REVERSE; WRITE(' AÇIKLAMA: '); NORMAL;
XY(9,10); WRITE('BİR ÖNCEKİ SÖMESTİRE AİT BİLGİLERİN ');
WRITE('SİLİNMESİNİ İSTEMİYORSANIZ HAFTALIK DERS PROGRAMI');
WRITE('BİLGİ GİRİŞİ YAPMAYINIZ'); XY(13,9);
WRITE('(4) NUMARALI SEÇENECTEN İSTEDİĞİNİZ İŞLEMLERİ');
WRITE('SEÇİNİZ');

```



```

XY(16,9); REVERSE;
WRITE('SÖMESTİRE AİT DERS PROGRAMI SİLİNSİN Mİ [E/H]');
NORMAL;
XY(16,67); REPEAT
EH:=READKEY; UNTIL EH IN['E','H'];
IF UPCASE(EH)='E' THEN
    BEGIN
        PDOS_SIL; THEN OGR:=(DREC.OGRGOR);
        END;
CLS;
XY(15,9); REVERSE;
WRITE('SÖMESTİRE AİT DERS BİLGİLERİ SİLİNSİN Mİ [E/H]');
NORMAL;
REPEAT
    XY(15,60); EH:=READKEY;
    UNTIL EH IN['E','H'];
    IF UPCASE(EH)='E' THEN BEGIN
        DDOS_SIL;
        END
    ELSE EXIT;
END;

PROCEDURE DERS_BUL2;
LABEL DA;
BEGIN
    S:=FILELEN(DERSDOS)-1;
    HOCA:='AAA'; OGR:='BBB';
    FOR Y:=1 TO 4 DO
        BEGIN
            IF SNF<>Y THEN
                BEGIN
                    PADI:=DPROG[Y,GUN,SAAT];
                    DADI:=DPROG[Y,GUN,SAAT+1];
                    IF (PADI=' ') AND (DADI=' ') THEN GOTO DA;
                    FOR A:=1 TO S DO
                        BEGIN
                            GETREC(DERSDOS,A,DREC);
                            IF DREC.DSAYI=0 THEN
                                BEGIN
                                    IF (DREC.DDERSADI)=DE[I] THEN OGR:=(DREC.OGRGOR);
                                    IF (DREC.DDERSADI=PADI) OR (DREC.DDERSADI=DADI)
                                        THEN HOCA:=(DREC.OGRGOR);
                                END;
                            END;
                        END;
                END;
            DA: END;
        END;

PROCEDURE DERS_BUL1;
LABEL DAA;
BEGIN
    S:=FILELEN(DERSDOS)-1;
    HOCA:='AAA'; OGR:='BBB';
    FOR Y:=1 TO 4 DO
        BEGIN
            IF SNF<>Y THEN
                BEGIN

```



```

PADI:=DPROG[Y,GUN,SAAT];
BEGIN IF (PADI=' ') THEN GOTO DAA;
FOR A:=1 TO S DO
  BEGIN
    GETREC(DERSDOS,A,DREC);
    IF DREC.DSAYI=0 THEN
      BEGIN
        IF (DREC.DDERSADI)=DE[I] THEN OGR:=(DREC.OGRGOR);
        IF (DREC.DDERSADI=PADI)
          THEN HOCA:=(DREC.OGRGOR);
      END;
    END;
  END;
END;
DAA: END;
END;

```

```

PROCEDURE KISIM;

```

```

LABEL 1;

```

```

BEGIN

```

```

  K:=0;

```

```

  FOR P:=1 TO 6 DO

```

```

    BEGIN

```

```

      DY[P]:=' ';

```

```

      SY[P]:=0; END;

```

```

      S:=FILELEN(DERSDOS)-1; L:=FILELEN(PROGDOS)-1;

```

```

      FOR I:=1 TO S DO

```

```

        BEGIN

```

```

          GETREC(DERSDOS,I,DREC);

```

```

          IF DREC.DSAYI=0 THEN

```

```

            BEGIN

```

```

              DNO:=DREC.DDERSNO;

```

```

              VAL(COPY(DNO,1,1),SEC,Z);

```

```

              IF SEC=SEC1 THEN

```

```

                BEGIN

```

```

                  DADI:=DREC.DDERSADI;

```

```

                  FOR J:=1 TO L DO

```

```

                    BEGIN

```

```

                      GETREC(PROGDOS,J,PREC);

```

```

                      IF PREC.PSAYI=0 THEN

```

```

                        BEGIN

```

```

                          PADI:=PREC.PROGADI;

```

```

                          IF DADI=PADI THEN GOTO 1;

```

```

                        END;

```

```

                    END;

```

```

                K:=K+1;

```

```

                VAL(COPY(DREC.TEOUYG,1,1),A,Z);

```

```

                VAL(COPY(DREC.TEOUYG,3,1),B,Z);

```

```

                SS[K]:=A+B; SY[K]:=A+B;

```

```

                DE[K]:=DADI; DY[K]:=DADI;

```

```

                IF O<>10 THEN BEGIN

```

```

                  XY(13+K,19); WRITE(DNO);

```

```

                  XY(13+K,41); WRITE(DE[K], '(',SS[K],')');

```

```

                END;

```

```

            END;

```

```

        END;

```

```

  1:END;

```

```

END;

```



```

PROCEDURE ATANM_DERS_YAZ;
BEGIN
O:=1;
KISIM;
IF K=0 THEN BEGIN
XY(23,4);
WRITE(' !!! BU SINIFA AIT BUTUN DERSLER ATANMIŞTIR ');
WRITE(' [ E N T E R ] !!! '); EH:=READKEY;
XY(23,4); BOSLUK;
END;
END;

PROCEDURE ATAMA_BASLIGI;
LABEL 1002;
BEGIN
SIFIR_ATA;
BOSLUK_ATA;
DOS_BIL_OKU_ATA;
1002:CLS;
CERCEVE(4,10,21,59);
XY(1,50); REVERSE;
WRITE(' SINIF <0> İÇİN ÇIKIŞ '); NORMAL;
XY(3,9); REVERSE;
WRITE(' HANGİ SINIF RASTGELE YERLEŞTİRİLECEK [1/4] ');
READ(SNF); NORMAL;
IF SNF=0 THEN EXIT;
XY(3,9); BOSLUK;
XY(23,10); REVERSE;
WRITE(' ATANACAK DERSLERİN YÜKLEMESİ YAPILIYOR ');
NORMAL; Y(6,28); REVERSE;
WRITE(' ',BL[1], ' BÖLÜMÜ '); NORMAL;
XY(8,12); REVERSE;
WRITE(SNF, ' . SINIF DERSLERİNİN YERLEŞTİRİLMESİ '); NORMAL;
XY(9,12); REVERSE;
WRITE(' ***** '); NORMAL;
XY(10,23); WRITE(' HENÜZ ATANMAYAN DERSLER ');
XY(11,23); WRITE(' ===== ');
XY(12,15); WRITE(' DERS NO ');
XY(12,42); WRITE(' DERS ADI ');
IF BIL='GUZ' THEN SEC1:=(2*SNF-1)
ELSE SEC1:=(2*SNF);
Z:=0;K:=1;
S:=FILELEN(DERSDOS)-1; L:=FILELEN(PROGDOS)-1;
R:=0;
ATANM_DERS_YAZ;
XY(23,10); BOSLUK;
IF K=0 THEN GOTO 1002;
XY(20,17); REVERSE;
WRITE(' DERS ATAMALARI İÇİN [ E N T E R ] '); NORMAL;
EH:=READKEY;
END;

PROCEDURE DIL_TAR_BOS_ATA2;
BEGIN
FOR GUN:=1 TO 5 DO
FOR SAAT:=1 TO 9 DO
BEGIN

```



```

PROCEDURE OKU;
LABEL 1;
IF ((DPROG[SNF-1,GUN,SAAT]='Y.DİL')
OR (DPROG[SNF-1,GUN,SAAT]='TARİH')
OR (DPROG[SNF-1,GUN,SAAT]='*****'))
OR (DPROG[SNF-1,GUN,SAAT]='*****'))
THEN DPROG[SNF-1,GUN,SAAT]:=' ';
IF ((DPROG[SNF+1,GUN,SAAT]='Y.DİL')
OR (DPROG[SNF+1,GUN,SAAT]='TARİH')
OR (DPROG[SNF+1,GUN,SAAT]='*****'))
THEN DPROG[SNF+1,GUN,SAAT]:=' ';
END;
END;

PROCEDURE SAKLA;
BEGIN
CLS;
FOR P:=1 TO K DO
BEGIN
DE[P]:=DY[P];
SS[P]:=SY[P];
END;
END;

PROCEDURE YER_DEGISTIR;
BEGIN
DADI:=DE[1]; DS:=SS[1];
FOR P:=1 TO K-1 DO
BEGIN
DE[P]:=DE[P+1];
SS[P]:=SS[P+1];
END;
DE[K]:=DADI; SS[K]:=DS;
END;

PROCEDURE TUT;
BEGIN
IF NO[R]>1 THEN BEGIN
FOR B:=1 TO (NO[R]-1) DO
BEGIN
YER_DEGISTIR;
END;
END;
END;

PROCEDURE OKU;
BEGIN
XY(5,5); REVERSE;
WRITE(' BU KISMI DİKKATLİCE OKUYUNUZ '); NORMAL;
XY(6,5); WRITE('=====');XY(10,4);
WRITE(K,' DERS İÇİN ',K,' FARKLI ATAMA YAPILABİLİR. ');
WRITE('YAPILAN BU ATAMALARDAN İSTEDİKLERİNİZİN');XY(12,1);
WRITE('SAKLANMASI YAPILABİLMEKTEDİR.BUNUN İÇİN YAPILAN');
WRITE('ATAMALAR İÇİNDE SİZE UYGUN'); XY(14,1);
WRITE('GELENLERİNİ TEKRAR İNCELEMEK İSTERSENİZ UYARI ');
WRITE('MESAJI GELDİĞİNDE BELLEKTE SAKLAN-');
XY(16,1); WRITE('MASI İÇİN GEREKLİ BİLGİYİ VERİNİZ. ');
XY(20,5); REVERSE; WRITE(' E N T E R '); NORMAL;
EH:=READKEY; M:=0; END;

```



```

PROCEDURE DERS_ATAMA1;
LABEL 9,10,11,20,30,35,110,113,
31,59,60,61,40,45,1003,111,112;
BEGIN
    SIFIR_ATA;
    BOSLUK_ATA;
    CLS;
    BOLUM_SOR;
    SOMESTIR_SOR;
    DDOSAC;
    PDOSAC;
    BOLUM;
    IF FILELEN(DERSDOS)-1=0 THEN KAYIT_YOK
ELSE
    DOS_BIL_OKU_ATA;
    YIL_SOR;
    IF FILELEN(DERSDOS)-1<>0 THEN
    BEGIN
        CLS; CERCEVE(6,3,14,69);
        XY(8,5); REVERSE; WRITE('DİKKAT : '); NORMAL; XY(10,5);
        WRITE('ŞU ANDA BİLGİ OKUTULMAKTADIR HAZIRSANIZ ŞİMDİYE');
        XY(12,5);
        WRITE('KADAR YAPILAN ATAMALARI GÖRMEK ');
        WRITE('İÇİN HERHANGİ BİR TUŞA BASINIZ'); EH:=READKEY;
        E_TAB_GUN_YAZ; CLS; XY(10,10);
        WRITE(' BİR SONRAKİ İŞLEMLERE GEÇMEK İÇİN [ E N T E R ] ');
        EH:=READKEY;
        END;
        CLS;
        XY(22,12); REVERSE;
        WRITE('BU BÖLÜMDEN ÇIKMAK İÇİN SINIF YERİNE <0> GİRİN ');
        NORMAL;
        XY(10,1); REVERSE;
        WRITE('İSTEDİĞİNİZ SINIFLARA ATAMA YAPMAK İÇİN [1/2]');
        WRITE('PARAMETRELERİNİ KULLANIN '); NORMAL;
        XY(12,10); WRITE(' [ 1. & 4. sınıf için (1) ');
        WRITE(' 2. & 3. sınıf için (2) parametresini kullanınız');
        READ(A);
        IF A=0 THEN GOTO 1003;
        IF A=1 THEN GOTO 30 ELSE GOTO 31;
    30: CLS;
        XY(1,3); REVERSE;
        WRITE('&& BU BÖLÜMDE HERHANGİ BİR UYARI GELMEDİĞİ ');
        WRITE('SURECE 2. ve 3. SINIF &&'); XY(3,13); REVERSE;
        WRITE('&& DERSLERİNİN RASTGELE ATAMASINI YAPMAYINIZ &&');
        NORMAL;
        CERCEVE(5,5,18,65);
        XY(7,10); REVERSE; WRITE(' AÇIKLAMA : '); NORMAL;
        XY(9,10); WRITE('4. SINIFIN DERSLERİ İLE 1. SINIFIN DERSLERİ');
        WRITE(' RASTGELE VEYA ÇAKIŞACAK ŞEKİLDE ATANMIŞ OLABİLİR. ');
        XY(11,10); WRITE(' ONUN İÇİN HER İKİ SINIFIN DERSLERİNİ DE ');
        WRITE(' RASTGELE YERLEŞTİREBİLİRSİNİZ. '); XY(16,15);
        WRITE(' *:*:* İŞLEMLER İÇİN BİR TUŞA BASINIZ *:*:* ');
        EH:=READKEY;
        CLS; O:=0; N:=0; R:=0;
        ATAMA_BASLIĞI;
        IF SNF=0 THEN EXIT; O:=0;

```



```

SAYDIR; CLS;
OKU;
35: CLS;
FOR I:=1 TO K DO
BEGIN
Z:=0; S:=0; SEC:=10;
FOR GUN:=1 TO 5 DO
BEGIN
IF DER_SAY[GUN]>=3 THEN GOTO 11;
FOR SAAT:=1 TO 7 DO
BEGIN
IF (SEC=SAAT) OR (SAAT=SEC+1) THEN GOTO 9;
IF TRUNC((SS[I]-Z)/2)=(SS[I]-Z)/2 THEN BEGIN
IF (DPROG[SNF,GUN,SAAT]=' ') AND (SAAT<>5) AND
(DPROG[SNF,GUN,SAAT+1]=' ') AND (SAAT+1<>5)
THEN BEGIN
IF SNF=1 THEN Y:=-1 ELSE Y:=1;
DERS_BUL2;
IF HOCA=OGR THEN GOTO 9;
DPROG[SNF,GUN,SAAT]:=DE[I];
DPROG[SNF,GUN,SAAT+1]:=DE[I];
Z:=Z+2;
SEC:=SAAT; DER_SAY[GUN]:=DER_SAY[GUN]+1;
GOTO 10; END
ELSE GOTO 9;
END;
IF (DPROG[SNF,GUN,SAAT]=' ') AND (SAAT<>5)
THEN BEGIN
IF SNF=1 THEN Y:=-1 ELSE Y:=1;
DERS_BUL1;
IF HOCA=OGR THEN GOTO 9;
IF (DPROG[SNF-Y,GUN,SAAT]=' ') THEN
DPROG[SNF,GUN,SAAT]:=DE[I];
DER_SAY[GUN]:=DER_SAY[GUN]+1;
Z:=Z+1; SEC:=SAAT; P:=0; GOTO 10;
END;
9: END;
10: IF Z>=SS[I] THEN GOTO 20;
11: END;
20: CLS; END;
IF M=100 THEN GOTO 112;
CLS;
E_TAB_GUN_YAZ; CLS;
XY(10,3); WRITE(' BU ATAMALARI ONAYLIYORSANIZ : ');
WRITE(' DOSYAYA KAYIT İÇİN [E] ');
XY(12,3); WRITE(' EĞER ONAYLAMİYORSANIZ : ');
WRITE(' DERSLERİN YENİDEN ATANMASI İÇİN [H] ');
EH_READ;
IF UPCASE(EH)='H' THEN
BEGIN
SIFIR_ATA;
BOSLUK_ATA;
DOS_BIL_OKU_ATA;
SAYDIR;
YER_DEGISTIR;
CLS; N:=N+1;

```



```

XY(10,5);
WRITE(' *** BU ATAMALARI AKLIMDA TUTAYIM MI *** ');
EH:=READKEY;
IF UPCASE(EH)='E' THEN BEGIN
    O:=O+1; NO[O]:=N; END;
    IF N=K THEN
BEGIN
    CLS; XY(10,5); REVERSE;
    WRITE(' UYARI: '); NORMAL;
    WRITE(' ',N,' DERS İÇİN YERDEĞİŞTİRME İŞLEMİ BAŞA ');
    WRITE('DÖNMEKTEDİR ');
    XY(15,5); REVERSE; WRITE(' E N T E R '); NORMAL;
    EH:=READKEY;
    REPEAT
        CLS;
    IF O=0 THEN
        BEGIN
            XY(10,5);
            WRITE('HİÇBİR ATAMA SEÇMEDİNİZ BAŞA DÖNMEK İÇİN <0> ');
            EH:=READKEY; GOTO 30;
            END;
            XY(5,5); REVERSE;
            WRITE('ATMALARIN BAŞTAN YAPILMASINI ');
            WRITE('İSTİYORSANIZ <0> GİRİN');
            NORMAL; XY(12,5); R:=0;
            WRITE('& SEÇTİĞİNİZ ',O);
            WRITE('ATAMADAN HANGİSİNİ GÖRMEK İSTİYORSUNUZ &');
            READ(R);
            IF R=0 THEN GOTO 30;
            SIFIR_ATA;
            BOSLUK_ATA;
            DOS_BIL_OKU_ATA;
            SAYDIR;
            SAKLA;
            TUT; CLS;
            M:=100; GOTO 35;
112: CLS;
            E_TAB_GUN_YAZ; CLS; XY(10,3);
            WRITE(' BU ATAMALARI ONAYLIYORSANIZ : ');
            WRITE(' DOSYAYA KAYIT İÇİN [E] '); XY(12,3);
            WRITE(' EGER ONAYLAMİYORSANIZ : DiĞER');
            WRITE(' ATAMA İÇİN [H] '); EH:=READKEY;
            UNTIL UPCASE(EH)='E';
            GOTO 111; END;
            GOTO 35;
            END;
111: CLS;
            XY(23,5); REVERSE;
            WRITE('DiĞER SINIFIN DA ATAMASININ YAPILMASINI ');
            WRITE(' İSTİYOR MUSUNUZ [ 1 / 4 ] '); NORMAL;
            EH:=READKEY;
            IF UPCASE(EH)='E' THEN GOTO 30;
            CLS;
            XY(15,10); REVERSE;
            WRITE(' #&# 2. veya 3. sınıfın atamalar da yapılsın ');
            WRITE(' m1 [E/H] #&# '); NORMAL;

```



```

EH:=READKEY;
IF UPCASE(EH)='E' THEN BEGIN
31 :
    CLS; O:=0; N:=0; R:=0;
    ATAMA_BASLIGI;
    SAYDIR;
    CLS;
    OKU;
45:
    CLS;
    FOR I:=1 TO K DO
        BEGIN
            Z:=0; S:=0; SEC:=10;
            FOR GUN:=1 TO 5 DO
                BEGIN
                    IF (P=2) OR (P=3) THEN GOTO 60;
                    IF DER_SAY[GUN]>=3 THEN GOTO 61;
                    Y:=10;
                    FOR SAAT:=1 TO 7 DO
                        BEGIN
                            IF (SEC=SAAT) OR (SAAT=SEC+1) THEN GOTO 59;
                            IF TRUNC((SS[I]-Z)/2)=(SS[I]-Z)/2 THEN BEGIN
                                IF (DPROG[SNF,GUN,SAAT]=' ') AND (SAAT<>5)
                                    AND (DPROG[SNF,GUN,SAAT+1]=' ') AND (SAAT+1<>5)
                                    THEN BEGIN
                                        IF I=Y THEN GOTO 59;
                                        DERS_BUL2;
                                        IF HOCA=OGR THEN GOTO 59;
                                    IF ((DPROG[SNF-1,GUN,SAAT]=' ')
                                        OR (DPROG[SNF-1,GUN,SAAT+1]=' ')
                                        AND (DPROG[SNF+1,GUN,SAAT]=' ')
                                        OR (DPROG[SNF+1,GUN,SAAT+1]=' '))
                                        THEN BEGIN
                                            DPROG[SNF,GUN,SAAT]:=DE[I];
                                            DPROG[SNF,GUN,SAAT+1]:=DE[I];
                                            Z:=Z+2; SEC:=SAAT; { P:=1; }
                                            DER_SAY[GUN]:=DER_SAY[GUN]+1;
                                        END
                                    ELSE GOTO 59;
                                END
                            ELSE GOTO 59;
                            END;
                        IF (DPROG[SNF,GUN,SAAT]=' ') AND (SAAT<>5)
                            THEN BEGIN
                                IF I=Y THEN GOTO 59;
                                DERS_BUL1;
                                IF HOCA=OGR THEN GOTO 59;
                            IF ((DPROG[SNF-1,GUN,SAAT]=' ')
                                OR (DPROG[SNF+1,GUN,SAAT]=' ')) THEN BEGIN
                                    DPROG[SNF,GUN,SAAT]:=DE[I];
                                    DER_SAY[GUN]:=DER_SAY[GUN]+1;
                                    Z:=Z+1; SEC:=SAAT;
                                GOTO 60;
                            END;
                        END;
                    END;
                END;
            END;
        59: END;
        60:

```



```

IF Z>=SS[1] THEN GOTO 40;
61: END;
40: CLS; END;
IF M=100 THEN GOTO 110;
E_TAB_GUN_YAZ;
CLS;
XY(10,3); WRITE(' BU ATAMALARI ONAYLIYORSANIZ : ');
WRITE(' DOSYAYA KAYIT İÇİN [E] ');
XY(12,3); WRITE(' EĞER ONAYLAMİYORSANIZ : ');
WRITE(' DERSLERİN YENİDEN ATANMASI İÇİN [H] ');
EH:=READKEY;
IF UPCASE(EH)='H' THEN
BEGIN
SIFIR_ATA;
DOS_BIL_OKU_ATA;
SAYDIR;
DIL_TAR_BOS_ATA2;
YER_DEGISTIR;
CLS;
N=N+1;
XY(10,5);WRITE('* BU ATAMALARI AKLIMDA TUTAYIM MI *');
EH:=READKEY;
IF UPCASE(EH)='E' THEN BEGIN
O:=O+1; NO[O]:=N; END;
IF N=K THEN
BEGIN
CLS; XY(10,5); REVERSE;
WRITE(' UYARI: '); NORMAL;
WRITE(' ',N,' DERS İÇİN YERDEĞİŞTİRME İŞLEMİ');
WRITE('BAŞA DÖNMEKTEDİR ');
XY(15,5); REVERSE; WRITE(' E N T E R '); NORMAL;
EH:=READKEY;
REPEAT
CLS;
IF O=0 THEN
BEGIN
XY(10,5); WRITE('HİÇBİR ATAMA SEÇMEDİNİZ');
WRITE('BAŞA DÖNMEK İÇİN <0> ');
EH:=READKEY; GOTO 31;
END;
CLS;
XY(12,5);
WRITE('&&& SEÇTİĞİNİZ ',O,' ATAMADAN HANGİSİNİ GÖRMEK');
WRITE(' İSTİYORSUNUZ &&& '); READ(R);
SIFIR_ATA;
BOSLUK_ATA;
DOS_BIL_OKU_ATA;
SAYDIR;
DIL_TAR_BOS_ATA2;
SAKLA;
TUT; CLS;
M:=100; GOTO 45;
110: CLS; E_TAB_GUN_YAZ; CLS;
XY(10,3); WRITE(' BU ATAMALARI ONAYLIYORSANIZ : ');
WRITE(' DOSYAYA KAYIT İÇİN [E] '); XY(12,3);
WRITE(' EĞER ONAYLAMİYORSANIZ DİĞER DERSLERİN');

```



```

        WRITE(' ATAMA İÇİN [H] '); EH:=READKEY;
        UNTIL UPCASE(EH)='E';
        GOTO 113; END;
    GOTO 45;
END;
113: PROGRAMDOS_KAYIT;
    CLS;
    XY(23,5); REVERSE;
    WRITE('DİĞER SINIFIN DA ATAMASININ YAPILMASINI');
    WRITE('İSTİYOR MUSUNUZ [ E / H ] '); NORMAL;
    EH:=READKEY;
    IF UPCASE(EH)='E' THEN GOTO 31;
    S;
1003: END;
    PROGDKAPA;
    DERSDKAPA;
END;

PROCEDURE İSTEKLER;
BEGIN
    CLS;
    SIFIR_ATA;
    BOSLUK_ATA;
    DOS_BIL_OKU_ATA;
    REPEAT
        M:=0;
        CLS;
        XY(2,7); WRITE(BL[H],' BÖLÜMÜ ');
        CERCEVE(4,7,18,63); Z:=3;
        GUNLER;
        XY(6,10); WRITE('İSTEDİĞİNİZ SINIF :');
        XY(6,36); WRITE('GÜNÜ :');
        XY(6,50); WRITE('SAATİ :');
        REPEAT
            XY(6,31); READ(SINIF);
            UNTIL SINIF IN[1..4];
            REPEAT
                XY(6,45); READ(GUN);
                UNTIL GUN IN[1..5];
            REPEAT
                XY(6,58); READ(SAAT);
                UNTIL SAAT IN[1..9];
                PADI:=DPROG[SINIF,GUN,SAAT];
                IF PADI='*****' THEN PADI:=' ';
                IF (PADI=' ') THEN BEGIN
                    XY(20,7); WRITE(SINIF,' . SINIFLARIN ',GN[GUN]);
                    WRITE(' GÜNÜ BU SAATTE HERHANGİ BİR DERSİ YOKTUR .');
                    GOTO 100;END;
                IF PADI<>' ' THEN BEGIN
                    XY(20,7);
                    WRITE(SINIF,' .SINIFLARIN',GN[GUN],'GÜNÜ',PADI);
                    WRITE(' DERSİ VARDIR . ');
                END;
                XY(22,7);
                WRITE('*****');
                XY(23,7); WRITE('&& BİR DİĞER GÖZLEM İÇİN [E] ');
                WRITE('AKŞİ HALDE [H] &&'); XY(24,7);
            
```



```

WRITE(' &&&*****&&&');
REPEAT EH:=READKEY; UNTIL EH IN['E','e','H','h'];
UNTIL UPCASE(EH)<>'E';
IF SEC=2 THEN PROGDKAPA;
END;

PROCEDURE BASLANGIC;
BEGIN
REPEAT
CLS;
SIFIR_ATA;
BOSLUK_ATA;
DOS_BIL_OKU_ATA;
CLS; XY(12,5);
WRITE('ATAMALARIN İSTERSENİZ SCREEN veya PRINTER');
WRITE(' ÇIKTISINI ALABİLİRSİNİZ');XY(15,5);
WRITE('TERCİHİNİZ : SCREEN => [S] PRINTER => [P]');
EH:=READKEY;
REPEAT
UNTIL EH IN['S','P'];
IF UPCASE(EH)='S' THEN E_TAB_GUN_YAZ ELSE P_TAB_GN_YAZ;
CLS;
XY(10,5); REVERSE;
WRITE('HAFTALIK DERS PROGRAMI TABLOSUNU TEKRAR GÖRMEK ');
WRITE('İSTERMİSİNİZ '); NORMAL;
EH:=READKEY;
UNTIL UPCASE(EH)='H';
CLS; XY(10,5);
WRITE('İSTEDİĞİNİZ HERHANGİ BİR SINIFA AIT GÖRMEK');
WRITE('İSTEDİĞİNİZ DERS VAR MI ');
XY(12,5); REVERSE; WRITE(' GÖZLEM [E] '); NORMAL;
XY(12,20); REVERSE; WRITE(' ÇIKIŞ [H] '); NORMAL;
REPEAT EH:=READKEY; UNTIL EH IN['E','e','H','h'];
IF UPCASE(EH)='E' THEN İSTEKLER;
PROGDKAPA;
END;

PROCEDURE PROG_BAS;
BEGIN
CLS; XY(5,5);REVERSE;
WRITE(' BU BÖLÜMDE İSTEGİNİZE BAĞLI OLARAK '); NORMAL;
XY(8,10); WRITE(' (1) HAFTALIK DERS PROGRAMININ');
WRITE(' TAMAMINI GÖZLEYEBİLİRSİNİZ ');
XY(10,30); REVERSE; WRITE(' veya '); NORMAL;
XY(12,10); WRITE(' (2) SINIF/GUN/SAAT SEÇEBİLİRSİNİZ ');
XY(17,10); WRITE(' (1) veya (2) '); READ(SEC);
IF SEC=2 THEN İSTEKLER ELSE BASLANGIC;
END;

PROCEDURE IPTAL_YAP;
BEGIN
XY(24,10); WRITE(' &&& KAÇ SAAT USTUSTE &&& ');
READ(SEC); END;
FOR I:=SAAT TO (SEC+SAAT-1) DO
BEGIN
STR(100*SINIF+10*GUN+I,TU);
FINDKEY(PROGINX,PRECNUM,TU);

```



```

IF OK THEN BEGIN
  GETREC(PROGDOS,PRECNUM,PREC);
  CLS; XY(10,5);
  WRITE(PREC.PROGADI,' &&& DERSİNİN İPTALİ İÇİN ');
  WRITE(' HERHANGİ BİR TUŞA BASINIZ &&& '); EH:=READKEY;
  DELETEKEY(PROGINX,PRECNUM,TU);
  DELETETEREC(PROGDOS,PRECNUM); END;
  END;
  XY(10,5);
  WRITE(PREC.PROGADI,'DERSİ İÇİN GEREKLİ İPTAL YAPILDI');
  WRITE(' [ E N T E R ] &&& ');
  EH:=READKEY; CLS;
END;

PROCEDURE KAYIT_YAP;
BEGIN
  FOR I:=SAAT TO (S+SAAT-1) DO
    BEGIN
      STR(100*SINIF+10*GUN+I,TU);
      PREC.PROGNO:=TU;
      PREC.PROGADI:=PADI;
      PREC.PSAYI:=0;
      ADDREC(PROGDOS,PRECNUM,PREC);
      ADDKEY(PROGINX,PRECNUM,PREC.PROGNO);
    END;
END;

PROCEDURE GEN_OZ_DEGIS;
BEGIN
  CLS;
  XY(5,5);REVERSE; WRITE(' U Y A R I : '); NORMAL;
  XY(7,5);
  WRITE('HAFTALIK DERS PROGRAMININ TAMAMEN DEĞİŞMESİNİ ');
  WRITE('İSTİYORSANIZ');
  XY(9,15);WRITE('& DERS PROGRAMI HAREKET MENUSUNDEN & ');
  XY(11,15); WRITE('1-1 ÖNCEKİ SÖMESTİRE AİT BİLGİLERİN ');
  WRITE('SİLİNMESİNİ SEÇİNİZ ');
  XY(17,15); WRITE(' TAMAMI DEĞİŞSİN Mİ [E/H] ');
  EH:= READKEY;
  IF UPCASE(EH)='E' THEN EXIT;
  REPEAT
    CLS;
    SEC:=0; M:=0;
    İSTEKLER;
  IF M<>0 THEN BEGIN
    XY(22,7);BOSLUK;XY(24,7);BOSLUK;XY(23,7);BOSLUK;XY(23,7);
    WRITE('BU SAATE HERHANGİ BİR DERS ATANMASINI İSTİYOR ');
    WRITE(' MUSUNUZ ');
    EH:=READKEY;
    IF UPCASE(EH)='E' THEN
      BEGIN
        CLS;
        XY(10,5); WRITE('KAÇ SAAT ÜSTÜSTE :'); READLN(S);
        XY(12,5);WRITE('DERSİN ADINI GİRİNİZ :');READLN(PADI);
        KAYIT_YAP;
        CLS; XY(10,5);
        WRITE(' GEREKLİ KAYIT YAPILMIŞTIR [ E N T E R ] ');

```



```

EH:=READKEY;
END;
END;
CLS;
IF M=0 THEN BEGIN
  XY(23,7); REVERSE;
  WRITE(' &&& BU DERS SAATİNİN DEĞİŞMESİNİ İSTİYOR ');
  WRITE('MUSUNUZ [E/H] &&& '); NORMAL;
  EH:=READKEY;
IF UPCASE(EH)='E' THEN BEGIN
  IPTAL_YAP;
REPEAT
  P:=0;
  REPEAT
    CLS; XY(23,5);
    WRITE(PADI,' &&& DERSİNİ HANGİ GÜN ATANMASINI ');
    WRITE('İSTİYORSUNUZ &&& '); READ(GUN); CLS;
    BOS_SAAATLER;
    IF K=0 THEN BEGIN
      XY(7,62); REVERSE; WRITE(' BUGÜN BOŞ ');
      NORMAL;
      XY(9,62); REVERSE; WRITE(' SAAT YOK ');
      NORMAL;
      XY(11,62);REVERSE;WRITE(' E N T E R '); NORMAL;
      EH:=READKEY;
      XY(2,62); WRITE(' ');
      FOR I:=1 TO 8 DO BEGIN
        XY(5+I,62); WRITE(' ');
      END;
      END;XY(20,5);
      WRITE(' BAŞKA BİR GÜN İSTİYOR MUSUNUZ [E/H] ');
      EH:=READKEY;
    UNTIL UPCASE(EH)='H' ;
    XY(20,5); BOSLUK;
    XY(22,15);WRITE('KACINCI SAATTE BASLAYACAK:');READ(SAAT);
    XY(23,15);WRITE('KAÇ SAAT USTUSTE YAPACAKSINIZ:');READ(S);
    P:=P+S ;
    KAYIT_YAP;
    CLS;
    UNTIL P=SEC; XY(10,10);
    WRITE(' GEREKLİ DEĞİŞİKLİKLER YAPILDI [ E N T E R ] ');
    EH:=READKEY; CLS;
  END;
END;
CLS; XY(10,10);
WRITE('BİR BAŞKA DERSİ DE GÖRMEK veya DEĞİŞTİRMEK');
WRITE('İSTİYOR MUSUNUZ '); EH:=READKEY;
UNTIL UPCASE(EH)='H';
PROGDKAPA;
END;

PROCEDURE DERS_P_IPTAL;
BEGIN
CLS;
XY(5,5); REVERSE; WRITE(' U Y A R I : '); NORMAL;
XY(7,5);
WRITE('HAFTALIK DERS PROGRAMININ TAMAMININ İPTALINI ');

```



```

WRITE('İSTİYORSANIZ'); XY(9,15);
WRITE(' &&& DERS PROGRAMI HAREKET MENUSUNDEN &&& ');XY(11,15);
WRITE('11 ÖNCEKİ SÖMESTİRE AİT BİLGİLERİN SİLİNMESİ ni ');
XY(13,15);WRITE('SEÇİNİZ');
XY(17,15); WRITE(' TAMAMI İPTAL EDİLSİN Mİ [E/H] ');
EH_READ;
IF UPCASE(EH)='E' THEN EXIT;
REPEAT
SEC:=0; CLS;
İSTEKLER;
IF M=0 THEN BEGIN
XY(22,7); BOSLUK;
XY(23,7); BOSLUK;
XY(24,7); BOSLUK; XY(23,7);
WRITE('BU DERS SAATİNİN İPTALINI İSTİYOR MUSUNUZ [E/H]');
EH:=READKEY; CLS;
IF UPCASE(EH)='E' THEN İPTAL_YAP; END; CLS; XY(10,10);
WRITE(' BİR BAŞKA DERSİ DE İPTAL ETMEK İSTİYOR MUSUNUZ ');
EH:=READKEY; CLS;
UNTIL UPCASE(EH)='H';
PROGDKAPA;
END;

PROCEDURE ACIK;
BEGIN
CLS;XY(10,10);WRITE('BİR BAŞKA İŞLEME GEÇMEK VEYA');
WRITE('PROGRAMDAN ÇIKMAK İSTERSENİZ';
XY(13,30); WRITE('BÖLÜM ve SÖMESTİRİ ');
XY(16,29); WRITE(' girin [ E N T E R ]');
EH:=READKEY;
END;

PROCEDURE İLK;
BEGIN
CLS;
CERCEVE(1,1,24,80);
FOR I:=1 TO 78 DO BEGIN
XY(6,1+I); WRITE('='); END;XY(3,5);
WRITE('BU PROGRAM 1990-1991 YILINDA MASTER TEZİ OLARAK' );
WRITE('YAPILMIŞTIR '); XY(4,5); REVERSE;
WRITE('COPYRIGHT: ARAŞ. GÖR. ÜNSAL ÖZDOĞRU '); NORMAL;
XY(8,3); REVERSE; WRITE('AÇIKLAMA : '); NORMAL; XY(10,3);
WRITE('HAFTALIK DERS PROGRAMININ HAZIRLANMASINDA İKİ YOL');
WRITE('İZLENMİŞTİR :');XY(11,3);
WRITE('İLKİ, BİRBİRİNDEN BAĞIMSIZ OLAN 1. ve 4. SINIF ');
DERSLERİNİN ');
WRITE('ATANMASINI TAKİB BUNA BAĞLI OLARAK 2. ve 3.SINIF');
WRITE('DERSLERİNİN BİRBİRİYLE MİNİMUM ÖLÇÜDE ÇAKIŞMASI ');
WRITE('İÇİN DAHA ÖNCEDEN BELİRLENEN VERİLERİN GÖZ ÖNÜNDE');
WRITE('TUTULMASIYLA OLUŞTURULMUŞTUR. '); XY(15,3);
WRITE('İKİNCİSİ İSE,1.SINIF DERSLERİNDEN BAŞLAYARAK');
WRITE('DAHA SONRAKİ DERSLERİN BİR ÖNCEKİYLE ÇAKIŞMAYACAK');
WRITE('BİÇİMDE HAZIRLANMASIDIR.BURADA SINIFLARDAKİ');
WRITE('TOPLAM DERS SAYISI KADAR FARKLI PROGRAM ');
WRITE('OLUŞACAKTIR.SİZ ÇIKAN PROGRAMLAR İÇİNDE KENDİNİZE');
XY(18,3); WRITE('UYGUN GELENİ SEÇİN. '); XY(19,3);
WRITE('BUNA GÖRE : HAZIRLAYACAGINIZ DERS PROGRAMININ ');

```



```

WRITE('1.YÖNTEME GÖRE Mİ YOKSA İKİNCİ YÖNTEME GÖRE Mİ');
WRITE('HAZIRLANMASINI İSTİYORSUNUZ.');
```

XY(23,3); REVERSE; WRITE(' 3 ÇIKIŞ '); NORMAL;

END;

PROCEDURE ATA2;

LABEL 99,100,121,133;

BEGIN

FOR I:=1 TO K DO

BEGIN

Z:=0; SEC:=10; P:=0;

FOR GUN:=1 TO 5 DO

BEGIN

IF (P=2) THEN GOTO 121;

IF DER_SAY[GUN]>=3 THEN GOTO 121;

FOR SAAT:=1 TO 7 DO

BEGIN

IF (SEC=SAAT) OR (SAAT=SEC+1) THEN GOTO 99;

IF TRUNC((SS[I]-Z)/2)=(SS[I]-Z)/2 THEN BEGIN

IF (DPROG[SNF,GUN,SAAT]=' ') AND (SAAT<>5)

AND (DPROG[SNF,GUN,SAAT+1]=' ') AND (SAAT+1<>5)

THEN BEGIN DERS_BUL2;

IF HOCA=OGR THEN GOTO 99;

IF (((DPROG[SNF-Y,GUN,SAAT]=' ')

OR (DPROG[SNF-Y,GUN,SAAT+1]=' '))

AND ((DPROG[SNF,GUN,SAAT]=' ')

OR (DPROG[SNF,GUN,SAAT+1]=' ')))

THEN

BEGIN

DPROG[SNF,GUN,SAAT]:=DE[I];

DPROG[SNF,GUN,SAAT+1]:=DE[I];

Z:=Z+2; SEC:=SAAT; P:=1;

DER_SAY[GUN]:=DER_SAY[GUN]+1; GOTO 100;

END

ELSE GOTO 99;

END;

END;

IF (DPROG[SNF,GUN,SAAT]=' ') AND (SAAT<>5)

THEN BEGIN

DERS_BUL1;

IF HOCA=OGR THEN GOTO 99;

IF ((DPROG[SNF-Y,GUN,SAAT]=' ')

AND (DPROG[SNF,GUN,SAAT]=' '))

THEN

BEGIN

DPROG[SNF,GUN,SAAT]:=DE[I];

DER_SAY[GUN]:=DER_SAY[GUN]+1;

Z:=Z+1; SEC:=SAAT; P:=1; GOTO 100;

END;

END;

99: END;

100:

IF Z>=SS[I] THEN GOTO 133;

121: P:=P+1; END;

133: END;

END;


```

PROCEDURE TUS_BAS;
BEGIN
  XY(16,23); WRITE('*****');
  XY(17,23); WRITE('**** HERHANGİ BİR TUŞA BASINIZ ****');
  XY(18,23); WRITE('*****');
  EH:=READKEY;
END;

PROCEDURE DERS_ATAMA2;
BEGIN
  CLS; O:=10;
  BOLUM_SOR;
  SOMESTIR_SOR;
  DDOSAC;
  PDOSAC;
  BOLUM;
  IF FILELEN(DERSDOS)-1=0 THEN KAYIT_YOK
ELSE
  CLS; XY(5,7);
  WRITE('DİKKAT : TABLOLARIN DÜZGÜN ÇIKMASI İÇİN PRINTER');
  WRITE('İN AYARINI DÜZELTİN ');
  TUS_BAS;
  YIL_SOR;
  Z:=0; R:=0;
  SIFIR_ATA;
  DOS_BIL_OKU_ATA;
  SAYDIR;
  FOR SF:=1 TO 4 DO
  BEGIN
    CLS; XY(10,10);
    WRITE('** ŞU ANDA ',SF,' SINIFININ DERSLERİNE GÖRE ');
    WRITE('ATAMA YAPILMAKTADIR **');
    TUS_BAS;
    IF SF<>1 THEN BEGIN
      Z:=0; R:=0;
      SIFIR_ATA;
      BOSLUK_ATA;
      DOS_BIL_OKU_ATA;
      SAYDIR;
    END;
    IF BIL='GUZ' THEN SEC1:=(2*SF-1)
      ELSE SEC1:=(2*SF);
  FOR M:=1 TO K DO
  BEGIN
    SNF:=SF;
    ATA2; CLS;
    FOR NF:=1 TO 4 DO BEGIN
      IF BIL='GUZ' THEN SEC1:=(2*Nf-1)
    ELSE SEC1:=(2*Nf);
      IF NF<>SF THEN BEGIN
        SIFIR_ATA;
        KISIM; SNF:=NF;
        ATA2;
      END;
    END;
  END;
  DS:=100;
  P_TAB_GUN_YAZ;

```



```

END;
'3': BEGIN

```



```

CLS;
DERSMENUSU; XY(20,20);
REVERSE; WRITE(' S E C İ M İ N İ Z ==> ');
REPEAT
  XY(20,44); SECIM:=READKEY; NORMAL;
  UNTIL SECIM IN['1','2','3','4'];
  CASE SECIM OF
    '1': SABIT;
    '2': GUN_BOSALT;
    '3': OGR_GOR_ISTEK;
  END;
  CLS;
UNTIL SECIM='4';
END;
'4': BEGIN
  REPEAT
    ILK;XY(22,23);
    WRITE('***** (1) *** (2) *****');
    REPEAT
      SECIM:=READKEY;
      UNTIL SECIM IN['1','2','3'];
      CASE SECIM OF
        '1': DERS_ATAMA1;
        '2': DERS_ATAMA2;
      END;
    UNTIL SECIM='3';
  END;
'5': BEGIN
  REPEAT
    ACIK;
    BOSLUK_ATA; CLS;
    BOLUM_SOR;
    SOMESTIR_SOR;
    BOLUM;
    PDOSAC;
    IF FILELEN(PROGDOS)-1=0 THEN BEGIN
      KAYIT_YOK;
      EXIT; END;
  S:=0;
  CLS;
  HAFTALIK_MENU;
  XY(16,48);
  REPEAT
    SECIM:=READKEY;
    UNTIL SECIM IN['1','2','3','4'];
    CASE SECIM OF
      '1' : PROG_BAS;
      '2' : GEN_OZ_DEGIS;
      '3' : DERS_P_IPTAL;
    END;
  UNTIL SECIM='4';
  END;
'6': END;
IF EH<>'6' THEN GOTO BASLA;
END.

```


UNIT YARDIMCI'da TEZ 'de kullanılan bazı procedure'ler biraraya getirilmiştir.

UNIT YARDIMCI;

INTERFACE

USES CRT,DOS,PRINTER,TACCESS;

```
PROCEDURE DDOSAC;
PROCEDURE PDOSAC;
PROCEDURE DERSDKAPA;
PROCEDURE PROGDKAPA;
PROCEDURE CLS;
PROCEDURE XY(X,Y:INTEGER);
PROCEDURE REVERSE;
PROCEDURE NORMAL;
PROCEDURE KAYIT_YOK;
PROCEDURE DENEME1;
PROCEDURE CERCEVE(CX1,CY1,CX2,CY2:INTEGER);
PROCEDURE MENU;
PROCEDURE DERSMENUSU;
PROCEDURE HAFTALIK_MENU;
PROCEDURE BOLUM;
PROCEDURE BOSLUK;
PROCEDURE EH_READ;
PROCEDURE DEVAM;
PROCEDURE ONAY;
PROCEDURE CIZGI;
PROCEDURE BOLUM_SOR;
PROCEDURE SOMESTIR_SOR;
PROCEDURE GUNLER;
PROCEDURE DEBASLIK;
PROCEDURE DPBASLIK;
PROCEDURE DEKRAN;
PROCEDURE PDOS_SIL;
PROCEDURE DDOS_SIL;
PROCEDURE BOS;
PROCEDURE CIZ1;
PROCEDURE CIZ;
PROCEDURE SIFIR_ATA;
PROCEDURE DOS_BIL_OKU_ATA;
PROCEDURE SAYDIR;
```

TYPE

```
DERSREC=RECORD
DSAYI      :LONGINT;
DDERSNO    :STRING[2];
DDERSADI   :STRING[15];
OGRGOR     :STRING[25];
TEOUYG     :STRING[3];
BILGI      :STRING[3];
END;
```



```

PROGREC=RECORD
PSAYI      :LONGINT;
PROGNO     :STRING[3];
PROGADI    :STRING[15];
END;
CONST
M22 : ARRAY[1..4] OF STRING[30]=
('1-] DERS NO :',
'2-] DERS ADI :',
'3-] ÖĞRETİM GÖREVLİSİ :',
'4-] TEORİK-UYG SYS [T/U] :');

M21 : ARRAY[1..4] OF STRING[50]=
('1] YABANCI DİL ve TARİH DERSLERİNİN ATANMASI',
'2] BELİRLİ9]=
('PAZARTESİ','SALI','ÇARŞAMBA','PERŞEMBE','CUMA');

BL : ARRAY[1..4] OF STRING[10]=
('MATEMATİK','FİZİK','KİMYA','İSTATİSTİK');

MAXDATARECSIZE =48;
MAXKEYLEN      =3;
PAGE SIZE      =24;
PAGESTACKSIZE  =10;
ORDER          =12;
MAXHEIGHT      =4;

TEKRARSIZKAYIT =0;
TEKRARLIKAYIT  =1;

VAR
DREC : DERSREC;
PREC : PROGREC;
PROGDOS : DATAFILE;
PROGINX : INDEXFILE;
DERSDOS : DATAFILE;
DERSINX : INDEXFILE;
DRECNUM : LONGINT;
PRECNUM : LONGINT;
SEC,I,T,S,J,L,
K,Z,DS,SEC1,A,
R,B,Y,H,P,SNF,
N,M,SINIF,GUN,
O,SAAT,SF,NF : INTEGER;
DNO : STRING[2];
PADI, DADI : STRING[15];
OGR,HOCA : STRING[25];
TU,BIL : STRING[3];
EH,SECIM : CHAR;
DPROG : ARRAY[1..4,1..5,1..9] OF STRING[15];
TDIL : STRING[5];
YIL : STRING[9];
G : STRING[1];
SS,DER_SAY,NO,
SY : ARRAY[1..6] OF INTEGER;

```



```

DE,DY REVERS: ARRAY[1..9] OF STRING[15];
IMPLEMENTATION
TEXTCOLOR(0);
PROCEDURE DDOSAC;
BEGIN
    OPENFILE(DERSDOS,G+BIL+'DERS.DAT',SIZEOF(DREC));
    IF NOT OK THEN
        BEGIN
            MAKEFILE(DERSDOS,G+BIL+'DERS.DAT',SIZEOF(DREC));
            MAKEINDEX(DERSINX,G+BIL+'DERS.INX',
                ,SIZEOF(DREC.DDERSNO)-1,TEKRARSIZKAYIT);
        END
    ELSE
        OPENINDEX(DERSINX,G+BIL+'DERS.INX',
            ,SIZEOF(DREC.DDERSNO)-1,TEKRARSIZKAYIT);
    END;
END;

PROCEDURE PDOSAC;
BEGIN
    OPENFILE(PROGDOS,G+BIL+'PROG.DAT',SIZEOF(PREC));
    IF NOT OK THEN
        BEGIN
            MAKEFILE(PROGDOS,G+BIL+'PROG.DAT',SIZEOF(PREC));
            MAKEINDEX(PROGINX,G+BIL+'PROG.INX',
                ,SIZEOF(PREC.PROGNO)-1,TEKRARSIZKAYIT);
        END
    ELSE
        OPENINDEX(PROGINX,G+BIL+'PROG.INX',
            ,SIZEOF(PREC.PROGNO)-1,TEKRARSIZKAYIT);
    END;
END;

PROCEDURE DERSDKAPA;
BEGIN
    CLOSEFILE (DERSDOS);
    CLOSEINDEX(DERSINX);
    END;

PROCEDURE PROGDKAPA;
BEGIN
    CLOSEFILE (PROGDOS);
    CLOSEINDEX(PROGINX);
    END;

PROCEDURE CLS;
BEGIN
    CLRSCR;
    END;

PROCEDURE XY(X,Y:INTEGER);
BEGIN
    GOTOXY(Y,X);
    END;

```



```

PROCEDURE REVERSE;
BEGIN
    TEXTCOLOR(0);
    TEXTBACKGROUND(15);
    END;

PROCEDURE NORMAL;
BEGIN
    TEXTCOLOR(15);
    TEXTBACKGROUND(0);
    END;

PROCEDURE KAYIT_YOK;
BEGIN
    CLS;
    XY(12,20);REVERSE;
    WRITE(' &&& DOSYADA KAYIT YOK [ENTER] &&& ');NORMAL;
    EH:=READKEY;EXIT;
END;

PROCEDURE DENEME1;
BEGIN
    LOWVIDEO;
    FOR I:=1 TO 51 DO BEGIN
        XY(21,14+I);WRITE('0');XY(22,15+I);WRITE('2');
        END;
    LOWVIDEO;
    FOR I:=1 TO 49 DO
    FOR S:=1 TO 13 DO BEGIN
        XY(6+S,14+I);WRITE('[');END;
    HIGHVIDEO;
    FOR I:=1 TO 45 DO
    FOR S:=1 TO 13 DO BEGIN
        XY(6+S,18+I);WRITE('[');END;
    LOWVIDEO;
    FOR I:=1 TO 18 DO BEGIN
        XY(3+I,65);WRITE('0');XY(4+I,66);WRITE('2');
        END;
    END;

PROCEDURE CERCEVE(CX1,CY1,CX2,CY2:INTEGER);
BEGIN
    XY(CX1,CY1);WRITE(#201);XY(CX1,CY2);WRITE(#187);
    XY(CX2,CY1);WRITE(#200);XY(CX2,CY2);WRITE(#188);
    FOR I:=CY1+1 TO CY2-1 DO BEGIN
        XY(CX1,I);WRITE(#205);XY(CX2,I);WRITE(#205);
        END;
    FOR I:=CX1+1 TO CX2-1 DO BEGIN
        XY(I,CY1);WRITE(#186);
        XY(I,CY2);WRITE(#186);
        END;
END;

```



```
PROCEDURE MENU;  
BEGIN
```

[illegible]

```

PROCEDURE DERSMENUSU;
BEGIN
    CERCEVE(5,15,22,67);
    FOR I:=1 TO 4 DO BEGIN
        XY(2*I+8,17);WRITE(M21[I]);END;
        XY(7,23);
    WRITE('D E R S L E R İ N A T A N M A S I');
        XY(8,23);WRITE('=====');
    END;

```

```
PROCEDURE HAFTALIK_MENU;  
BEGIN  
    CLS;  
    CERCEVE(5,5,20,75);  
    FOR I:=1 TO 5 DO BEGIN  
        XY(6+I*2,15);WRITE(M2[I]);END;  
        XY(6,25);REVERSE;WRITE('HAFTALIK PROGRAM MENUSU');  
    NORMAL;  
END;
```

```
PROCEDURE BOLUM;  
BEGIN  
    H:=0;  
    IF G='M' THEN H:=1;  
    IF G='F' THEN H:=2;  
    IF G='K' THEN H:=3;  
    IF G='I' THEN H:=4;  
END;
```

```
PROCEDURE BOSLUK;  
BEGIN  
    WRITE(' ');  
END;
```

```
PROCEDURE EH_READ;  
BEGIN  
    REPEAT  
        EH:=READKEY;  
    UNTIL EH IN ['E', 'H'];  
END;
```



```
PROCEDURE DEVAM;
BEGIN
```

```
  XY(23,17);
  WRITE(' DEVAM ETMEK İSTİYOR MUSUNUZ [E/H] !!!!! ');
  EH_READ;
END;
```

```
PROCEDURE ONAY;
BEGIN
```

```
  XY(16,14);
  WRITE(' ## BİLGİLERİ ONAYLIYOR MUSUNUZ [E/H] ## ');
  EH_READ;
END;
```

```
PROCEDURE CIZGI;
BEGIN
```

```
  WRITE('..');
  END;
```

```
PROCEDURE BOLUM_SOR;
LABEL A,B;
BEGIN
CLS;
```

```
  CERCEVE(3,20,18,53);
  XY(5,27);REVERSE;WRITE(' BÖLÜMLER & KODLARI ');NORMAL;
  XY(6,27);WRITE('=====');
  XY(8,25);WRITE('MATEMATİK BÖLÜMÜ [ M ]');
  XY(10,25);WRITE('FİZİK BÖLÜMÜ [ F ]');
  XY(12,25);WRITE('KİMYA BÖLÜMÜ [ K ]');
  XY(14,25);WRITE('İSTATİSTİK BÖLÜMÜ [ I ]');
  XY(16,23);REVERSE; WRITE('HANGİ BÖLÜM [M/F/K/I] :');
```

```
NORMAL;
```

```
  REPEAT
```

```
A:   XY(16,51);READLN(G);
      IF (G='M') OR (G='F') OR (G='K') OR (G='I')
        THEN GOTO B
```

```
      ELSE GOTO A;
```

```
B: UNTIL LENGTH(G)=1;
  END;
```

```
PROCEDURE SOMESTIR_SOR;
BEGIN
```

```
  XY(20,22);REVERSE; WRITE('HANGİ SÖMERTİR [GUZ/BAH] : ');
```

```
NORMAL;
```

```
  REPEAT
```

```
    XY(20,51);
```

```
    READLN(BIL);
```

```
    IF LENGTH(BIL)>3 THEN BEGIN
```

```
      XY(20,51);WRITE(' '); END;
```

```
    UNTIL (BIL='GUZ') OR (BIL='BAH');
```

```
  END;
```



```
PROCEDURE GUNLER;
BEGIN
```

```
  XY(5+Z,25);WRITE('PAZARTESİ.....(1)');
  XY(7+Z,25);WRITE('SALI.....(2)');
  XY(9+Z,25);WRITE('ÇARŞAMBA.....(3)');
  XY(11+Z,25);WRITE('PERŞEMBE.....(4)');
  XY(13+Z,25);WRITE('CUMA.....(5)');
```

```
END;
```

```
PROCEDURE DEBASLIK;
BEGIN
```

```
  CLS;
  XY(1,20); WRITE('&&&&&& ',BL[H],' BÖLÜMÜ &&&&&&');
  XY(3,2);WRITE('DERS NO');XY(3,23);WRITE('DERS ADI');
  XY(4,2);WRITE('====');XY(4,15); WRITE('=====');
  XY(3,42);WRITE('TEO/UYG');XY(3,57);
  WRITE('ÖĞRETİM GÖREVLİSİ'); XY(4,42);
  WRITE('===='); XY(4,53); WRITE('=====');
```

```
END;
```

```
PROCEDURE DPBASLIK;
BEGIN
```

```
  CLS;
  WRITELN(LST,' &&&&&& ',BL[H],' BÖLÜMÜ &&&&&&');
  WRITELN(LST,' ');
  WRITE(LST,' DERS NO          DERS ADI          TEO/UYG');
  WRITELN(LST,'                ÖĞRETİM GÖREVLİSİ');
  WRITE(LST,'=====');
  WRITELN(LST,'=====');
```

```
END;
```

```
PROCEDURE DEKRAN;
BEGIN
```

```
  FOR I:=1 TO 4 DO BEGIN
    XY(2*I+4,12);WRITE(M22[I]);END;
    XY(3,17);WRITE('D E R S      D A G I L I M L A R I');
    XY(4,17);WRITELN('=====');
```

```
END;
```

```
PROCEDURE PDOS_SIL;
BEGIN
```

```
  BOLUM_SOR;  SOMESTIR_SOR;
  PDOSAC; ERASEFILE(PROGDOS); ERASEINDEX(PROGINX);
  XY(23,14);REVERSE;
  WRITE(' *** GEREKLİ DOSYA SİLİNMIŞTİR [E N T E R l*** ');
  IF LENGTH(BIL)>3 THEN
    BEGIN
      XY(20,51);WRITE(' ');
      END;
  UNTIL (BIL='GUZ') OR (BIL='BAH');
END;
```



```
PROCEDURE GUNLER;
BEGIN
```

```
  XY(5+Z,25);WRITE('PAZARTESİ.....(1)');
  XY(7+Z,25);WRITE('SALI.....(2)');
  XY(9+Z,25);WRITE('ÇARŞAMBA.....(3)');
  XY(11+Z,25);WRITE('PERŞEMBE.....(4)');
  XY(13+Z,25);WRITE('CUMA.....(5)');
  END;
```

```
PROCEDURE DEBASLIK;
BEGIN
```

```
  CLS;
  XY(1,20);WRITE('&&&&&& ',BL[H],' BÖLÜMÜ &&&&&&');
  XY(3,2);WRITE('DERS NO');XY(3,23);WRITE('DERS ADI');
  XY(4,2);WRITE('=====');
  XY(4,15);WRITE('=====');
  XY(3,42);WRITE('TEO/UYG');XY(3,57);
  WRITE('ÖĞRETİM GÖREVLİSİ'); XY(4,42);WRITE('=====');
  XY(4,53);WRITE('=====');
  END;
```

```
PROCEDURE DPBASLIK;
BEGIN
```

```
  CLS;
  WRITELN(LST,' &&&&&& ',BL[H],' BÖLÜMÜ &&&&&&');
  WRITELN(LST,' ');
  WRITE(LST,' DERS NO DERS ADI TEO/UYG');
  WRITELN(LST,' ÖĞRETİM GÖREVLİSİ');
  WRITE(LST,'=====');
  WRITELN(LST,'=====');
  END;
```

```
PROCEDURE DEKRAN;
BEGIN
```

```
  FOR I:=1 TO 4 DO BEGIN
    XY(2*I+4,12);WRITE(M22[I]);END;
  XY(3,17);WRITE('DERS D A G I L I M L A R I');
  XY(4,17);WRITELN('=====');
  END;
```

```
PROCEDURE PDOS_SIL;
BEGIN
```

```
  BOLUM_SOR;
  SOMESTIR_SOR;
  PDOSAC;
  ERASEFILE(PROGDOS);
  ERASEINDEX(PROGINX);
  XY(23,14);REVERSE;
  WRITE('*** GEREKLİ DOSYA SİLİNMİŞTİR [ENTER]***');
  NORMAL;EH:=READKEY;
  END;
```



```

PROCEDURE DDOS_SIL;
BEGIN
    BOLUM_SOR;
    SOMESTIR_SOR;
    DDOSAC;
    ERASEFILE(DERSDOS);
    ERASEINDEX(DERSINX);
    XY(23,14);REVERSE;
    WRITE('*** GEREKLI DOSYA SILINMIŞTİR [ E N T E R ] *** ');
    NORMAL;EH:=READKEY;
END;

```

```

PROCEDURE SAYDIR;

```

```

PROCEDURE BOS;

```

```

BEGIN

```

```

    FOR P:=1 TO 10 DO
        DE[P]:=' ';

```

```

    END;

```

```

PROCEDURE CIZ1;

```

```

BEGIN

```

```

    WRITE(LST,'3 3');

```

```

    FOR I:=1 TO 76 DO

```

```

        BEGIN

```

```

            WRITE(LST,'D');

```

```

        END;

```

```

    WRITELN(LST,'3');

```

```

END;

```

```

PROCEDURE CIZ;

```

```

BEGIN

```

```

    FOR I:=1 TO 78 DO

```

```

        BEGIN

```

```

            WRITE(LST,'D');

```

```

        END;

```

```

    END;

```

```

PROCEDURE SIFIR_ATA;

```

```

BEGIN

```

```

    FOR I:=1 TO 6 DO BEGIN

```

```

        DER_SAY[I]:=0;

```

```

        SS[I]:=0;

```

```

        DE[I]:=' ';

```

```

    END;

```

```

END;

```

```

PROCEDURE DOS_BIL_OKU_ATA;

```

```

BEGIN

```

```

    L:=FILELEN(PROGDOS)-1;

```

```

    FOR I:=1 TO L DO

```

```

        BEGIN

```

```

            GETREC(PROGDOS,I,PREC);

```



```

IF PREC.PSAYI=0 THEN BEGIN
    TU:=PREC.PROGNO;
    Z:=0;
    VAL(COPY(TU,1,1),SINIF,Z);
    VAL(COPY(TU,2,1),GUN,Z);
    VAL(COPY(TU,3,1),SAAT,Z);
    DPROG[SINIF,GUN,SAAT]:=PREC.PROGADI;
    END;
END;

PROCEDURE SAYDIR;
LABEL 1;
BEGIN
    FOR GUN:=1 TO 5 DO
        BEGIN
            FOR SAAT:=1 TO 9 DO
                BEGIN
                    IF ((DPROG[SNF,GUN,SAAT]='*****')
                        OR (DPROG[SNF,GUN,SAAT]='')
                        OR (DPROG[SNF,GUN,SAAT]='TARİH')
                        OR (DPROG[SNF,GUN,SAAT]='Y.DİL'))
                        THEN GOTO 1;
                    IF DPROG[SNF,GUN,SAAT] <> DPROG[SNF,GUN,SAAT+1]
                        THEN DER_SAY[GUN]:=DER_SAY[GUN]+1;
1:      END;
                END;
            END;
        END;

    BEGIN
        END.
    
```


ÖRNEK ÇIKIŞLAR :

Bu kısımda kullanılan iki farklı atama şekli örnekleriyle verilmektedir. Güz dönemine ait dersler, ikinci bölümde anlatılan ilk yöntemle , atama başlamadan önce bazı öğretim üyelerinin ders saatlerine göre istekleri gözönünde tutularak önceden atanırken, Haftalık Ders Programı , ilgili sınıfa göre atanan derslerin arasından bir karar verici tarafından anında belirlenmesiyle oluşturulur.

Diğer bir atama ise ilgili bölümün , o sömestirde vereceği ders sayısı kadar farklı , kısıtlara bağlı atanan tabloların arasından , yine bir karar verici tarafından uygun bir tanesini seçmesiyle belirlenir. Bahar dönemine ait dersler ikinci yöntemle göre atanmadan önce, bazılarının (özellikle 3 saat üstüste yapılması istenen) baştan istenen saate atanması sağlanmıştır.

Örnek çıktılarına geçmeden önce güz ve bahar dönemine ait dersler, öğretim üyeleri ve ders saati sayısı verilmiştir.

GUZ YARIYILINA AIT ATANAN DERSLER :

Y I L D I Z U N I V E R S I T E S I

\$\$\$ FEN-EDEBIYAT FAKULTESİ \$\$\$

MATEMATİK BÖLÜMÜ GUZ YY. DERS LİSTESİ

DERS NO	DERS ADI	TEO/UYG	ÖĞRETİM GÖREVLİSİ
11	MATEMATİK ANL. I	4/2	PROF. Ü. ÖRNEK
12	FİZİK	4/2	ÖĞR. GÖR. S. TOKCAN
13	LINEER CEBİR	3/0	PROF. M. ABACI
14	ELKT. HESAP I	3/2	Y.D.DR. A. KURUUZUM
31	MATEMATİK ANL.	4/2	PROF. A. TAŞDİZEN
32	DİF. DENKLEM I	3/1	PROF. Y. AKSOY
33	CEBİR I	3/1	PROF. DR. E. BALKANAY
34	ELKT. HESAP III	3/0	PROF. DR. B. ÇAĞAL
35	LINEER PROG. I	2/0	PROF. DR. I. SEZGINMAN
51	SOYUT C. SAY. T	3/1	PROF. DR. E. BALKANAY
52	NUMERİK ANALİZ	3/0	PROF. DR. B. ÇAĞAL
53	KOMP. FONK. TEO	2/1	DOÇ. DR. Y. KAHRAMANER
54	TOPOLOJİ	3/1	PROF. H. AVCI
79	ÇEŞİTLİ GEO.	3/0	PROF. DR. I. SEZGINMAN
71	SİSTEM ANALİZİ	3/0	PROF. DR. B. ÇAĞAL
72	YÖNEYLEM ARAS.	3/0	Y.D.DR. M. AHLATÇIOĞLU
73	FONK. ANALİZ	3/1	PROF. H. AVCI
77	İNTEGRAL D. GİR	3/0	PROF. Ü. ÖRNEK
78	MATRİS ANALİZİ	3/0	PROF. A. TAŞDİZEN

SEÇİME BAĞLI (I. TİP) ATAMALARDAN ÖRNEK :

YILDIZ UNIVERSİTESİ

FEN EDEBİYAT FAKÜLTESİ

1990-1991 GÜZ Y. HAFTALIK DERS PROGRAMI MATEMATİK BÖLÜMÜ

SAATLER	1. YY	3. YY	5. YY	7. YY
P 09-09.50	MATEMATİK ANL. I	ELKT. HESAP III		MATRİS ANALİZİ
A 10-10.50	MATEMATİK ANL. I	LINEER PROG. I		SİSTEM ANALİZİ
Z 11-11.50	FİZİK	LINEER PROG. I		FONK. ANALİZ
A 12-12.50	FİZİK		NUMERİK ANALİZ	FONK. ANALİZ
R 13-13.50		ÖĞLE	TATİL	
T 14-14.50	Y.DİL	MATEMATİK ANL.	*****	
E 15-15.50	Y.DİL	MATEMATİK ANL.	*****	
S 16-16.50		Y.DİL	*****	TARİH
I 17-17.50		Y.DİL	*****	TARİH

Y I L D I Z U N I V E R S I T E S I
F E N E D E B İ Y A T F A K Ü L T E S İ

1990-1991 G U Z Y Y . H A F T A L I K D E R S P R O G R A M I M A T E M A T İ K B Ö L U M U

SAATLER	1. YY	3. YY	5. YY	7. YY
S 09-09.50	LINEER CEBİR	DİF. DENKLEM I	TOPOLOJİ	
A 10-10.50	ELKT. HESAP I	DİF. DENKLEM I	TOPOLOJİ	
L 11-11.50		CEBİR I	KOMP. FONK. TEO	
I 12-12.50		CEBİR I		
13-13.50		O Ğ L E	T A T İ L İ	
14-14.50	Y.DİL		SOYUT C. SAY. T	YÖNEYLEM ARAS.
15-15.50	Y.DİL		SOYUT C. SAY. T	YÖNEYLEM ARAS.
16-16.50		TARİH		YÖNEYLEM ARAS.
17-17.50		TARİH		

YILDIZ UNIVERSİTESİ

FEN EDEBİYAT FAKÜLTESİ

1990-1991 GÜZ YY. HAFTALIK DERS PROGRAMI MATEMATİK BÖLÜMÜ

SAATLER	1. YY	3. YY	5. YY	7. YY
09-09.50	FİZİK	MATEMATİK ANL.	NUMERİK ANALİZ	*****
10-10.50	FİZİK	MATEMATİK ANL.	NUMERİK ANALİZ	*****
11-11.50	MATEMATİK ANL. I	ELKT. HESAP III		*****
12-12.50	MATEMATİK ANL. I	ELKT. HESAP III		*****
13-13.50		ÖĞLE	TATİL	
14-14.50		*****		*****
15-15.50		*****		*****
16-16.50	TARİH	*****		*****
17-17.50	TARİH	*****		*****

Y I L D I Z U N I V E R S I T E S I
F E N E D E B I Y A T F A K Ü L T E S İ

1990-1991 G U Z Y Y . H A F T A L I K D E R S P R O G R A M I M A T E M A T I K B Ö L Ü M Ü

SAATLER	1. YY	3. YY	5. YY	7. YY
P 09-09.50	ELKT. HESAP I	ÖEBİR I		
E 10-10.50	ELKT. HESAP I	ÖEBİR I		ÇESİTLİ GEO.
R 11-11.50	LINEER ÖEBİR	DİF. DENKLEM I		ÇESİTLİ GEO.
S 12-12.50	LINEER ÖEBİR	DİF. DENKLEM I		ÇESİTLİ GEO.
E 13-13.50		Ö Ç L E	T A T I L I	
M 14-14.50	Y.DİL			
B 15-15.50	Y.DİL			
E 16-16.50	Y.DİL		TARİH	
17-17.50	Y.DİL		TARİH	

Y I L D I Z U N I V E R S I T E S I
F E N E D E B I Y A T F A K U L T E S I

1990-1991 G U Z Y Y . H A F T A L I K D E R S P R O G R A M I M A T E M A T I K B Ö L Ü M Ü

SAATLER	1. YY	3. YY	5. YY	7. YY
C 09-09.50	MATEMATİK ANL. I		SOYUT C. SAY. T	SİSTEM ANALİZİ
U 10-10.50	MATEMATİK ANL. I		SOYUT C. SAY. T	SİSTEM ANALİZİ
M 11-11.50	FİZİK		TOPOLOJİ	MATRİS ANALİZİ
A 12-12.50	FİZİK		TOPOLOJİ	MATRİS ANALİZİ
13-13.50		Ö Ğ L E	T A T İ L İ	
14-14.50	ELKT. HESAP I	MATEMATİK ANL.		İNTEGRAL D. GIR
15-15.50	ELKT. HESAP I	MATEMATİK ANL.		İNTEGRAL D. GIR
16-16.50		Y.DİL		İNTEGRAL D. GIR
17-17.50		Y.DİL		

BAHAR YARIYILINA AİT ATANAN DERSLER :

SEÇİLEN ÖRNEK :

Y I L D I Z U N I V E R S I T E S İ

§§§§§ FEN-EDEBİYAT FAKÜLTESİ §§§§§

MATEMATİK BÖLÜMÜ BAH YY. DERS LİSTESİ

DERS NO	DERS ADI	TEO/UYG	ÖĞRETİM GÖREVLİSİ
21	MAT. ANALİZ II	4/2	PROF. Ü. ÖRNEK
22	FİZİK II	3/1	ÖĞR. GÖR. S. TOKCAN
23	ANALİTİK GEO.	4/0	PROF. M. ABACI
24	ELKT. HESAP II	3/0	Y.D.DR. A. KURUUZUM
25	SOYUT MATEMATİK	3/0	PROF. Y. AKSOY
41	MAT. ANALİZ IV	4/2	PROF. A. TAŞDİZEN
42	DİF. DENKLEM II	3/1	PROF. Y. AKSOY
43	CEBİR II	3/1	PROF. DR. E. BALKANAY
44	ELKT. HESAP IV	3/0	PROF. DR. B. ÇAĞAL
45	LİN. PROGRAM. I	2/0	PROF. DR. İ. SEZGİNMAN
61	REEL ANALİZ	4/1	PROF. H. AVCI
62	NUMERİK ANL. II	2/0	PROF. DR. B. ÇAĞAL
63	KOMP. FONK. TEO	3/0	D.DR. Y. KAHRAMANER
64	KİSMİ T. DİF. D	3/1	PROF. Y. ÖZDEMİR
65	DİF. GEOMETRİ	4/0	Y.D.DR. F. URAS
81	OPERASYONEL H.	3/0	PROF. Y. ÖZDEMİR
80	OYUNLAR TEORİ	3/0	Y.D.DR. M. AHLATÇIOĞLU
83	TANSÖREL ANALİZ	2/0	DOÇ. Y. GUVEN
88	VARYASYONLAR H.	3/0	DOÇ. Y. GUVEN

RASTGELE (II. TİP) ATAMAYLA OLUSTURULAN 17 TABLO ARASINDAN

SEÇİLEN ÖRNEK :

Y I L D I Z U N I V E R S I T E S İ

FEN EDEBİYAT FAKULTESİ

1990-1991 BAH YY. HAFTALIK DERS PROGRAMI MATEMATİK BÖLÜMÜ

SAATLER	1. YY	3. YY	5. YY	7. YY
P 09-09.50	SOYUT MATEMATİK		REEL ANALİZ	
A 10-10.50	MAT.ANALİZ II		NUMERİK ANL. II	
Z 11-11.50	MAT.ANALİZ II		NUMERİK ANL. II	
A 12-12.50	ELKT.HESAP II		KOMP. FONK. TEO	
R 13-13.50		Ö Ğ L E	T A T İ L İ	
T 14-14.50	Y.DİL			OPERASYONEL H.
E 15-15.50	Y.DİL		MAT. ANALİZ IV	OYUNLAR TEORİ
S 16-16.50	TURKÇE		MAT. ANALİZ IV	
I 17-17.50	TURKÇE			

Y I L D I Z U N I V E R S I T E S İ
F E N E D E B İ Y A T F A K Ü L T E S İ

1990-1991 B A H Y Y . H A F T A L I K D E R S P R O G R A M I M A T E M A T İ K B Ö L Ü M Ü

SAATLER	1. YY	3. YY	5. YY	7. YY
S 09-09.50	FİZİK II	*****	KİSMİ T. DİF. D	
A 10-10.50	FİZİK II	*****	KİSMİ T. DİF. D	
L 11-11.50	ANALİTİK GEO.	*****	DİF. GEOMETRİ	
I 12-12.50	ANALİTİK GEO.	*****	DİF. GEOMETRİ	
13-13.50		Ö Ğ L E	T A T İ L İ	
14-14.50	Y.DİL	*****		
15-15.50	Y.DİL	*****		
16-16.50		*****		
17-17.50		*****		

Y I L D I Z U N I V E R S I T E S I

FEN EDEBİYAT FAKÜLTESİ

1990-1991 BAH YY. HAFTALIK DERS PROGRAMI MATEMATİK BÖLÜMÜ

SAATLER	1. YY	3. YY	5. YY	7. YY
Ç 09-09.50	MAT. ANALİZ II	MAT. ANALİZ IV	KOMP. FONK. TEO	
A 10-10.50	MAT. ANALİZ II	MAT. ANALİZ IV	KOMP. FONK. TEO	
R 11-11.50	SOYUT MATEMATİK		REEL ANALİZ	
S 12-12.50	SOYUT MATEMATİK		REEL ANALİZ	
A 13-13.50		Ö Ğ L E	T A T İ L İ	
M 14-14.50	ELKT. HESAP II			OYUNLAR TEORİ
B 15-15.50	ELKT. HESAP II			OYUNLAR TEORİ
A 16-16.50	TARİH			
17-17.50	TARİH			

Y I L D I Z U N I V E R S I T E S I

FEN EDEBİYAT FAKÜLTESİ

1990-1991 BAH YY. HAFTALIK DERS PROGRAMI MATEMATİK BÖLÜMÜ

SAATLER	1. YY	3. YY	5. YY	7. YY
P 09-09.50		MAT. ANALİZ IV	DİF. GEOMETRİ	*****
E 10-10.50		MAT. ANALİZ IV	DİF. GEOMETRİ	*****
R 11-11.50	FİZİK II		KİSMİ T. DİF. D	*****
S 12-12.50	FİZİK II		KİSMİ T. DİF. D	*****
E 13-13.50		Ö Ğ L E	T A T İ L İ	
M 14-14.50	*****	*****	*****	*****
B 15-15.50	*****	*****	*****	*****
E 16-16.50	Y.DİL	*****	*****	*****
17-17.50	Y.DİL	*****	*****	*****

Y I L D I Z U N I V E R S I T E S İ

FEN EDEBİYAT FAKULTESİ

1990-1991 BAH YY. HAFTALIK DERS PROGRAMI MATEMATİK BÖLÜMÜ

SAATLER	1. YY	3. YY	5. YY	7. YY
C 09-09.50		DİF.DENKLEM II	*****	OPERASYONEL H.
U 10-10.50		DİF.DENKLEM II	*****	OPERASYONEL H.
M 11-11.50	MAT.ANALİZ II	MAT. ANALİZ IV	*****	TANSOREL ANALİZ
A 12-12.50	MAT.ANALİZ II	MAT. ANALİZ IV	*****	TANSOREL ANALİZ
13-13.50	Ö Ğ L E T A T I L I			
14-14.50	Y.DİL	CEBİR II	*****	VARYASYONLAR H.
15-15.50	Y.DİL	CEBİR II	*****	VARYASYONLAR H.
16-16.50			*****	VARYASYONLAR H.
17-17.50			*****	

4. B Ö L Ü M

S O N U Ç

Yapılan Bilgisayar Programı, bir ders programının hazırlanması sırasında ortaya çıkan problemlerin, önce bir kısıtlar formuna dönüştürülmesi daha sonra bu kısıtların kabul edilen bazı verilerle birlikte bilgisayar programı haline getirilmesinden ibarettir.

Herşeyden önce böyle bir konuyu incelemekle, Eğitim ve Öğretimde karşılaşılan problemlerden biri olan Haftalık Ders Programlarının hazırlanmasındaki güçlükleri, bilhassa öğrenci ve öğretim görevlileri açısından minumuma indirmeyi amaçladık. Yapılan çalışmada ortaya çıkan en önemli fayda, istenen verilerin bilgisayara verilmesiyle, bu verileri gerçekleştirecek atamanın tabloya dağılımındaki yerleştirme zamanını elle atamaya nazaran, minumuma indirilmesidir.

Uygulamada tabloların bazılarında herhangi bir çakışmaya rastlanmamakta fakat, sınıfların herhangi birinde veya ikisinde bazı derslerin hiç atanmadığı, bazı derslerinde ders saatinin eksik atandığı görülmüştür. Bunun nedeni, o ders için tabloda uygun bir yerleştirme işlemi yapamamasından kaynaklanmaktadır. Bu da kısıtların gerçekleşmemesidir.

Böyle bir konuya getirilen yaklaşım ise, atanmayan derslerin veya ders saatlerinin karar verici veya öğretim görevlisi tarafından en uygun yere atanmasıdır.

Türkçe, Tarih, Yabancı Dil dersleri ve Öğretim Görevlilerinin isteklerine bağlı olarak yapılan bazı ders atamaları tablodaki atamalara esneklik getirmekle birlikte istenen optimum sonuç için yarar sağlamıştır. Dolayısıyla herhangi bir isteğe bağlı olmadan yapılan bağımsız atamalar için bir alt yapı oluşturmaktadır. Bu alt yapının üzerine kurulan diğer atamaların gerçekleştirilmesi aşamasında, ön plana alınan kısıtların gerçekleşmesiyle oluşan tablolar, optimum çözüme yakın sonuçlar vermekle birlikte bazı derslerin koşulları sağlamadığı için atanmaması konu üzerinde tekrar çalışılmasını ve yeni incelemelerin yapılmasını gerektirmektedir.

K A Y N A K L A R

- [1] Geoffrion , A. , M. , Dyer , J. , S. , Feinberg, A. ,
" An Interactive Approrach for Multi-Criterion
Optimization, with an Application to the Operation of
an Academic Department " , Management Sciences ,
Vol:19 , No:4 , 1972
- [2] Fox, K. , A. , " Models of Resource Allocation and
Planning in Educational Institutions and Systems " ,
Optimization Techniques in Quantative Economic Models,
Economic Analysis and Operation Research , Editing
by Henri Theil , Vol:10 , North-Holland Publishing
Company Amsterdam-Canada, 1969
- [3] Anderson, J. , M. ,Bernhard , R. , H. , " A University
Examination-Scheduling Model to Minimize Multiple
Examination Days for Students " , Decision Sciences ,
Vol.12, 1981
- [4] Mc Clure, R. , H. , Wells , C. , E. , " Application
and Implementation , A Mathematical Programming Model
for Faculty Course Assignments " , Decision Sciences ,
Vol:15 , 1984
- [5] Owners Hand Book, Turbo Pascal (Ver 4.0) , Database
Toolbox, Academic Press, A.B.D. , 1987



Ö Z G E Ç M İ S

01.01.1965 yılında İstanbul'da doğdum. İlk öğrenimimi 1976 yılında Lütü Erçin ilkokulu, orta öğrenimimi 1982 yılında Beylerbeyi Lisesinde ,İstanbul'da tamamladım. 1985 yılında girmiş olduğum Mimar Sinan Üniversite'si Matematik Bölümü'nü 1989 yılında bitirdim. Halen Yıldız Üniversitesi Matematik Bölümü Uygulamalı Matematik Anabilim Dalı 'nda araştırma görevlisi olarak görev yapmaktayım.



