

**YILDIZ TEKNİK ÜNİVERSİTESİ**

**ELEKTRİK-ELEKTRONİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ**

**YAPISAL PROGRAMLAMA**

**ÖDEV**

**C’de Öğrenci Ders Kayıt Sistemi Otomasyonu**

**Hazırlayan**

**Bahattin KOÇ**

**18011704**

**KOD**

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <conio.h>

#include <string.h>

#define TYPE\_OF\_OGRENCI 0

#define TYPE\_OF\_OGRETMEN 1

#define TYPE\_OF\_DERS 2

#define TYPE\_OF\_DERSKAYIT 3

#define TYPE\_OF\_DERSOGRETMENKONTROL 4

#define TYPE\_OF\_AYARLAR 5

#define FILE\_EXIST 1

#define FILE\_NONE 0

#define FILE\_EMPTY -1

#define PATH\_SIZE 260

/\*This value(260) is Maximum Path Length Limitation of Windows 10\*/

//STRUCTLAR

typedef struct Ogrenci{

int id;

char adi[20];

char soyadi[20];

int toplam\_ders;

int toplam\_kredi;

struct Ogrenci \*next;

}OGRENCI;

typedef struct Ogretmen{

int id;

char adi[20];

char soyadi[20];

char unvan[20];

struct Ogretmen \*next;

}OGRETMEN;

typedef struct Ders{

char id[10];

char adi[30];

int kredi;

int kontenjan;

int ogretmen\_id;

struct Ders \*next;

}DERS;

typedef struct OgrenciDersKayit{

int id;//AUTO INCREMENT

int ogrenci\_id;

char ders\_id[10];

int kayit\_durumu;

char kayit\_tarihi[10];

struct OgrenciDersKayit \*next;

}OGRENCIDERSKAYIT;

typedef struct Ayarlar{

int dersSinir;

int krediSinir;

int dersKayitAutoINC;

int dersKontenjan;

}AYARLAR;

/////////////

///PROTOTIPLER

void OPENFILE(FILE \*\*, char \*, char \*);

void FILETOSTRUCT(void \*\*, int, char \*);

void SHOWSTRUCT(void \*, int);

void LOADSTRUCT(OGRENCI \*\*, OGRETMEN \*\*, DERS \*\*, OGRENCIDERSKAYIT \*\*, char \*, char \*, char \*, char \*);

int FILEISEXIST(char \*);

int IS\_ID\_EXIST(void \*\*, int, char \*, int);

void STRUCTTOFILE(void \*\*, int, char \*);

void DELETEBYID(void \*\*, int, char \*, int, char \*);//bu ikisi için fonksiyon pointerı kullanarak yap

void UPDATEBYID(void \*\*, void\*\*, int, char \*, int, char \*);

void VERILENDERSLER(DERS \*, int);

void DERSIALANLAR(OGRENCIDERSKAYIT \*, OGRENCI \*, char \*);

void OGRENCIDERSLISTELE(OGRENCIDERSKAYIT \*, DERS \*, int);

void SINIFLISTESITXT(DERS \*, OGRENCIDERSKAYIT \*, OGRENCI \*, int);

/////////////

int main(){

setlocale(LC\_ALL, "Turkish");

FILE \*dosya;

OGRETMEN \*ogretmenHead = NULL, \*ogretmenTemp;

OGRENCI \*ogrenciHead = NULL, \*ogrenciTemp;

DERS \*dersHead = NULL, \*dersTemp;

OGRENCIDERSKAYIT \*dersKayitHead, \*dersKayitTemp, \*dersKayitTemp2;

AYARLAR \*ayarlar = (AYARLAR \*) malloc(sizeof(AYARLAR));

int secim = 0, tempNumber, kayitEr, ogrenciEr, dersEr;

void \*tempVoid, \*tempVoid2;

char \*OGRENCIYOL, \*OGRETMENYOL, \*DERSYOL, \*DERSKAYITYOL, \*AYARLARYOL, tempChar[10];

OGRENCIYOL = (char \*) malloc(PATH\_SIZE \* sizeof(char));

OGRETMENYOL = (char \*) malloc(PATH\_SIZE \* sizeof(char));

DERSYOL = (char \*) malloc(PATH\_SIZE \* sizeof(char));

DERSKAYITYOL = (char \*) malloc(PATH\_SIZE \* sizeof(char));

AYARLARYOL = (char \*) malloc(PATH\_SIZE \* sizeof(char));

printf("\*\*\*ÖN AYAR SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

printf("Öğrenci Dosyası Yolunu Giriniz (OGRENCILER.txt): "); scanf("%s", OGRENCIYOL);

printf("Öğretmen Dosyası Yolunu Giriniz (OGRETMENLER.txt): "); scanf("%s", OGRETMENYOL);

printf("Ders Dosyası Yolunu Giriniz (DERSLER.txt): "); scanf("%s", DERSYOL);

printf("Ders-Kayıt Dosyası Yolunu Giriniz (OGRENCIDERSKAYIT.txt): "); scanf("%s", DERSKAYITYOL);

printf("Ayarlar Dosyası Yolunu Giriniz (AYARLAR.txt): "); scanf("%s", AYARLARYOL);

LOADSTRUCT(&ogrenciHead, &ogretmenHead, &dersHead, &dersKayitHead, OGRENCIYOL, OGRETMENYOL, DERSYOL, DERSKAYITYOL);

if(ISFILEEXIST(AYARLARYOL) != FILE\_EXIST){//AYAR DOSYASI KONTROLÜ

if(ISFILEEXIST(AYARLARYOL) == FILE\_NONE)//Dosya var değilse ve dosya yoktuysa oluştur

if((dosya = fopen(AYARLARYOL, "w")) == NULL){

printf("Ayarlar dosyası olşturulurken hata meydana geldi! Çıkış yapılıyor..");

exit(1);

}else fclose(dosya);

OPENFILE(&dosya, AYARLARYOL, "w");

printf("Görünüşe bakılırsa Ayarlar dosyan oluşturulmamış veya içerisi boş.\nHadi beraber oluşturalım.\n1-Ders Sınırı: ");

scanf("%d", &ayarlar->dersSinir);

printf("2-Kredi Sınır: ");

scanf("%d", &ayarlar->krediSinir);

printf("3-Ders Kontenjan Sınır: ");

scanf("%d", &ayarlar->dersKontenjan);

ayarlar->dersKayitAutoINC = 0;

fprintf(dosya, "%d %d %d %d", ayarlar->dersSinir, ayarlar->krediSinir, ayarlar->dersKayitAutoINC, ayarlar->dersKontenjan);

fclose(dosya);

}else{//Dosya varsa

OPENFILE(&dosya, AYARLARYOL, "r");

fscanf(dosya, "%d %d %d %d", &ayarlar->dersSinir, &ayarlar->krediSinir, &ayarlar->dersKayitAutoINC, &ayarlar->dersKontenjan);

fclose(dosya);

}

while(secim != -1){

system("cls");

printf("\*\*\*WELCOME TO MY HOMEWORK\*\*\*\n\n");

printf("---Öğretim Üyesi İşlemleri---\n1-Ekle\n2-Sil\n3-Güncelle\n---\n");

printf("---Ders İşlemleri---\n4-Ekle\n5-Sil\n6-Güncelle\n---\n");

printf("---Öğrenci İşlemleri---\n7-Ekle\n8-Sil\n9-Güncelle\n---\n");

printf("---Öğrenci-Ders Kayıt İşlemleri---\n10-Ekle\n11-Güncelle\n---\n");

printf("---Diğer İşlemler---\n12-IDsi verilen öğretim üyesinin verdiği dersler\n13-Ders kodu verilen bir derse kayıtlı olan tüm öğrencileri listesi\n14-Öğrenci numarası verilen bir öğrencinin aldığı tüm dersler\n15-ID’si verilen öğretim üyesinin verdiği bir derse ait kod kullanarak dersi alan tüm öğrencilerin sınıf listesini DERSKODU\_SINIFLİSTESİ.txt dosyasına yazan fonksiyon\n---\n");

printf("Seciminiz (çıkmak içim -1): ");

scanf("%d", &secim);

if(secim == 1){//Öğretmen Ekle

system("cls");

printf("\*\*\*ÖĞRETMEN EKLEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

ogretmenTemp = (OGRETMEN \*) malloc(sizeof(OGRETMEN));

tempVoid = &ogretmenHead;//uyarı almamak için böyle bir aktarma kullandım.

do{

printf("Öğretmen ID: ");

scanf("%d", &ogretmenTemp->id);

if(IS\_ID\_EXIST(tempVoid, ogretmenTemp->id, NULL, TYPE\_OF\_OGRETMEN) || ogretmenTemp->id < 0)

printf("Var olan bir ID veya negatif bir sayı girdiniz. Tekrar deneyiniz!\n");

}while(IS\_ID\_EXIST(tempVoid, ogretmenTemp->id, NULL, TYPE\_OF\_OGRETMEN) || ogretmenTemp->id < 0);

printf("Öğretmen Adı: ");

scanf("%s", &ogretmenTemp->adi);

printf("Öğretmen Soyadı: ");

scanf("%s", &ogretmenTemp->soyadi);

printf("Öğretmen Unvan: ");

scanf("%s", &ogretmenTemp->unvan);

ogretmenTemp->next = ogretmenHead;

ogretmenHead = ogretmenTemp;

STRUCTTOFILE(tempVoid, TYPE\_OF\_OGRETMEN, OGRETMENYOL);

printf("Ders kaydı başarılı!\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 2){//Öğretmen Sil

system("cls");

printf("\*\*\*ÖĞRETMEN SİLME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(ogretmenHead, TYPE\_OF\_OGRETMEN);

tempVoid = &ogretmenHead;

do{

printf("\nÖğretmen ID (çıkış için -1): ");

scanf("%d", &tempNumber);

if(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1)

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1);

tempVoid = &dersHead;

if(tempNumber != -1)

if(IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_DERSOGRETMENKONTROL))

printf("Ders veren bir öğretmen silmek istiyorsunuz. Bunun için önce dersi silmelisiniz!");

else{

tempVoid = &ogretmenHead;

DELETEBYID(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN, OGRETMENYOL);

printf("Silme işlemi başarılı!");

}

printf("\nDevam etmek için bir tuşa basınız...");

getch();

}

else if(secim == 3){//Öğretmen Güncelle

system("cls");

printf("\*\*\*ÖĞRETMEN GÜNCELLEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(ogretmenHead, TYPE\_OF\_OGRETMEN);

tempVoid = &ogretmenHead;

do{

printf("\nÖğretmen ID (çıkış için -1): ");

scanf("%d", &tempNumber);

if(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1)

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1);

if(tempNumber != -1){

UPDATEBYID(tempVoid, NULL, tempNumber, NULL, TYPE\_OF\_OGRETMEN, OGRETMENYOL);

printf("Güncelleme işlemi başarılı!");

}

printf("\nDevam etmek için bir tuşa basınız...");

getch();

}

else if(secim == 4){//Ders Ekleme

system("cls");

printf("\*\*\*DERS EKLEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

dersTemp = (DERS \*) malloc(sizeof(DERS));

tempVoid = &dersHead;//uyarı almamak için böyle bir aktarma kullandım.

do{

printf("Ders ID: ");

scanf("%s", &dersTemp->id);

if(IS\_ID\_EXIST(tempVoid, 0, dersTemp->id, TYPE\_OF\_DERS))

printf("Var olan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(IS\_ID\_EXIST(tempVoid, 0, dersTemp->id, TYPE\_OF\_DERS));

printf("Ders Adı: ");

scanf("%s", &dersTemp->adi);

printf("Ders Kredisi: ");

scanf("%d", &dersTemp->kredi);

dersTemp->kontenjan = 0; //Varsayılan değer

tempVoid = &ogretmenHead;

do{

printf("Öğretmen ID (yoksa -1): ");

scanf("%d", &dersTemp->ogretmen\_id);

}while(!IS\_ID\_EXIST(tempVoid, dersTemp->ogretmen\_id, NULL, TYPE\_OF\_OGRETMEN) && dersTemp->ogretmen\_id != -1);

dersTemp->next = dersHead;

dersHead = dersTemp;

tempVoid = &dersHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERS, DERSYOL);

printf("Ders kaydı başarılı!\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 5){//Ders Silme ---- ÖĞRENCI DERS KAYITLARI YAPILDIKTAN SONRA YAPILACAK

system("cls");

printf("\*\*\*DERS SİLME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(dersHead, TYPE\_OF\_DERS);

tempVoid = &dersHead;

do{

printf("\nDers ID (çıkış için -1): ");

scanf("%s", tempChar);

if(!IS\_ID\_EXIST(tempVoid, 0, tempChar, TYPE\_OF\_DERS) && strcmp(tempChar, "-1"))

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, 0, tempChar, TYPE\_OF\_DERS) && strcmp(tempChar, "-1"));

if(strcmp(tempChar, "-1")){//dersKayitta ders ID'si aratılıcak ve bulunan her ders silinip ders id sindeki öğrenci için ders ve kredi düzenlemesi yapılıcak

dersKayitTemp = dersKayitHead;

dersTemp = dersHead;//Silinecek dersin bilgilerini çek

while(strcmp(dersTemp->id, tempChar) && dersTemp != NULL)

dersTemp = dersTemp->next;

while(dersKayitTemp != NULL){

if(!strcmp(dersKayitTemp->ders\_id, tempChar)){//Ders IDsi bulunduysa

//Öğrenci işlemleri

ogrenciTemp = ogrenciHead;

while(ogrenciTemp->id != dersKayitTemp->ogrenci\_id)

ogrenciTemp = ogrenciTemp->next;

ogrenciTemp->toplam\_ders -= 1;

ogrenciTemp->toplam\_kredi -= dersTemp->kredi;

tempVoid = &ogrenciHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_OGRENCI, OGRENCIYOL);

}

dersKayitTemp = dersKayitTemp->next;

}

dersKayitTemp = dersKayitHead;

while(dersKayitTemp != NULL){

if(!strcmp(dersKayitTemp->ders\_id, tempChar)){

tempVoid = &dersKayitHead;

DELETEBYID(tempVoid, dersKayitTemp->id, NULL, TYPE\_OF\_DERSKAYIT, DERSKAYITYOL);

dersKayitTemp = dersKayitHead;

}

else

dersKayitTemp = dersKayitTemp->next;

}

tempVoid = &dersHead;

DELETEBYID(tempVoid, 0, tempChar, TYPE\_OF\_DERS, DERSYOL);

printf("Silme işlemi başarılı!");

}else

printf("Silme işlemi gerçekletirilmedi!");

printf("\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 6){//Ders Güncelleme

system("cls");

printf("\*\*\*DERS GÜNCELLEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(dersHead, TYPE\_OF\_DERS);

tempVoid = &dersHead;

tempVoid2 = &ogretmenHead;

do{

printf("\nDers ID (çıkış için -1): ");

scanf("%s", tempChar);

if(!IS\_ID\_EXIST(tempVoid, 0, tempChar, TYPE\_OF\_DERS) && strcmp(tempChar, "-1"))

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, 0, tempChar, TYPE\_OF\_DERS) && strcmp(tempChar, "-1"));

if(strcmp(tempChar, "-1")){

UPDATEBYID(tempVoid, tempVoid2, 0, tempChar, TYPE\_OF\_DERS, DERSYOL);

printf("Güncelleme işlemi başarılı!");

}

printf("\nDevam etmek için bir tuşa basınız...");

getch();

}

else if(secim == 7){//Öğrenci Ekleme

system("cls");

printf("\*\*\*ÖĞRENCİ EKLEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

ogrenciTemp = (OGRENCI \*) malloc(sizeof(OGRENCI));

tempVoid = &ogrenciHead;//uyarı almamak için böyle bir aktarma kullandım.

do{

printf("Öğrenci ID: ");

scanf("%d", &ogrenciTemp->id);

if(IS\_ID\_EXIST(tempVoid, ogrenciTemp->id, NULL, TYPE\_OF\_OGRENCI) || ogrenciTemp->id < 0)

printf("Var olan bir ID veya negatif bir sayı girdiniz. Tekrar deneyiniz!\n");

}while(IS\_ID\_EXIST(tempVoid, ogrenciTemp->id, NULL, TYPE\_OF\_OGRENCI) || ogrenciTemp->id < 0);

printf("Öğrenci Adı: ");

scanf("%s", &ogrenciTemp->adi);

printf("Öğrenci Soyadı: ");

scanf("%s", &ogrenciTemp->soyadi);

ogrenciTemp->toplam\_ders = ogrenciTemp->toplam\_kredi = 0;//Varsayılan olarak 0

ogrenciTemp->next = ogrenciHead;

ogrenciHead = ogrenciTemp;

STRUCTTOFILE(tempVoid, TYPE\_OF\_OGRENCI, OGRENCIYOL);

printf("Öğrenci kaydı başarılı!\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 8){//Öğrenci Sil

system("cls");

printf("\*\*\*ÖĞRENCİ SİLME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(ogrenciHead, TYPE\_OF\_OGRENCI);

tempVoid = &ogrenciHead;

do{

printf("\nÖğrenci ID (çıkış için -1): ");

scanf("%d", &tempNumber);

if(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRENCI) && tempNumber != -1)

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRENCI) && tempNumber != -1);

if(tempNumber != -1){

dersKayitTemp = dersKayitHead;

while(dersKayitTemp != NULL){

if(dersKayitTemp->ogrenci\_id == tempNumber && dersKayitTemp->kayit\_durumu == 1){//Öğrenci derse kayıtlı ise dersin kontenjanını bir azalt

dersTemp = dersHead;

while(strcmp(dersTemp->id, dersKayitTemp->ders\_id))

dersTemp = dersTemp->next;

dersTemp->kontenjan -= 1;

tempVoid = &dersHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERS, DERSYOL);

}

dersKayitTemp = dersKayitTemp->next;

}

dersKayitTemp = dersKayitHead;

while(dersKayitTemp != NULL){

if(dersKayitTemp->ogrenci\_id == tempNumber){

tempVoid = &dersKayitHead;

DELETEBYID(tempVoid, dersKayitTemp->id, NULL, TYPE\_OF\_DERSKAYIT, DERSKAYITYOL);

dersKayitTemp = dersKayitHead;

}

else

dersKayitTemp = dersKayitTemp->next;

}

tempVoid = &ogrenciHead;

DELETEBYID(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRENCI, OGRENCIYOL);

printf("Silme işlemi başarılı!");

}

printf("\nDevam etmek için bir tuşa basınız...");

getch();

}

else if(secim == 9){//Öğrenci Güncelle

system("cls");

printf("\*\*\*ÖĞRENCİ GÜNCELLEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(ogrenciHead, TYPE\_OF\_OGRENCI);

tempVoid = &ogrenciHead;

do{

printf("\nÖğrenci ID (çıkış için -1): ");

scanf("%d", &tempNumber);

if(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRENCI) && tempNumber != -1)

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRENCI) && tempNumber != -1);

if(tempNumber != -1){

UPDATEBYID(tempVoid, NULL, tempNumber, NULL, TYPE\_OF\_OGRENCI, OGRENCIYOL);

printf("Güncelleme işlemi başarılı!");

}

printf("\nDevam etmek için bir tuşa basınız...");

getch();

}

else if(secim == 10){//Öğrenci-Ders Kayıt Ekleme

system("cls");

printf("\*\*\*ÖĞRENCİ-DERS KAYIT SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

dersKayitTemp = (OGRENCIDERSKAYIT \*) malloc(sizeof(OGRENCIDERSKAYIT));

do{

tempVoid = &ogrenciHead;//uyarı almamak için böyle bir aktarma kullandım.

do{

printf("\nÖğrenci ID: ");

scanf("%d", &dersKayitTemp->ogrenci\_id);

if(!IS\_ID\_EXIST(tempVoid, dersKayitTemp->ogrenci\_id, NULL, TYPE\_OF\_OGRENCI) || dersKayitTemp->ogrenci\_id < 0)

printf("Olmayan bir ID veya negatif bir sayı girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, dersKayitTemp->ogrenci\_id, NULL, TYPE\_OF\_OGRENCI) || dersKayitTemp->ogrenci\_id < 0);

ogrenciTemp = ogrenciHead;

while(ogrenciTemp->id != dersKayitTemp->ogrenci\_id)

ogrenciTemp = ogrenciTemp->next;

tempVoid = &dersHead;

do{

printf("Ders ID: ");

scanf("%s", &dersKayitTemp->ders\_id);

if(!IS\_ID\_EXIST(tempVoid, 0, dersKayitTemp->ders\_id, TYPE\_OF\_DERS))

printf("Olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, 0, dersKayitTemp->ders\_id, TYPE\_OF\_DERS));

dersTemp = dersHead;

while(strcmp(dersTemp->id, dersKayitTemp->ders\_id))

dersTemp = dersTemp->next;

printf("Kayıt Tarihi (gg-aa-yyyy): ");

scanf("%s", &dersKayitTemp->kayit\_tarihi);

//Aynı öğrenci-ders kaydının olup olmadığını kontrol ediyor

dersKayitTemp2 = dersKayitHead;

tempNumber = 1;

while(dersKayitTemp2 != NULL && tempNumber){

if(dersKayitTemp2->ogrenci\_id == dersKayitTemp->ogrenci\_id)

if(!strcmp(dersKayitTemp2->ders\_id, dersKayitTemp->ders\_id))

//if(!strcmp(dersKayitTemp2->kayit\_tarihi, dersKayitTemp->kayit\_tarihi))

tempNumber = 0;

if(tempNumber)

dersKayitTemp2 = dersKayitTemp2->next;

}

kayitEr = tempNumber;

//Dersin kontenjanının aşılıp aşılmadığını kontrol ediyor

while(strcmp(dersTemp->id, dersKayitTemp->ders\_id))

dersTemp = dersTemp->next;

if(dersTemp->kontenjan + 1 <= ayarlar->dersKontenjan) dersEr = 1;

else dersEr = 0;

//Öğrencinin ders sınırını veya kredi sınırını aşıp aşmadığı kontrol ediliyor

while(ogrenciTemp->id != dersKayitTemp->ogrenci\_id)

ogrenciTemp = ogrenciTemp->next;

if(ogrenciTemp->toplam\_ders + 1 < ayarlar->dersSinir && ogrenciTemp->toplam\_kredi + dersTemp->kredi < ayarlar->krediSinir) ogrenciEr = 1;

else ogrenciEr = 0;

if(!(kayitEr \* dersEr \* ogrenciEr)){

printf("Bu hatayı alma sebepleriniz;\nVar olan bir kayıt girmiş olabilirsiniz.\n\* Dersin kontenjanını aşmış olabilirsiniz.\n\* Öğrencinin ders sınırı veya kredi sınırını aşmış olabilirsiniz.\nDeğişiklik yapmak için Güncelleme yapabilirsiniz\nÇıkış için -1, Tekrar denemek için 0!: ");

scanf("%d", &tempNumber);

}

}while(!(kayitEr \* dersEr \* ogrenciEr) && tempNumber != -1);

if(tempNumber != -1){

//dersKayit ID si arttırıldı

ayarlar->dersKayitAutoINC++;

dersKayitTemp->id = ayarlar->dersKayitAutoINC;

dersKayitTemp->kayit\_durumu = 1;//Varsayılan olarak kayıtlı

dersKayitTemp->next = dersKayitHead;

dersKayitHead = dersKayitTemp;

dersTemp->kontenjan++;

ogrenciTemp->toplam\_ders++;

ogrenciTemp->toplam\_kredi += dersTemp->kredi;

tempVoid = &dersHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERS, DERSYOL);

tempVoid = &ogrenciHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_OGRENCI, OGRENCIYOL);

tempVoid = &dersKayitHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERSKAYIT, DERSKAYITYOL);

tempVoid = &ayarlar;

STRUCTTOFILE(tempVoid, TYPE\_OF\_AYARLAR, AYARLARYOL);

printf("Öğrenci-Ders kaydı başarılı!");

}else{

free(dersKayitTemp);

}

printf("\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 11){//Öğrenci-Ders Kayıt Güncelleme

system("cls");

printf("\*\*\*ÖĞRENCİ-DERS GÜNCELLEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

dersKayitTemp = (OGRENCIDERSKAYIT \*) malloc(sizeof(OGRENCIDERSKAYIT));

SHOWSTRUCT(dersKayitHead, TYPE\_OF\_DERSKAYIT);

tempVoid = &ogrenciHead;//uyarı almamak için böyle bir aktarma kullandım.

do{

printf("\nÖğrenci ID: ");

scanf("%d", &dersKayitTemp->ogrenci\_id);

if(!IS\_ID\_EXIST(tempVoid, dersKayitTemp->ogrenci\_id, NULL, TYPE\_OF\_OGRENCI) || dersKayitTemp->ogrenci\_id < 0)

printf("Olmayan bir ID veya negatif bir sayı girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, dersKayitTemp->ogrenci\_id, NULL, TYPE\_OF\_OGRENCI) || dersKayitTemp->ogrenci\_id < 0);

ogrenciTemp = ogrenciHead;

while(ogrenciTemp->id != dersKayitTemp->ogrenci\_id)

ogrenciTemp = ogrenciTemp->next;

tempVoid = &dersHead;

do{

printf("Ders ID: ");

scanf("%s", &dersKayitTemp->ders\_id);

if(!IS\_ID\_EXIST(tempVoid, 0, dersKayitTemp->ders\_id, TYPE\_OF\_DERS))

printf("Olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, 0, dersKayitTemp->ders\_id, TYPE\_OF\_DERS));

dersTemp = dersHead;

while(strcmp(dersTemp->id, dersKayitTemp->ders\_id))

dersTemp = dersTemp->next;

//Girilen verilere göre kaydı bul

dersKayitTemp2 = dersKayitHead;

tempNumber = 1;

while(dersKayitTemp2 != NULL && tempNumber){

if(dersKayitTemp2->ogrenci\_id == dersKayitTemp->ogrenci\_id)

if(!strcmp(dersKayitTemp2->ders\_id, dersKayitTemp->ders\_id))

//if(!strcmp(dersKayitTemp2->kayit\_tarihi, dersKayitTemp->kayit\_tarihi))

tempNumber = 0;

if(tempNumber)

dersKayitTemp2 = dersKayitTemp2->next;

}

if(dersKayitTemp2 != NULL){

printf("\n----\nID: %d\nÖğrenci ID: %d\nDers ID: %s\nKayıt Durumu: ", dersKayitTemp2->id, dersKayitTemp2->ogrenci\_id, dersKayitTemp2->ders\_id);

(dersKayitTemp2->kayit\_durumu == 1) ? printf("KAYITLI\n") : printf("BIRAKTI\n");

printf("Kayıt Tarihi: %s\n----", dersKayitTemp2->kayit\_tarihi);

do{

printf("\nKayıt Durumu (1->KAYITLI, 0->BIRAKTI): ");

scanf("%d", &tempNumber);

}while(tempNumber != 0 && tempNumber != 1);

if(dersKayitTemp2->kayit\_durumu > tempNumber){//Yani KAYITLIDAN->BIRAKTI yapıldıysa, öğrencinin dersi adedini ve kredisini düşür, dersin kontenjanını bir azalt

//Öğrenci-Kayit işlemleri

dersKayitTemp2->kayit\_durumu = 0;

tempVoid = &dersKayitHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERSKAYIT, DERSKAYITYOL);

//Öğrenci işlemleri

ogrenciTemp->toplam\_ders -= 1;

ogrenciTemp->toplam\_kredi -= dersTemp->kredi;

tempVoid = &ogrenciHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_OGRENCI, OGRENCIYOL);

//Ders işlemleri

dersTemp->kontenjan -= 1;

tempVoid = &dersHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERS, DERSYOL);

printf("Güncelleme işleminiz başarılı!");

}

else if(dersKayitTemp2->kayit\_durumu < tempNumber){//Yani BIRAKTIDAN->KAYITLI yapıldıysa

if(ogrenciTemp->toplam\_ders + 1 <= ayarlar->dersSinir && ogrenciTemp->toplam\_kredi + dersTemp->kredi <= ayarlar->krediSinir && dersTemp->kontenjan + 1 <= ayarlar->dersKontenjan){

//Kayıt yapılabilir

dersKayitTemp2->kayit\_durumu = 1;

tempVoid = &dersKayitHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERSKAYIT, DERSKAYITYOL);

//Öğrenci işlemleri

ogrenciTemp->toplam\_ders += 1;

ogrenciTemp->toplam\_kredi += dersTemp->kredi;

tempVoid = &ogrenciHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_OGRENCI, OGRENCIYOL);

//Ders işlemleri

dersTemp->kontenjan += 1;

tempVoid = &dersHead;

STRUCTTOFILE(tempVoid, TYPE\_OF\_DERS, DERSYOL);

printf("Güncelleme işleminiz başarılı!");

}

else{//Kayıt yapılamaz

printf("Öğrenci derse kayıt olamaz. Olası sebepler;\n\* Öğrenci ders sınırını doldurmuş\n\* Öğrenci kredi sınırını doldurmuş\n\* Dersin kontenjanı dolmuş olabilir!");

}

}

else

printf("Değişiklik yapılmamıştır!");

}

else{

printf("Kayit Bulunamamıştır!");

}

free(dersKayitTemp);

printf("\nDevam etmek için bir tuşa basın!!!");

getch();

}

else if(secim == 12){

system("cls");

printf("\*\*\*ÖĞRETMEN DERS LİSTELEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(ogretmenHead, TYPE\_OF\_OGRETMEN);

tempVoid = &ogretmenHead;

do{

printf("\nÖğretmen ID (çıkış için -1): ");

scanf("%d", &tempNumber);

if(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1)

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1);

if(tempNumber != -1)

VERILENDERSLER(dersHead, tempNumber);

printf("\n\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 13){

system("cls");

printf("\*\*\*DERSİ ALAN ÖĞRENCİLERİ LİSTELEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(dersHead, TYPE\_OF\_DERS);

tempVoid = &dersHead;

do{

printf("\nDers ID (çıkış için -1): ");

scanf("%s", tempChar);

if(!IS\_ID\_EXIST(tempVoid, 0, tempChar, TYPE\_OF\_DERS) && strcmp(tempChar, "-1"))

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, 0, tempChar, TYPE\_OF\_DERS) && strcmp(tempChar, "-1"));

if(strcmp(tempChar, "-1"))

DERSIALANLAR(dersKayitHead, ogrenciHead, tempChar);

printf("\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 14){

system("cls");

printf("\*\*\*ÖĞRENCİ DERSLERİ LİSTELEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(ogrenciHead, TYPE\_OF\_OGRENCI);

tempVoid = &ogrenciHead;

do{

printf("\nÖğrenci ID (çıkış için -1): ");

scanf("%d", &tempNumber);

if(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRENCI) && tempNumber != -1)

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRENCI) && tempNumber != -1);

if(tempNumber != -1)

OGRENCIDERSLISTELE(dersKayitHead, dersHead, tempNumber);

printf("\nDevam etmek için bir tuşa basın...");

getch();

}

else if(secim == 15){

system("cls");

printf("\*\*\*ÖĞRETMEN DERS LİSTELEME SAYFASINA HOŞGELDİNİZ\*\*\*\n\n");

SHOWSTRUCT(ogretmenHead, TYPE\_OF\_OGRETMEN);

tempVoid = &ogretmenHead;

do{

printf("\nÖğretmen ID (çıkış için -1): ");

scanf("%d", &tempNumber);

if(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1)

printf("Sistemde kayıtlı olmayan bir ID girdiniz. Tekrar deneyiniz!\n");

}while(!IS\_ID\_EXIST(tempVoid, tempNumber, NULL, TYPE\_OF\_OGRETMEN) && tempNumber != -1);

if(tempNumber != -1)

SINIFLISTESITXT(dersHead, dersKayitHead, ogrenciHead, tempNumber);

printf("\nDevam etmek için bir tuşa basın...");

getch();

}

}

return 0;

}

void SINIFLISTESITXT(DERS \*dersHead, OGRENCIDERSKAYIT \*dersKayitHead, OGRENCI \*ogrenciHead, int ogretmen\_id){

DERS \*tempDers = dersHead;

OGRENCIDERSKAYIT \*tempDersKayit;

OGRENCI \*tempOgrenci;

FILE \*dosya;

char liste\_adi[27];

printf("İşlem sürüyor..\n");

while(tempDers != NULL){

if(tempDers->ogretmen\_id == ogretmen\_id){

tempDersKayit = dersKayitHead;

strcpy(liste\_adi, tempDers->id);

strcat(liste\_adi, "\_SINIFLISTESI.txt");

OPENFILE(&dosya, liste\_adi, "w");

while(tempDersKayit != NULL){

if(!strcmp(tempDersKayit->ders\_id, tempDers->id)){

tempOgrenci = ogrenciHead;

while(tempOgrenci->id != tempDersKayit->ogrenci\_id)

tempOgrenci = tempOgrenci->next;

fprintf(dosya, "%d %s %s\n", tempOgrenci->id, tempOgrenci->adi, tempOgrenci->soyadi);

}

tempDersKayit = tempDersKayit->next;

}

fclose(dosya);

printf("%s dosyası oluşturuldu!\n", liste\_adi);

}

tempDers = tempDers->next;

}

printf("İşlem halledildi!\n");

}

void OGRENCIDERSLISTELE(OGRENCIDERSKAYIT \*dersKayitHead, DERS \*dersHead, int ogrenci\_id){

OGRENCIDERSKAYIT \*tempDersKayit = dersKayitHead;

DERS \*tempDers = dersHead;

printf("\nÖğrencinin aldığı dersler;\n");

while(tempDersKayit != NULL){

if(tempDersKayit->ogrenci\_id == ogrenci\_id){

tempDers = dersHead;

while(strcmp(tempDersKayit->ders\_id, tempDers->id))

tempDers = tempDers->next;

printf("%s %s\n", tempDers->id, tempDers->adi);

}

tempDersKayit = tempDersKayit->next;

}

}

void DERSIALANLAR(OGRENCIDERSKAYIT \*dersKayitHead, OGRENCI \*ogrenciHead, char \*ders\_id){

OGRENCIDERSKAYIT \*tempDersKayit = dersKayitHead;

OGRENCI \*tempOgrenci = ogrenciHead;

while(tempDersKayit != NULL){

if(!strcmp(tempDersKayit->ders\_id, ders\_id)){

tempOgrenci = ogrenciHead;

while(tempDersKayit->ogrenci\_id != tempOgrenci->id)

tempOgrenci = tempOgrenci->next;

printf("%d %s %s\n", tempOgrenci->id, tempOgrenci->adi, tempOgrenci->soyadi);

}

tempDersKayit = tempDersKayit->next;

}

}

void VERILENDERSLER(DERS \*dersHead, int ogretmen\_id){

DERS \*tempDers = dersHead;

while(tempDers != NULL){

if(tempDers->ogretmen\_id == ogretmen\_id)

printf("\nDersin Adı: %s", tempDers->adi);

tempDers = tempDers->next;

}

}

void UPDATEBYID(void \*\*headP, void\*\* headP2, int ID, char \*IDC, int choise, char \*path){

OGRENCI \*tempOgrenci;

OGRETMEN \*tempOgretmen;

DERS \*tempDers;

OGRENCIDERSKAYIT \*tempKayit;

if(choise == TYPE\_OF\_OGRENCI){

tempOgrenci = \*headP;

while(tempOgrenci->id != ID)

tempOgrenci = tempOgrenci->next;

printf("----\nÖğrenci ID: %d\nÖğrenci Adı: %s\nÖğrenci Soyadı: %s\nÖğrenci Toplam Ders: %d\nÖğrenci Toplam Kredi: %d\n----\n", ID, tempOgrenci->adi, tempOgrenci->soyadi, tempOgrenci->toplam\_ders, tempOgrenci->toplam\_kredi);

printf("Öğrenci Adı: "); scanf("%s", tempOgrenci->adi);

printf("Öğrenci Soyadı: "); scanf("%s", tempOgrenci->soyadi);

}else if(choise == TYPE\_OF\_OGRETMEN){

tempOgretmen = \*headP;

while(tempOgretmen->id != ID)

tempOgretmen = tempOgretmen->next;

printf("----\nÖğretmen ID: %d\nÖğretmen Adı: %s\nÖğretmen Soyadı: %s\nÖğretmen Unvan: %s\n----\n", ID, tempOgretmen->adi, tempOgretmen->soyadi, tempOgretmen->unvan);

printf("Öğretmen Adı: "); scanf("%s", tempOgretmen->adi);

printf("Öğretmen Soyadı: "); scanf("%s", tempOgretmen->soyadi);

printf("Öğretmen Unvan: "); scanf("%s", tempOgretmen->unvan);

}else if(choise == TYPE\_OF\_DERS){

tempDers = \*headP;

while(strcmp(tempDers->id, IDC))

tempDers = tempDers->next;

printf("----\nDers ID: %s\nDers Adı: %s\nDers Kredisi: %d\nDers Kontenjanı: %d\nDers Öğreticisi ID: %d\n----\n", IDC, tempDers->adi, tempDers->kredi, tempDers->kontenjan, tempDers->ogretmen\_id);

printf("Ders Adı: "); scanf("%s", tempDers->adi);

do{

printf("Öğretmen ID (yoksa -1): ");

scanf("%d", &tempDers->ogretmen\_id);

}while(!IS\_ID\_EXIST(headP2, tempDers->ogretmen\_id, NULL, TYPE\_OF\_OGRETMEN) && tempDers->ogretmen\_id != -1);

}else if(choise == TYPE\_OF\_DERSKAYIT){

tempKayit = \*headP;

while(tempKayit->id != ID)

tempKayit = tempKayit->next;

printf("----\nID: %d\nÖğrenci ID: %d\nDers ID: %s\nKayıt Durumu: ", tempKayit->id, tempKayit->ogrenci\_id, tempKayit->ders\_id);

(tempKayit->kayit\_durumu == 0) ? printf("BIRAKTI (0)\n") : printf("KAYITLI (1)\n");

printf("Kayıt Tarihi: %s\n----\n", tempKayit->kayit\_tarihi);

do{

printf("Kayıt Durumu (0->BIRAKTI, 1->KAYITLI): ");

scanf("%d", &tempKayit->kayit\_durumu);

}while(tempKayit->kayit\_durumu != 0 && tempKayit->kayit\_durumu != 1);

}

STRUCTTOFILE(headP, choise, path);

}

void DELETEBYID(void \*\*headP, int ID, char \*IDC, int choise, char \*path){//Ders, Öğretmen ve Öğrenci kayıt silme

OGRENCI \*tempOgrenci, \*trashOgrenci;

OGRETMEN \*tempOgretmen, \*trashOgretmen;

DERS \*tempDers, \*trashDers;

OGRENCIDERSKAYIT \*tempDersKayit, \*trashDersKayit;

if(choise == TYPE\_OF\_OGRENCI){//

tempOgrenci = \*headP;

if(tempOgrenci->id == ID){//İlk eleman silinecek ise

trashOgrenci = tempOgrenci;

tempOgrenci = tempOgrenci->next;

\*headP = tempOgrenci;

free(trashOgrenci);

}else{//ortada veya son eleman silinecekse

while(tempOgrenci->next->id != ID)//Bulunamama ihtimali olmadığı için o koşulu yazmadık

tempOgrenci = tempOgrenci->next;

trashOgrenci = tempOgrenci->next;

if(tempOgrenci->next->next == NULL)//son elemanı siliyorsak

tempOgrenci->next = NULL;

else

tempOgrenci->next = tempOgrenci->next->next;

free(trashOgrenci);

}

}

else if(choise == TYPE\_OF\_OGRETMEN){

tempOgretmen = \*headP;

if(tempOgretmen->id == ID){//İlk eleman silinecek ise

trashOgretmen = tempOgretmen;

tempOgretmen = tempOgretmen->next;

\*headP = tempOgretmen;

free(trashOgretmen);

}else{//ortada veya son eleman silinecekse

while(tempOgretmen->next->id != ID)//Bulunamama ihtimali olmadığı için o koşulu yazmadık

tempOgretmen = tempOgretmen->next;

trashOgretmen = tempOgretmen->next;

if(tempOgretmen->next->next == NULL)//son elemanı siliyorsak

tempOgretmen->next = NULL;

else

tempOgretmen->next = tempOgretmen->next->next;

free(trashOgretmen);

}

}else if(choise == TYPE\_OF\_DERS){

tempDers = \*headP;

if(!strcmp(tempDers->id, IDC)){//İlk eleman silinecek ise

trashDers = tempDers;

tempDers = tempDers->next;

\*headP = tempDers;

free(trashDers);

}else{//ortada veya son eleman silinecekse

while(strcmp(tempDers->next->id, IDC))//Bulunamama ihtimali olmadığı için o koşulu yazmadık

tempDers = tempDers->next;

trashDers = tempDers->next;

if(tempDers->next->next == NULL)//son elemanı siliyorsak

tempDers->next = NULL;

else

tempDers->next = tempDers->next->next;

free(trashDers);

}

}

else if(choise == TYPE\_OF\_DERSKAYIT){

tempDersKayit = \*headP;

if(tempDersKayit->id == ID){//İlk eleman silinecek ise

trashDersKayit = tempDersKayit;

tempDersKayit = tempDersKayit->next;

\*headP = tempDersKayit;

free(trashDersKayit);

}else{//ortada veya son eleman silinecekse

while(tempDersKayit->next->id != ID)//Bulunamama ihtimali olmadığı için o koşulu yazmadık

tempDersKayit = tempDersKayit->next;

trashDersKayit = tempDersKayit->next;

if(tempDersKayit->next->next == NULL)//son elemanı siliyorsak

tempDersKayit->next = NULL;

else

tempDersKayit->next = tempDersKayit->next->next;

free(trashDersKayit);

}

}

STRUCTTOFILE(headP, choise, path);

}

void STRUCTTOFILE(void \*\*headP, int choise, char \*path){

FILE \*dosya;

OGRENCI \*tempOgrenci;

OGRETMEN \*tempOgretmen;

DERS \*tempDers;

OGRENCIDERSKAYIT \*tempKayit;

AYARLAR \*tempAyarlar;

OPENFILE(&dosya, path, "w");

if(choise == TYPE\_OF\_OGRENCI){

tempOgrenci = \*headP;

while(tempOgrenci != NULL){

fprintf(dosya, "\n%d %s %s %d %d", tempOgrenci->id, &tempOgrenci->adi, &tempOgrenci->soyadi, tempOgrenci->toplam\_ders, tempOgrenci->toplam\_kredi);

tempOgrenci = tempOgrenci->next;

}

}else if(choise == TYPE\_OF\_OGRETMEN){

tempOgretmen = \*headP;

while(tempOgretmen != NULL){

fprintf(dosya, "\n%d %s %s %s", tempOgretmen->id, &tempOgretmen->adi, &tempOgretmen->soyadi, &tempOgretmen->unvan);

tempOgretmen = tempOgretmen->next;

}

}else if(choise == TYPE\_OF\_DERS){

tempDers = \*headP;

while(tempDers != NULL){

fprintf(dosya, "\n%s %s %d %d %d", &tempDers->id, &tempDers->adi, tempDers->kredi, tempDers->kontenjan, tempDers->ogretmen\_id);

tempDers = tempDers->next;

}

}else if(choise == TYPE\_OF\_DERSKAYIT){

tempKayit = \*headP;

while(tempKayit != NULL){

fprintf(dosya, "\n%d %d %s %d %s", tempKayit->id, tempKayit->ogrenci\_id, &tempKayit->ders\_id, tempKayit->kayit\_durumu, &tempKayit->kayit\_tarihi);

tempKayit = tempKayit->next;

}

}else if(choise == TYPE\_OF\_AYARLAR){

tempAyarlar = \*headP;

fprintf(dosya, "%d %d %d %d", tempAyarlar->dersSinir, tempAyarlar->krediSinir, tempAyarlar->dersKayitAutoINC, tempAyarlar->dersKontenjan);

}

fclose(dosya);

}

int IS\_ID\_EXIST(void \*\*headP, int ID, char \*IDC, int choise){//Varsa ise 1, yoksa 0

OGRENCI \*tempOgrenci;

OGRETMEN \*tempOgretmen;

DERS \*tempDers;

if(choise == TYPE\_OF\_OGRENCI){

tempOgrenci = \*headP;

while(tempOgrenci != NULL && tempOgrenci->id != ID)

tempOgrenci = tempOgrenci->next;

if(tempOgrenci != NULL) return 1;

else return 0;

}else if(choise == TYPE\_OF\_OGRETMEN){

tempOgretmen = \*headP;

while(tempOgretmen != NULL && tempOgretmen->id != ID)

tempOgretmen = tempOgretmen->next;

if(tempOgretmen != NULL) return 1;

else return 0;

}else if(choise == TYPE\_OF\_DERS){

tempDers = \*headP;

while(tempDers != NULL && strcmp(tempDers->id, IDC) != 0)

tempDers = tempDers->next;

if(tempDers != NULL) return 1;

else return 0;

}else if(choise == TYPE\_OF\_DERSOGRETMENKONTROL){

tempDers = \*headP;

while(tempDers != NULL && tempDers->ogretmen\_id != ID)

tempDers = tempDers->next;

if(tempDers != NULL) return 1;

else return 0;

}

}

void LOADSTRUCT(OGRENCI \*\*ogrPtr, OGRETMEN \*\*ogretmenPtr, DERS \*\*dersPtr, OGRENCIDERSKAYIT \*\*kayitPtr, char \*ogrYol, char \*ogretYol, char \*dersYol, char \*kayitYol){

FILE \*dosya;

void \*p;//Warning oluşmaması için oluşturuldu

if(ISFILEEXIST(ogrYol) == FILE\_NONE){

dosya = fopen(ogrYol, "w");

fclose(dosya);

}

p = \*ogrPtr;

FILETOSTRUCT(&p, TYPE\_OF\_OGRENCI, ogrYol);

\*ogrPtr = p;

if(ISFILEEXIST(ogretYol) == FILE\_NONE){

dosya = fopen(ogretYol, "w");

fclose(dosya);

}

p = \*ogretmenPtr;

FILETOSTRUCT(&p, TYPE\_OF\_OGRETMEN, ogretYol);

\*ogretmenPtr = p;

if(ISFILEEXIST(dersYol) == FILE\_NONE){

dosya = fopen(dersYol, "w");

fclose(dosya);

}

p = \*dersPtr;

FILETOSTRUCT(&p, TYPE\_OF\_DERS, dersYol);

\*dersPtr = p;

if(ISFILEEXIST(kayitYol) == FILE\_NONE){

dosya = fopen(kayitYol, "w");

fclose(dosya);

}

p = \*kayitPtr;

FILETOSTRUCT(&p, TYPE\_OF\_DERSKAYIT, kayitYol);

\*kayitPtr = p;

}

void FILETOSTRUCT(void \*\*headP, int choise, char \*path){

FILE \*dosya;

OGRENCI \*tempOgrenci;

OGRETMEN \*tempOgretmen;

DERS \*tempDers;

OGRENCIDERSKAYIT \*tempKayit;

OPENFILE(&dosya, path, "r");

if(choise == TYPE\_OF\_OGRENCI){

if(ISFILEEXIST(path) != FILE\_EMPTY)

while(!feof(dosya)){

tempOgrenci = (OGRENCI \*) malloc(sizeof(OGRENCI));

fscanf(dosya, "%d %s %s %d %d", &tempOgrenci->id, tempOgrenci->adi, tempOgrenci->soyadi, &tempOgrenci->toplam\_ders, &tempOgrenci->toplam\_kredi);

tempOgrenci->next = \*headP;

\*headP = tempOgrenci;

}

}else if(choise == TYPE\_OF\_OGRETMEN){

if(ISFILEEXIST(path) != FILE\_EMPTY)

while(!feof(dosya)){

tempOgretmen = (OGRETMEN \*) malloc(sizeof(OGRETMEN));

fscanf(dosya, "%d %s %s %s", &tempOgretmen->id, tempOgretmen->adi, tempOgretmen->soyadi, tempOgretmen->unvan);

tempOgretmen->next = \*headP;

\*headP = tempOgretmen;

}

}else if(choise == TYPE\_OF\_DERS){

if(ISFILEEXIST(path) != FILE\_EMPTY)

while(!feof(dosya)){

tempDers = (DERS \*) malloc(sizeof(DERS));

fscanf(dosya, "%s %s %d %d %d", tempDers->id, tempDers->adi, &tempDers->kredi, &tempDers->kontenjan, &tempDers->ogretmen\_id);

tempDers->next = \*headP;

\*headP = tempDers;

}

}else if(choise == TYPE\_OF\_DERSKAYIT){

if(ISFILEEXIST(path) != FILE\_EMPTY)

while(!feof(dosya)){

tempKayit = (OGRENCIDERSKAYIT \*) malloc(sizeof(OGRENCIDERSKAYIT));

fscanf(dosya, "%d %d %s %d %s", &tempKayit->id, &tempKayit->ogrenci\_id, tempKayit->ders\_id, &tempKayit->kayit\_durumu, tempKayit->kayit\_tarihi);

tempKayit->next = \*headP;

\*headP = tempKayit;

}

}

fclose(dosya);

}

int ISFILEEXIST(char \*yol){

FILE \*dosya;

if((dosya = fopen(yol, "r"))){

fseek(dosya, 0, SEEK\_END);

if(ftell(dosya) == 0){

fclose(dosya);

return FILE\_EMPTY;//Dosya var ama boş

}else{

fclose(dosya);

return FILE\_EXIST;//Dosya dolu ve var

}

}

else return FILE\_NONE;//Dosya yok

}

void OPENFILE(FILE \*\*dosya, char \*yol, char \*tur){

if((\*dosya = fopen(yol, tur)) == NULL){

if(tur == "r")

printf("Okumak istediğiniz dosya açılırken hatayla karşılaşıldı!");

else

printf("Dosya açılırken hatayla karşılaşıldı!");

exit(1);

}

}

void SHOWSTRUCT(void \*pointer, int choise){

OGRENCI \*tempOgrenci;

OGRETMEN \*tempOgretmen;

DERS \*tempDers;

OGRENCIDERSKAYIT \*tempKayit;

if(choise == TYPE\_OF\_OGRENCI){

tempOgrenci = pointer;

printf("Öğrenci Listesi\nID\tADI\tSOYADI\tTOPLAM DERS\tTOPLAM KREDİ\n");

while(tempOgrenci != NULL){

printf("%-10d%-20s%-20s%-8d%d\n", tempOgrenci->id, tempOgrenci->adi, tempOgrenci->soyadi, tempOgrenci->toplam\_ders, tempOgrenci->toplam\_kredi);

tempOgrenci = tempOgrenci->next;

}

}

else if(choise == TYPE\_OF\_OGRETMEN){

tempOgretmen = pointer;

printf("Öğretmen Listesi\nID\tADI\tSOYADI\tUNVAN\n");

while(tempOgretmen != NULL){

printf("%-10d%-15s%-15s%s\n", tempOgretmen->id, tempOgretmen->adi, tempOgretmen->soyadi, tempOgretmen->unvan);

tempOgretmen = tempOgretmen->next;

}

}

else if(choise == TYPE\_OF\_DERS){

tempDers = pointer;

printf("Ders Listesi\nID\tADI\tKREDI\tKONTENJAN\tOGRETMEN ID\n");

while(tempDers != NULL){

printf("%-10s%-30s%-8d%-8d%d\n", tempDers->id, tempDers->adi, tempDers->kredi, tempDers->kontenjan, tempDers->ogretmen\_id);

tempDers = tempDers->next;

}

}

else if(choise == TYPE\_OF\_DERSKAYIT){

tempKayit = pointer;

printf("DERS-KAYIT Listesi\nID\tOGRENCI ID\tDERS ID\tKAYIT DURUMU\tKAYIT TARİHİ\n");

while(tempKayit != NULL){

printf("%-10d%-10d%-10s ", tempKayit->id, tempKayit->ogrenci\_id, tempKayit->ders\_id);

(tempKayit->kayit\_durumu == 0) ? printf("BIRAKTI ") : printf("KAYITLI ");

printf("%s\n", tempKayit->kayit\_tarihi);

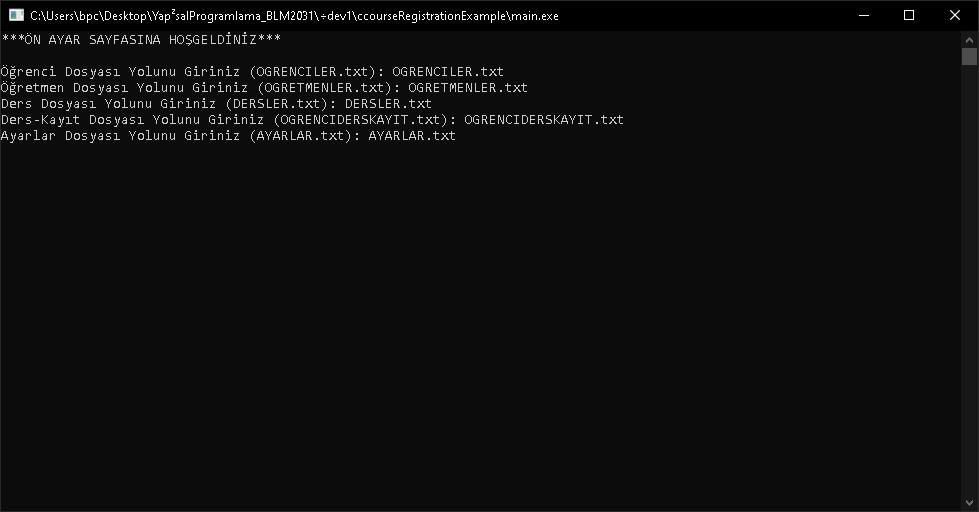
tempKayit = tempKayit->next;

}

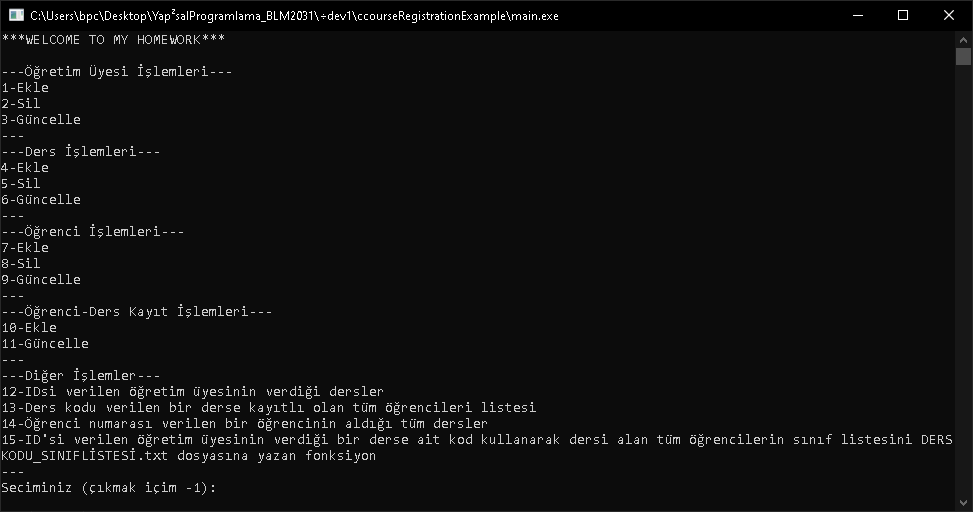
}

}

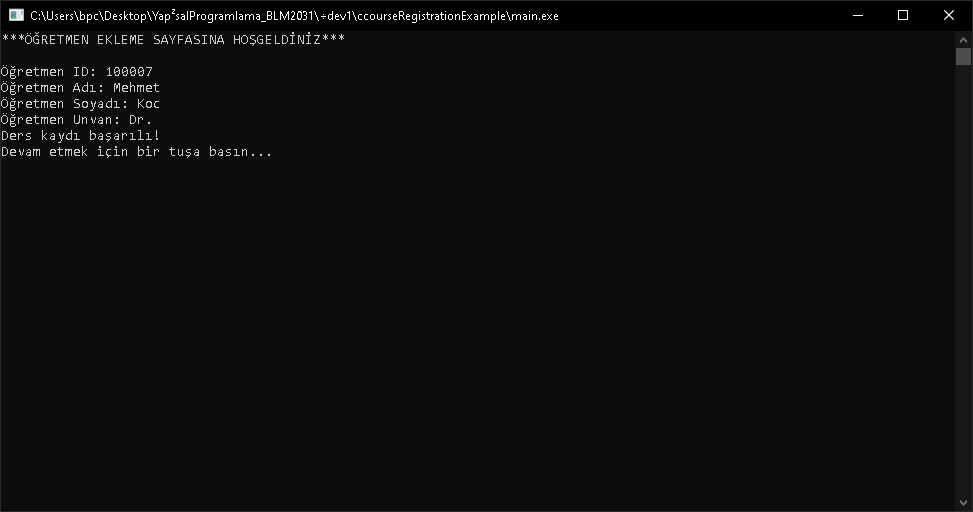
**RESİMLİ ÇIKTILAR**

****

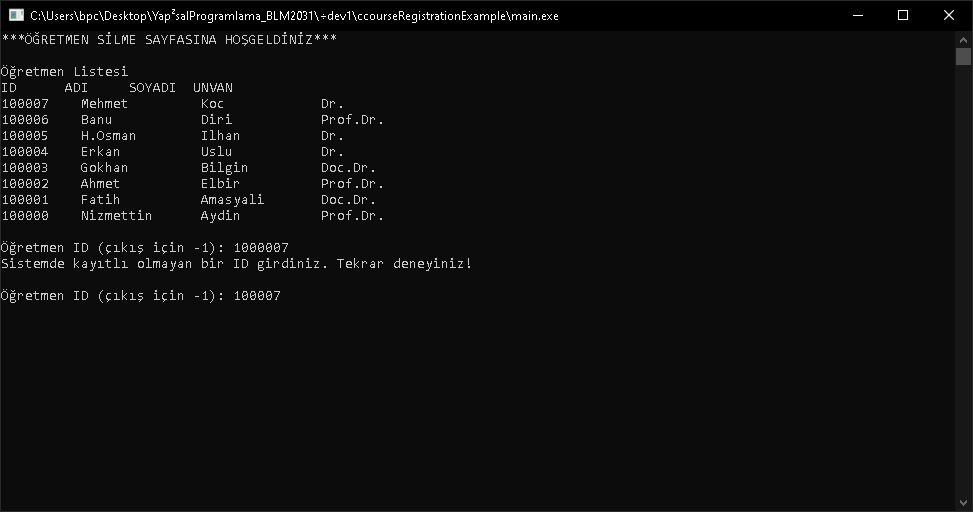
Resim 1: Program ilk açıldığın da gelen ekran / dosya isimlerinin girildiği yer



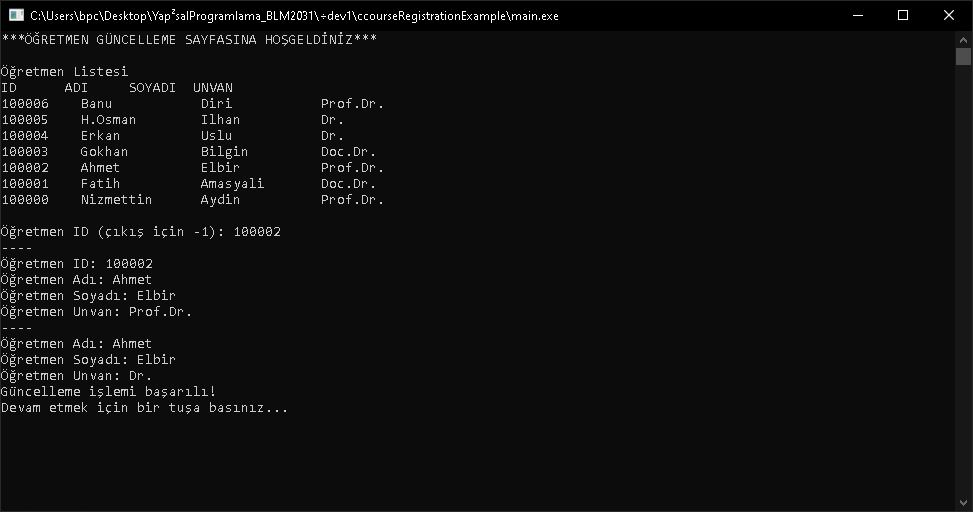
Resim 2:Programın ana menüsü



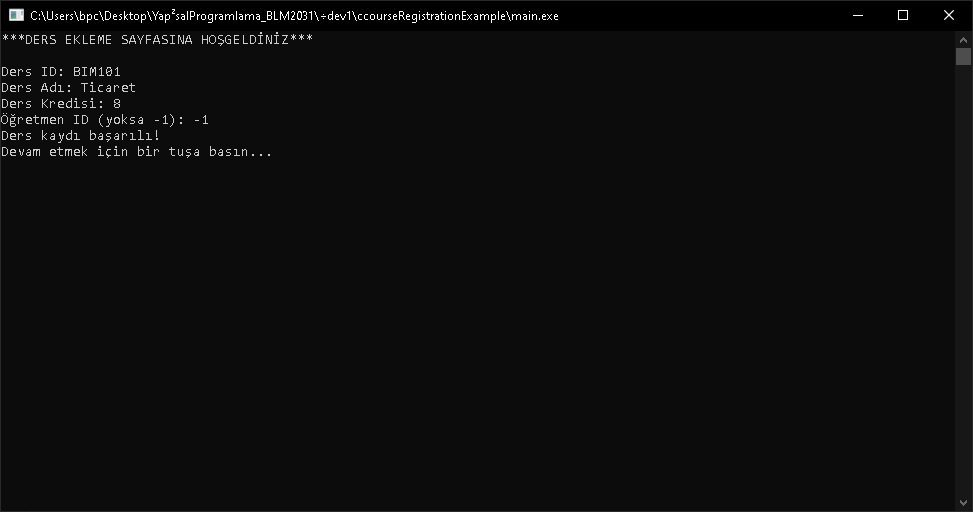
Resim 3: Öğretmen ekleme sayfası



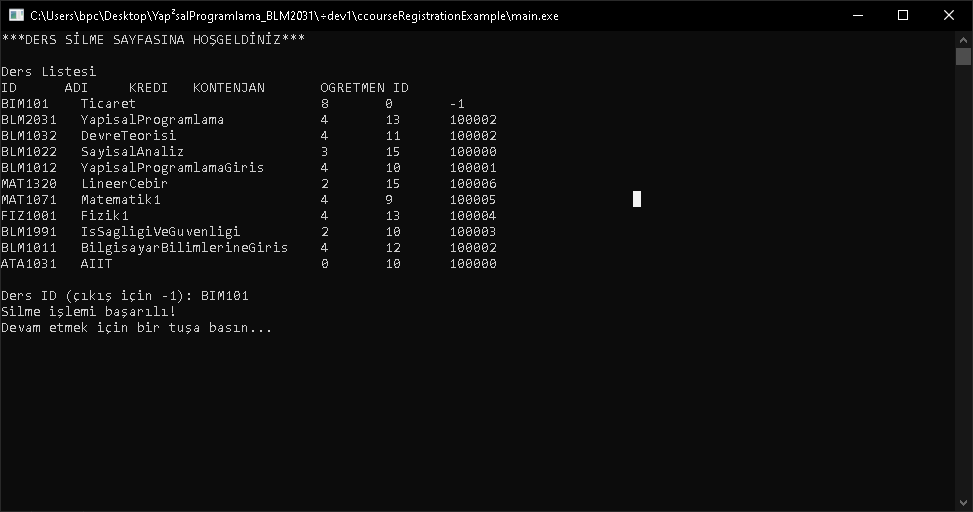
Resim 4: Öğretmen silme ekranı



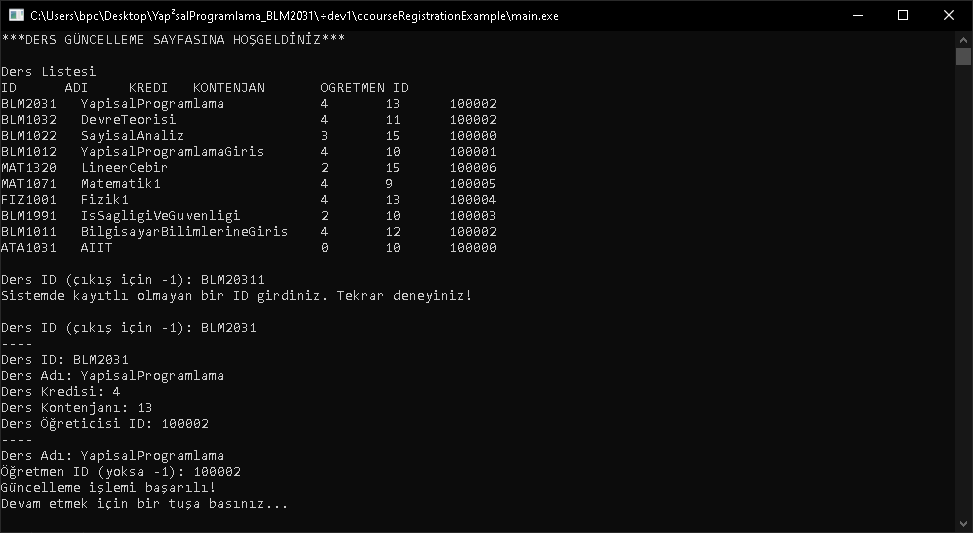
Resim 5: Öğretmen güncelleme ekranı



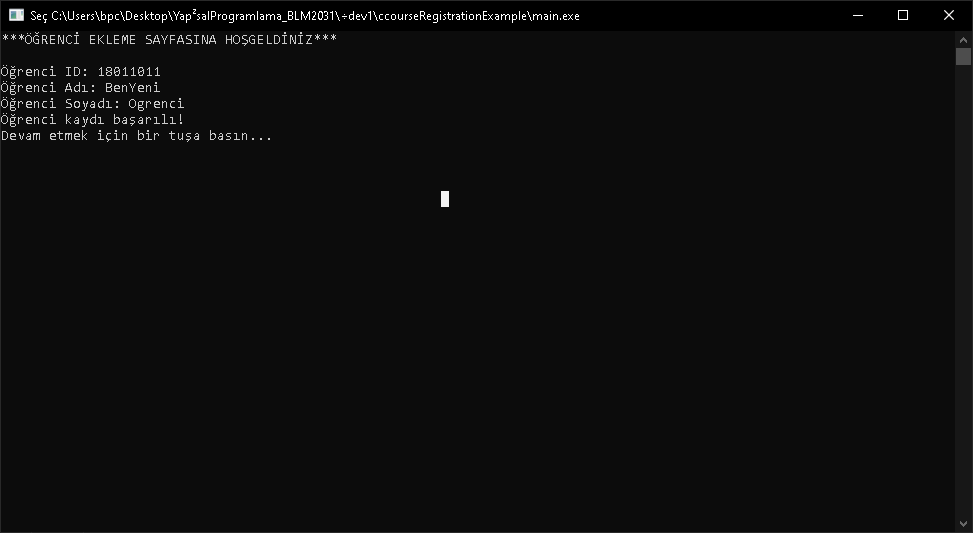
Resim 6: Ders ekleme ekranı



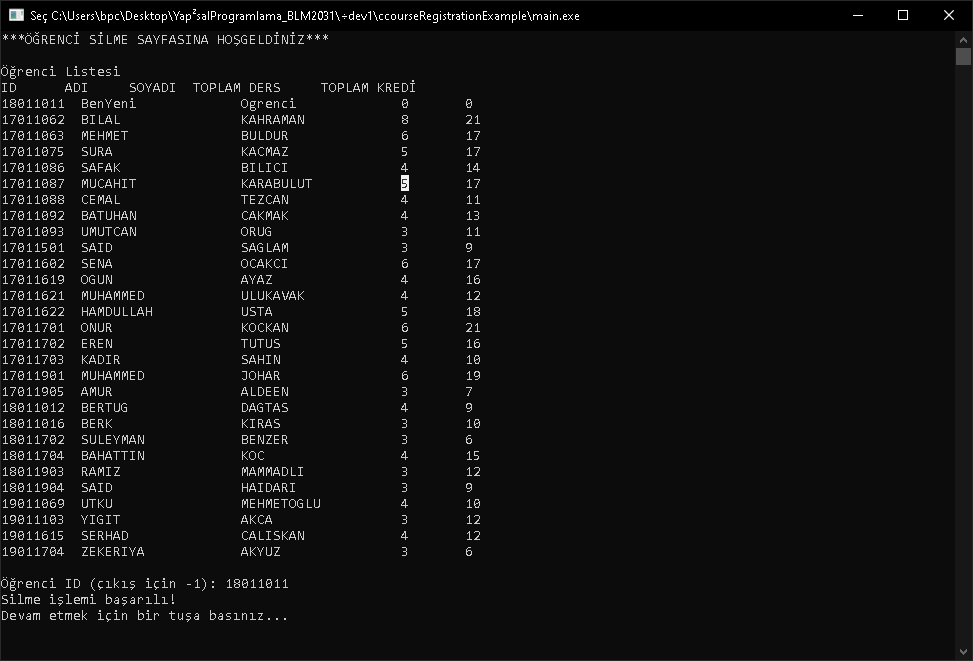
Resim 7: Ders silme ekranı



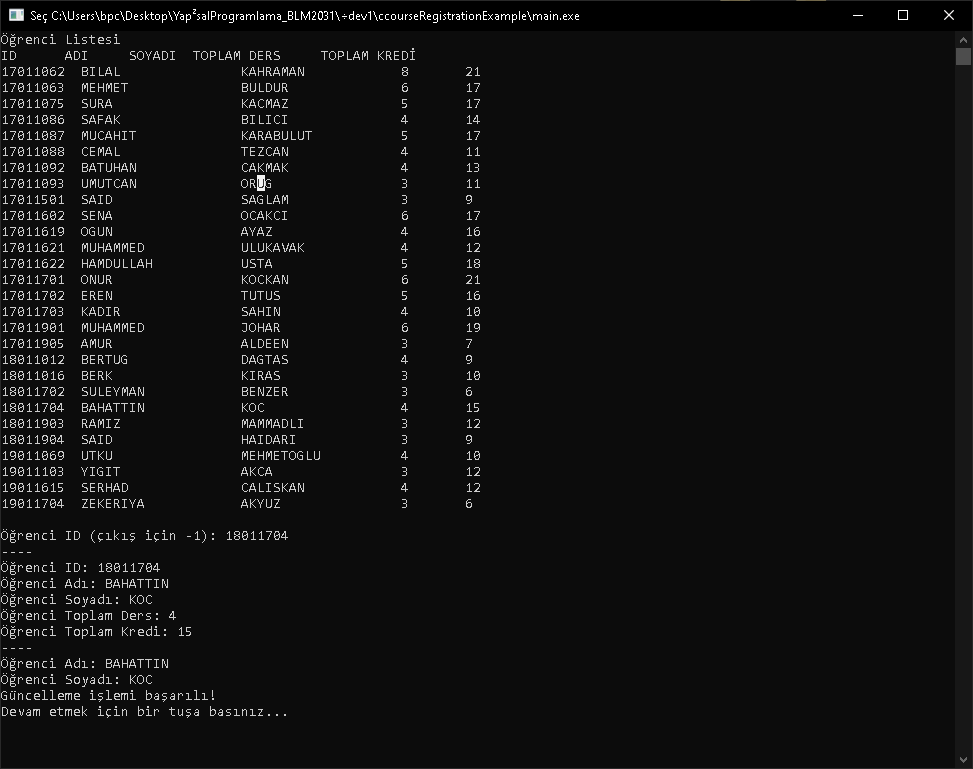
Resim 8: Ders güncelleme ekranı



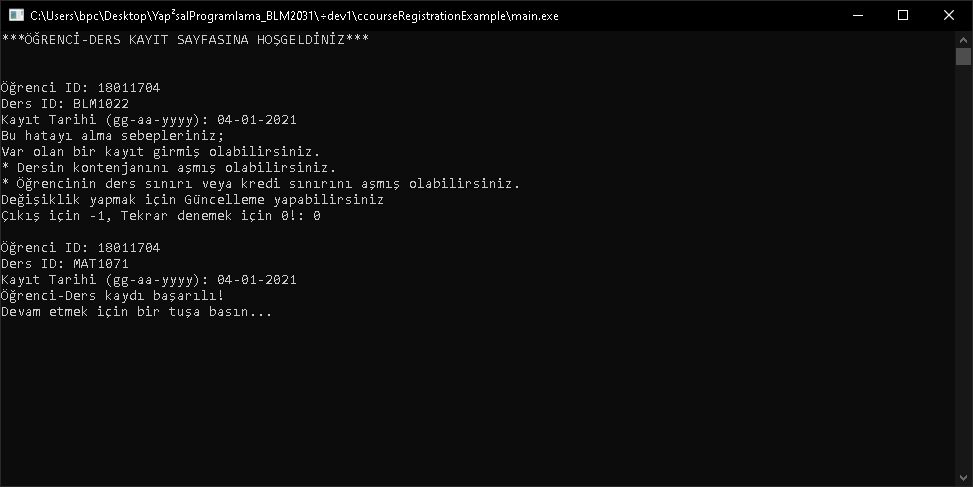
Resim 9: Öğrenci ekleme ekranı



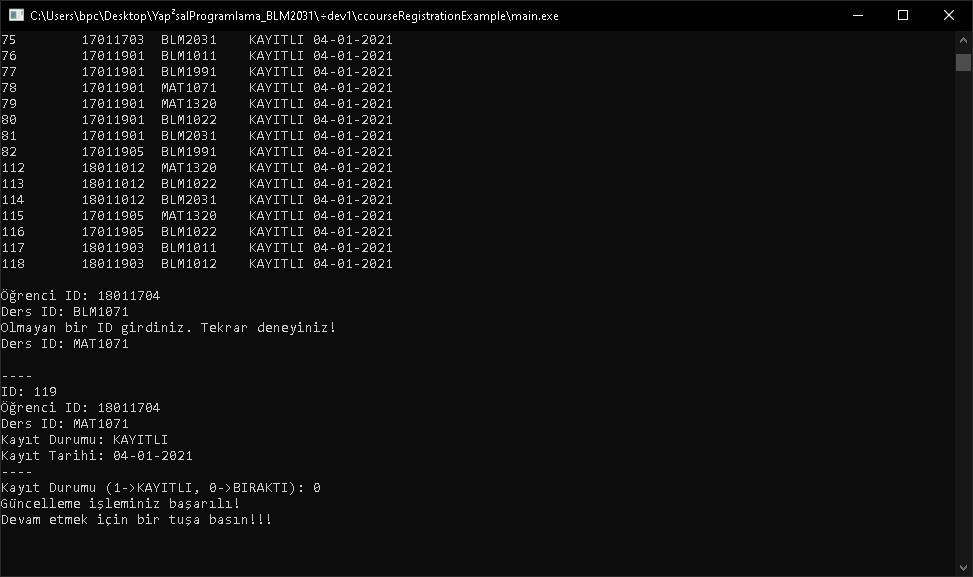
Resim 10: Öğrenci silme ekranı



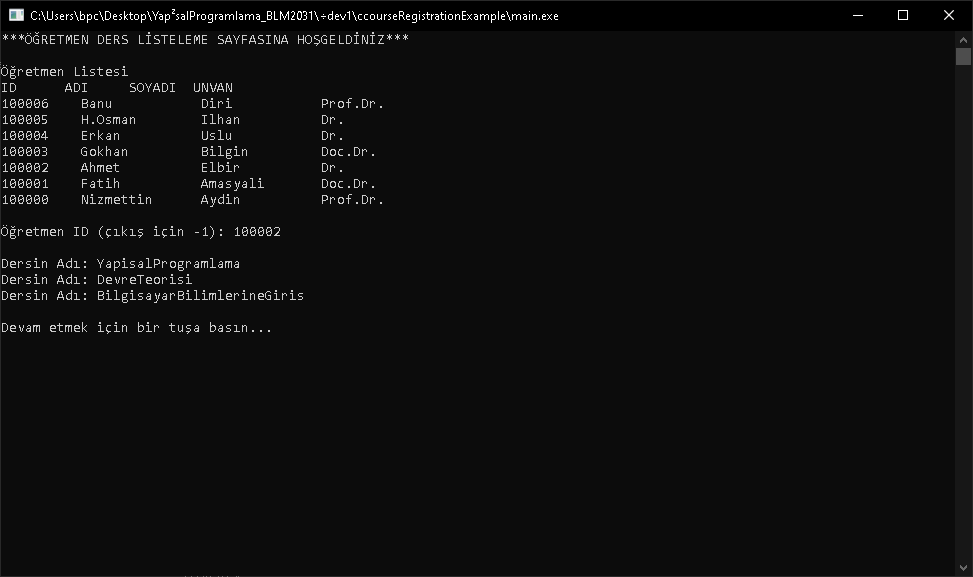
Resim 11: Öğrenci güncelleme ekranı



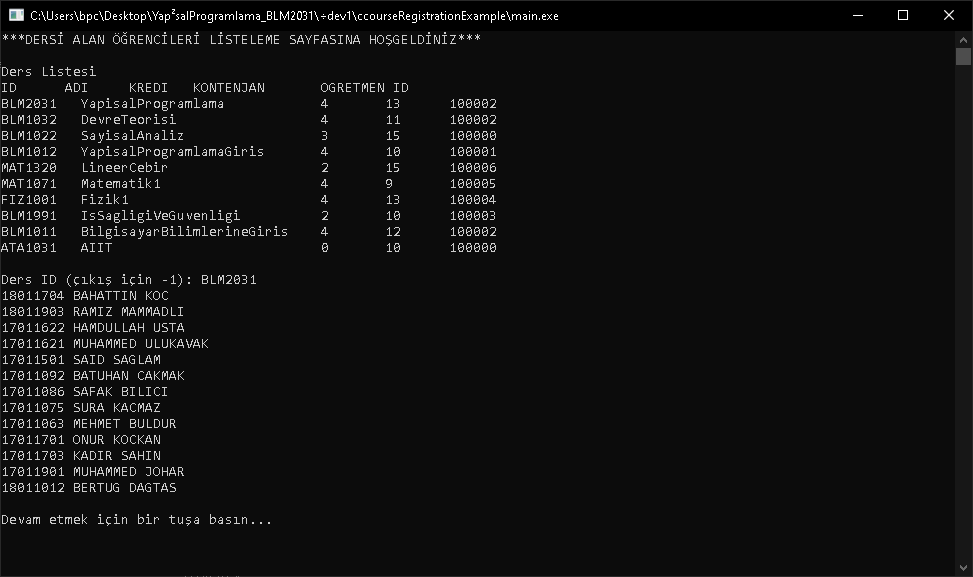
Resim 12: Öğrenci-Ders kayıt ekranı



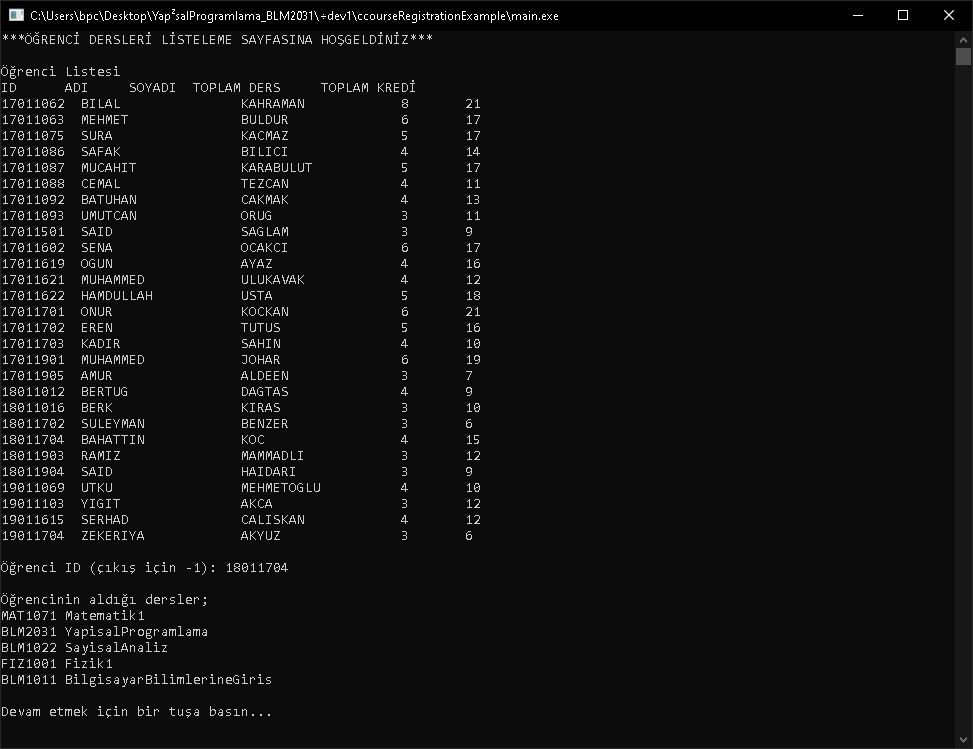
Resim 13: Öğrenci-Ders kayıt güncelleme ekranı



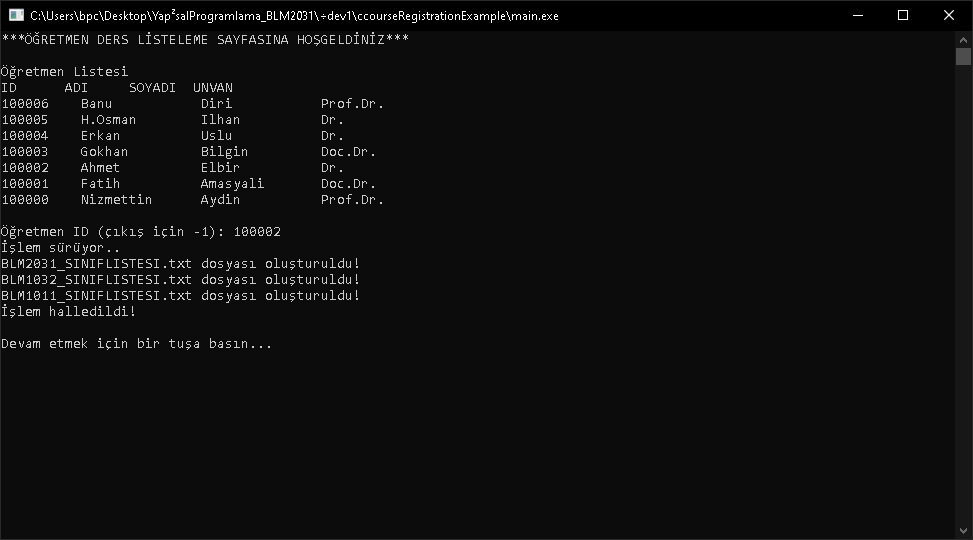
Resim 14: Öğretmen derslerini listeleme ekranı



Resim 15: Dersli alan öğrenci listesi ekranı



Resim 16: Öğrencinin aldığı dersleri listeleme ekranı



Resim 17: Numarası verilen öğretmenin aldığı derslerin listelerini DERSNO\_SINIFLİSTESİ.txt şeklinde kaydetme ekranı