Giriş

Koddaki kütüphaneler:

<stdio.h> : standart g/ç

<stdlib.h> : dinamik bellek kulanımı için eklendi (malloc)

<string.h> : string'lerle işlemler için eklendi (strcmp, strcpy)

Koddaki Macrolar:

```
#define CLASSSIZE                   35 // SINIFKAPASITE
#define STUDENTNUMBERLENGTH  10 // OGRENCINUMARAUZUNLUGU
#define NAMELENGTH                  25 // ADIUZUNLUGU
#define COURSENAMELENGTH            50 // DERSADIUZUNLUGU
#define COURSECODELENGTH            10 // DERSKODUZUNLUGU
#define COURSESREGISTERED           5  // KAYITLIDERSLER
#define MIDTERMWEIGHT        30 // VIZEAGIRLIK
#define FINALWEIGHT                 30 // FINALAGIRLIK
#define QUIZWEIGHT           20 // QUIZAGIRLIK
#define HOMEWORKWEIGHT              20 // ODEVAGIRLIK
#define MAXCOURSESIZE        20 // MAKSKURSKAPASITE
#define MAXSTUDENTSIZE       400 // MAKSOGRENCIKAPASITE
```

Koddaki Fonksiyonlar:

```
int findStudent(student_t *s, char *st);
int findCourse(courses_t *c, char *code);
double calculateStudentGPA(student_t s);
double calculateCourseAverage(courses_t   c);
```

void calculateStudentCourseGrade(courseReg_t *r);

void displayStudentInfo(student_t s);

void displayCourseInfo(courses_t c);

void printClass(courses_t c);

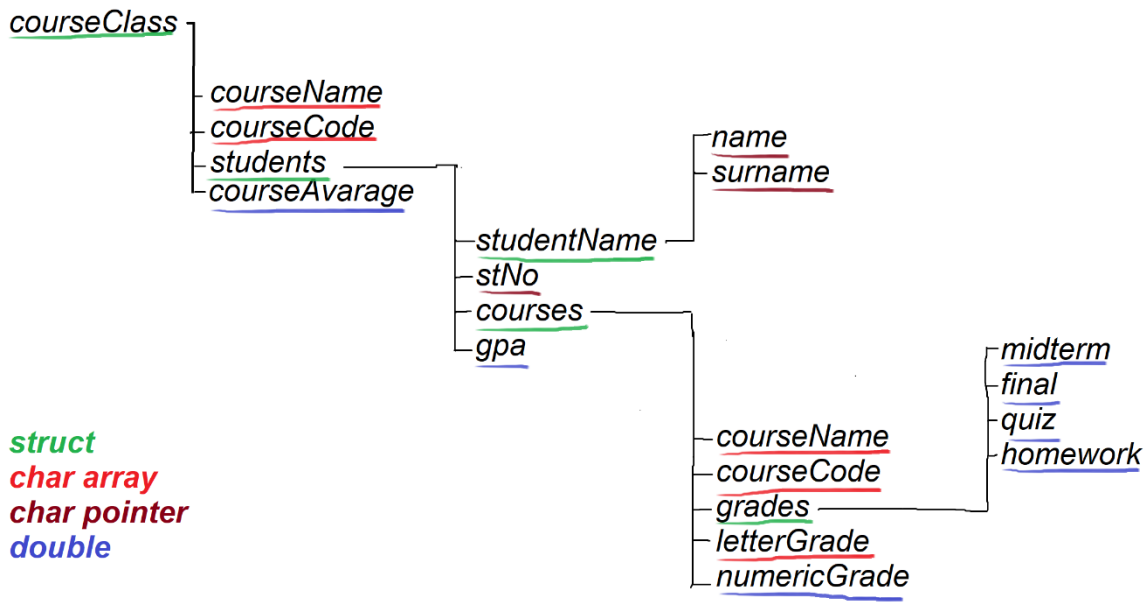
void printMenu();

courses_t addCourse();

student_t addStudent();

courseReg_t registerCourse();

Koddaki değişkenler:



Kodların İncelenmesi

main():

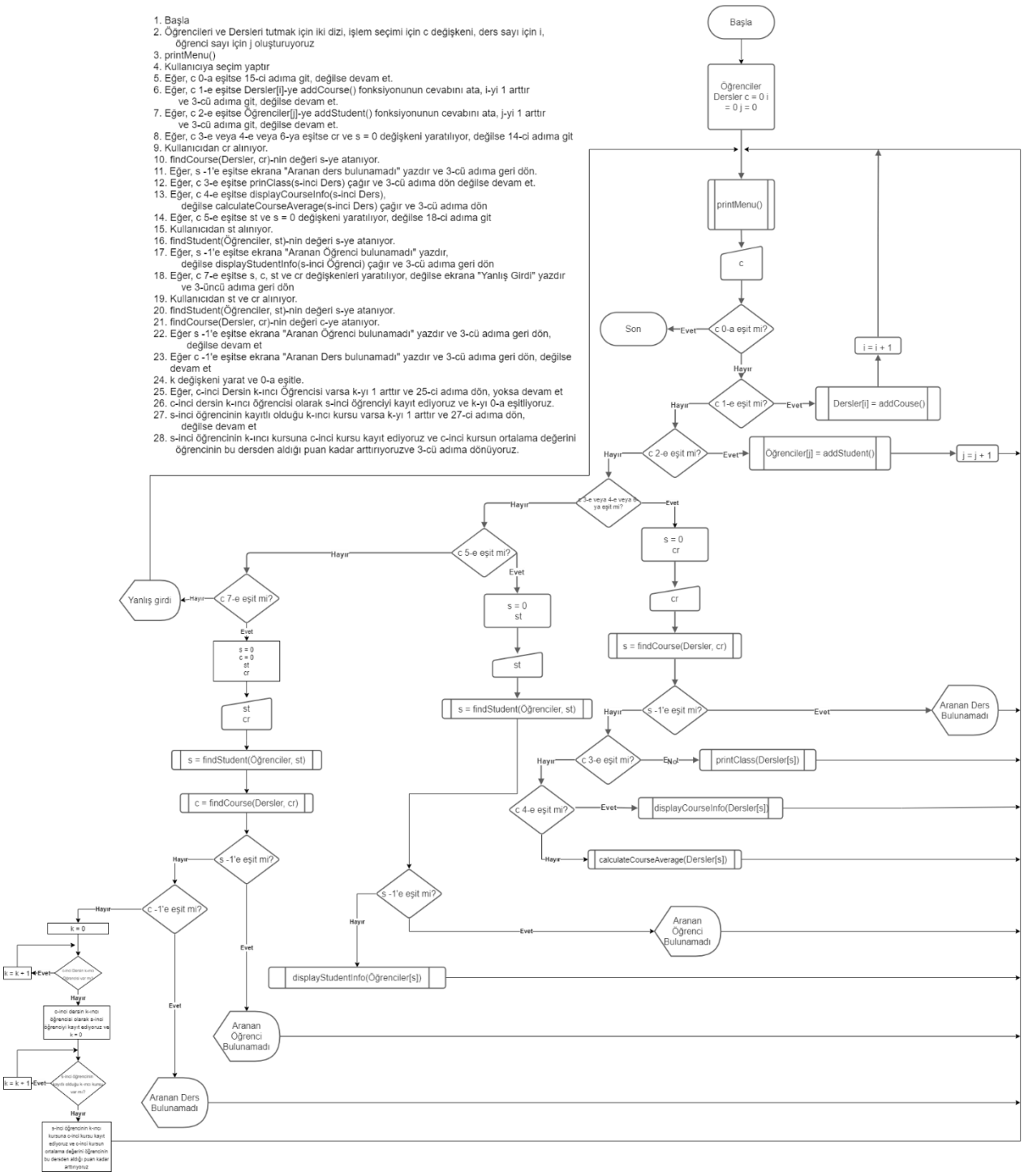
*main() fonksiyonu kodun sağlıklı çalışa bilmesi için integer değer döderen bir fonksiyon. Kodun en başında courseClass isimli MAXCOURSESIZE boyutunda courses_t yapı dizisi ve Students isimli MAXSTUDENTSIZE boyutunda student_t yapı dizisi oluşturuluyor. courseClass programa eklenen dersleri Students ise programa eklenen öğrencileri tutacak. Sonrasında integer c, i ve j değişkenlerini yaaratıp 0 a eşitliyoruz. i kaç ders eklendiğini, j ise kaç öğrenci eklendiğini tutacak. Bu bize öğrenci ekleren de yardımcı olacak. Sonrakı satırda system("cls") fonksiyonu terminali temizliyor. Ardından programımız "while" döngüsüne giriyor. "while"-a parametre olarak 1 veriyoruz ki, döngü biz durdurmadan bitmesin. Döngüye giriyoruz ilk satırda printMenu() fonksiyonu çağırılıyor. Bu fonksiyon ekrana statik menümüzü yazdırıyor. Sonrasında kullanıcıdan menüden bir değer seçmesini istiyoruz. Aldığımız girdiyi c değişkenine atıyoruz. Ardından girilen c değerine göre "if-else if-else" yapısında bir bloğa girmeye çalışıyoruz. İlk olarak c nin 0-a eşit olup olmaması kontrol ediliyor. Eğer ki, eşitse bloğa giriyoruz. Blokda tek satır kodumuz var o da break. Yani döngüden çıkmasını istiyor. Eğer ki, c 0-a değil de 1-e eşitse courseClass yapı dizisinin i-inci elemanına addCourse() fonksiyonundan dönen courses_t tipi yapıyı ekliyoruz. Ardından i-yi 1 arttırıyoruz ve bloktan çıkıyoruz. Eğer c 2-e eşitse Students yapı dizisinin j-inci elemanına addStudent() fonksiyonundan dönen student_t tipi yapıyı ekliyoruz. Ardından j-yi 1 arttırıyoruz ve bloktan çıkıyoruz. Eğer c 3, 4, 6 rakamlarından her hangi birine eşitse integer s değişkeni yaratıp 0-a eşitliyor ve char pointerı cr-yi yaratıp malloc(COURSECODELENGTH * (sizeof(*cr))) fonkiyonu ile bellekte yer açıyoruz. malloc(size) fonksiyonu size boyutunda bellekte yer açar. sizeof(*cr) ise verilen parametrenin bellekte tuttuğu yeri dönderiyor. Sonrasında ekrana printf fonksiyonu yardımı ile "Enter course Code: " yazısını bastırıyoruz ve ardından scanf ile kullanıcıdan aldığımız stringi cr pointerına atıyoruz. Ardından s değişkenine findCourse(courseClass, cr) döndürdüğü değeri atıyoruz. findCourse fonksiyonu verdiğimiz courses_t tipi yapı dizisinde verdiğimiz char* tipli kodu arayıp bulursa dizideki sırasını bulamazsa -1 dönderiyor. Buna göre sonraki satırda if bloğuna giriyoruz. Eğer ki*

*s -1'e eşitse ekrana "Course with the code you entered was not found" yazısını yazdırıyoruz. Değilse else bloğuna giriyoruz. Burada bizi bir kontrol daha bekliyor. İf de eğer ki, c 3-e eşitse printClass(courseClass[s]) fonksiyonu çağrılıyor. Bu fonksiyon verdiğimiz courses_t tipi yapının içeriğini ekrana yazdırıyor. Eğer c 4-e eşitse displayCourseInfo(courseClass[s]) fonksiyonu çağırılıyor. Bu fonksiyon verdiğimiz courses_t tipi yapının bilgilerini ekrana yazdırıyor. Eğer c 5-e eşitse ekrana "Course Avarage: " ve calculateCourseAverage(courseClass[s]) fonksiyonun dönderdiği double değerini yazdırıyor. Bu fonksiyon verdiğimiz courses_t tipi yapının öğrencilerinin ortalama notunu ekrana yazdırıyor. Bunlardan biri olmak zorunda çünki eğer bunlardan birine eşit değilse c buraya kadar gelmezdi. Diğer "else if"-de eğer c 5-e eşitse giriyor ve integer s değişkeni yaratıp 0-a eşitliyor ve char pointerı st-yi yaratıp malloc(STUDENTNUMBERLENGTH * (sizeof(*st))) fonkiyonu ile bellekte yer açıyoruz. malloc(size) fonksiyonu size boyutunda bellekte yer açar. sizeof(*st) ise verilen parametrenin bellekte tuttuğu yeri dönderiyor. Sonrasında ekrana printf fonksiyonu yardımı ile "Enter Student No: " yazısını bastırıyoruz ve ardından scanf ile kullanıcıdan aldığımız stringi st pointerına atıyoruz. Ardından s değişkenine findStudent(Students, st) döndürdüğü değeri atıyoruz. findStudent fonksiyonu verdiğimiz student_t tipi yapı dizisinde verdiğimiz char* tipli numarayı arayıp bulursa dizideki sırasını bulamazsa -1 dönderiyor. Buna göre sonraki satırda if bloğuna giriyoruz. Eğer ki s -1'e eşitse ekrana "Student with the number you entered was not found" yazısını yazdırıyoruz. Değilse else bloğuna giriyoruz. Else bloğunda da displayStudentInfo(students[s]) fonksiyonu çağırılıyor. Bu fonksiyon verdiğimiz student_t tipi yapının bilgilerini ekrana yazdırıyor. Eğer ki, c 7-e eşitse integer s ve c değişkeni yaratıp 0-a eşitliyor ve char pointerı st-yi ve cr-yi yaratıp malloc(STUDENTNUMBERLENGTH * (sizeof(*st))) fonkiyonu ile st için, malloc(COURSECODELENGTH * (sizeof(*cr))) fonkiyonu ile cr için bellekte yer açıyoruz. malloc(size) fonksiyonu size boyutunda bellekte yer açar. sizeof(*st) ise verilen parametrenin bellekte tuttuğu yeri dönderiyor. Sonrasında ekrana printf fonksiyonu yardımı ile "Enter Student No: " yazısını bastırıyoruz ve ardından scanf ile kullanıcıdan aldığımız stringi st pointerına atıyoruz. Sonrasında ekrana printf fonksiyonu yardımı ile "Enter course Code: " yazısını bastırıyoruz ve ardından scanf ile kullanıcıdan aldığımız stringi cr pointerına atıyoruz. Ardından s değişkenine findStudent(Students, st) , c değişkenine ise findCourse(courseClass, cr) döndürdüğü değeri atıyoruz. findStudent fonksiyonu verdiğimiz student_t tipi*

yapı dizisinde verdiğimiz char tipli numarayı arayıp bulursa dizideki sırasını bulamazsa -1 dönderiyor. findCourse fonksiyonu verdiğimiz courses_t tipi yapı dizisinde verdiğimiz char* tipli kodu arayıp bulursa dizideki sırasını bulamazsa -1 dönderiyor. Buna göre sonraki satırda if bloğuna giriyoruz. Eğer ki s -1'e eşitse ekrana "Student with the number you entered was not found" yazısını yazdırıyoruz. Eğer ki c -1'e eşitse ekrana "Course with the code you entered was not found" yazısını yazdırıyoruz. Değilse else bloğuna giriyoruz. Else bloğunda da integer k değişkeni tanımlayıp 0-a eşitliyoruz. Ardından while döngüsü ile c-inci dersin k-ıncı öğrencisinin numarası NULL olana kadar k-yı 1 arttırıyoruz. Bu sayede c-inci kursun kaç tane öğrencisi olduğunu bulup sonrakine s-inci öğrenciyi kayıt ediyoruz. Ardından k-yı 0-a eşitliyoruz. Ardından while döngüsü ile s-inci öğrencinin k-ıncı dersininin isminin ilk harfi "*" ise k-yı 1 arttırıyoruz. Bu sayede s-inci öğrencinin kaç tane dersi olduğunu buluyoruz ve sıradakı dersine registerCourse(Students[s], courseClass[c]) fonksiyonunun dönderdiği cevabı ekliyoruz. Ve son olarak eklediğimiz öğrencinin notunu dersin ortalamasına ekliyoruz. Eğer c yukarıdakilerin hiç birine ait değilse o zaman ekrana "Wrong Input" yazdırıyoruz ve döngünün başına gidiyoruz. Eğer program döngüden çıkarsa 0 dönderip programı bitiriyoruz.*

```c
int main()
{
    courses_t courseClass[MAXCOURSESIZE];                   // Dersleri tutmak için kullanıyoruz
    student_t Students[MAXSTUDENTSIZE];                     // ogrencileri tutmak için kullanıyoruz
    int c = 0, i = 0, j = 0;                                // c - islem secimini tutuyor; i - Ders sayisi; j - ogrenci sayisi
    system("cls");                                          // Konsolu temizlemek için

    while (1) {                                             // c 0 olmadıkca devam edecek
        printMenu();
        scanf("%d", &c);                                    // secimi yapıyoruz

        if (c == 0)                                         // eger 0 ise programı bitiriyoruz
        {
            break;
        }
        else if (c == 1)                                    // Ders ekleme
        {
            courseClass[i] = addCourse();
            i++;
        }
        else if (c == 2)                                    // ogrenci ekleme
        {
            Students[j] = addStudent();
            j++;
        }
        else if (c == 3 || c == 4 || c == 6)                // hepsi baslangicda girilen koda gore dersi bulup ardından islem yaptigından birlestirdim.
        {
            int x = 0;                                      // aranan index x-de tutulacak
            char *cr;                                       // aranan kod

            cr = malloc(COURSECODELENGTH * (sizeof(*cr)));  // bellekte yer ayiriyoruz

            printf("\nEnter Course Code: ");
            scanf(" %s", cr);                               // kullanicidan aranacak kelimeyi aliyoruz

            x = findCourse(courseClass, cr);                // ve dersi ariyoruz...

            if(x == -1) printf("\n\nCourse with the code you entered was not found\n\n");   // x = -1 ise bu bulunamadi demek
            else                                                                            // aksi halde girilen secime gore fonksiyon cagiriyoruz
            {
                if (c == 3) printClass(courseClass[x]);
                else if(c == 4) displayCourseInfo(courseClass[x]);
                else if(c == 6) printf("Course Average: %.2f\n\n", calculateCourseAverage(courseClass[x]));
            }

        }
        else if (c == 5)
        {
            int x = 0;                                                  // aranan index x-de tutulacak
            char *st;                                                   // aranan numara

            st = malloc(STUDENTNUMBERLENGTH * (sizeof(*st)));           // bellekte yer ayiriyoruz

            printf("\nEnter Student No: ");
            scanf(" %s", st);                                           // kullanicidan aranacak kelimeyi aliyoruz

            x = findStudent(Students, st);                              // ve ogrenciyi ariyoruz...

            if(x == -1) printf("\n\nStudent with the number you entered was not found\n\n");     // x = -1 ise bu bulunamadi demek
            else displayStudentInfo(Students[x]);                       // aksi halde ogrenci bilgilerini aliyoruz.

        }
        else if (c == 7)
        {
            int x = 0, c = 0;                                           // ogrencinin indexi x-de dersin indexi c-de tutulacak
            char *st, *cr;                                              // aranan numarayi ve kodu tutacaklar

            cr = malloc(COURSECODELENGTH * (sizeof(*cr)));              // bellekte yer ayiriyoruz
            st = malloc(STUDENTNUMBERLENGTH * (sizeof(*st)));

            printf("\nEnter Student No: ");
            scanf(" %s", st);                                           // kullanicidan aranacak numarayi aliyoruz

            printf("\nEnter Course Code: ");
            scanf(" %s", cr);                                           // kullanicidan aranacak kodu aliyoruz

            c = findCourse(courseClass, cr);                            // ve dersi ariyoruz...
            x = findStudent(Students, st);                              // ve ogrenciyi ariyoruz...

            if(x == -1)                                                 // Ogrenci bulunamamissa true olacak
            {
                printf("\n\nStudent with the number you entered was not found\n\n");
            }
            else if (c == -1)                                           // Ders bulunamamissa true olacak
            {
                printf("\n\nCourse with the code you entered was not found\n\n");
            }
            else
            {
                int k = 0;

                while(courseClass[c].students[k].stNo != NULL) k++;     // derse yeni kaydi yapabilmek için ogrenci numarasi NULL olana kadar k-yi arttiriyoruz.

                courseClass[c].students[k] = Students[x];               // ve kaydi yapiyoruz.

                k = 0;                                                  // baska degisken kullanmamak için k-yi 0 yapip yeniden kullaniyoruz

                while(Students[x].courses[k].courseName[0] != 42) k++;  // Ogrencinin yeni kaydini eklemek için ekli olan derslerin bi sonrasina bu dersi de ekliyoruz

                Students[x].courses[k] = registerCourse(Students[x], courseClass[c]);
                courseClass[c].courseAverage += Students[x].courses[k].numericGrade;        // ve ogrenci kayit ettikce notunu da courseAverage degiskenine ekliyoruz.
            }
        }
        else
        {
            printf("\nWrong Input !!!\n\n");                             // belirtilmeyen bir giris girdiginde hata verip yeniden soruyoruz.
        }
    }
    return 0;
}
```

1. Başla
2. Öğrencileri ve Dersleri tutmak için iki dizi, işlem seçimi için c değişkeni, ders sayı için i, öğrenci sayı için j oluşturuyoruz
3. printMenu()
4. Kullanıcıya seçim yaptır
5. Eğer, c 0-a eşitse 15-ci adıma git, değilse devam et.
6. Eğer, c 1-e eşitse Dersler[i]-ye addCourse() fonksiyonunun cevabını ata, i-yi 1 arttır ve 3-cü adıma git, değilse devam et.
7. Eğer, c 2-e eşitse Öğrenciler[j]-ye addStudent() fonksiyonunun cevabını ata, j-yi 1 arttır ve 3-cü adıma git, değilse devam et.
8. Eğer, c 3-e veya 4-e veya 6-ya eşitse cr ve s = 0 değişkeni yaratılıyor, değilse 14-ci adıma git
9. Kullanıcıdan cr alınıyor.
10. findCourse(Dersler, cr)-nin değeri s-ye atanıyor.
11. Eğer, s -1'e eşitse ekrana "Aranan ders bulunamadı" yazdır ve 3-cü adıma geri dön.
12. Eğer, c 3-e eşitse prinClass(s-inci Ders) çağır ve 3-cü adıma dön değilse devam et.
13. Eğer, c 4-e eşitse displayCourseInfo(s-inci Ders), değilse calculateCourseAverage(s-inci Ders) çağır ve 3-cü adıma dön
14. Eğer, c 5-e eşitse st ve s = 0 değişkeni yaratılıyor, değilse 18-ci adıma git
15. Kullanıcıdan st alınıyor.
16. findStudent(Öğrenciler, st)-nin değeri s-ye atanıyor.
17. Eğer, s -1'e eşitse ekrana "Aranan Öğrenci bulunamadı" yazdır, değilse displayStudentInfo(s-inci Öğrenci) çağır ve 3-cü adıma geri dön
18. Eğer, c 7-e eşitse s, c, st ve cr değişkenleri yaratılıyor, değilse ekrana "Yanlış Girdi" yazdır ve 3-üncü adıma geri dön
19. Kullanıcıdan st ve cr alınıyor.
20. findStudent(Öğrenciler, st)-nin değeri s-ye atanıyor.
21. findCourse(Dersler, cr)-nin değeri c-ye atanıyor.
22. Eğer s -1'e eşitse ekrana "Aranan Öğrenci bulunamadı" yazdır ve 3-cü adıma geri dön, değilse devam et
23. Eğer c -1'e eşitse ekrana "Aranan Ders bulunamadı" yazdır ve 3-cü adıma geri dön, değilse devam et
24. k değişkeni yarat ve 0-a eşitle.
25. Eğer, c-inci Dersin k-ıncı Öğrencisi varsa k-yı 1 arttır ve 25-ci adıma dön, yoksa devam et
26. c-inci dersin k-ıncı öğrencisi olarak s-inci öğrenciyi kayıt ediyoruz ve k-yı 0-a eşitliyoruz.
27. s-inci öğrencinin kayıtlı olduğu k-ıncı kursu varsa k-yı 1 arttır ve 27-ci adıma dön, değilse devam et
28. s-inci öğrencinin k-ıncı kursuna c-inci kursu kayıt ediyoruz ve c-inci kursun ortalama değerini öğrencinin bu dersden aldığı puan kadar arttırıyoruzve 3-cü adıma dönüyoruz.



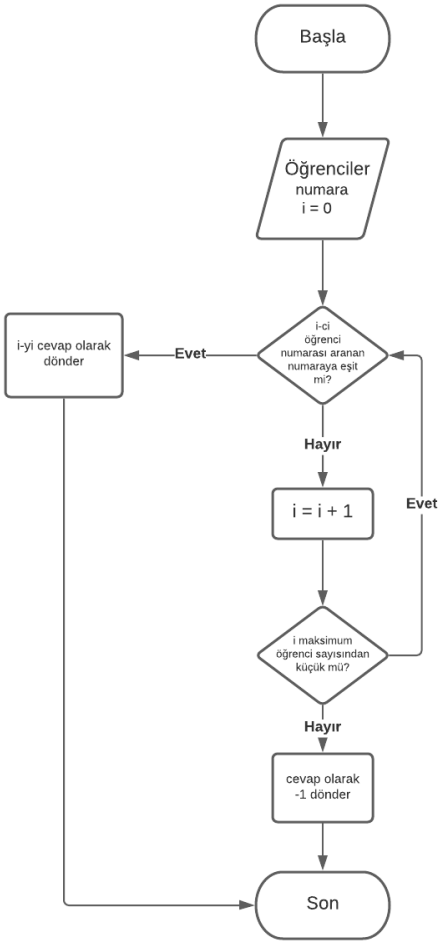
Şema ve Sözde Kodun Açıklaması:

Program başlıyor, Öğrencileri ve Dersleri tutmak için iki dizi, işlem seçimi için c değişkeni, ders sayı için i, öğrenci sayı için j oluşturuyoruz. printMenu() fonksiyonu ile menüyü bastırıyoruz ve döngü başlatıyoruz. Kullanıcıdan c-yi alıyoruz eğer, c 0-a eşit ise programı sonlandırıyoruz, değilse devam ediyoruz. Eğer, c 1-e eşitse Dersler[i]-ye addCourse() fonksiyonunun cevabını atıyıp, i-yi 1 arttırıyoruz ve döngünün başına gidiyoruz, değilse devam et. Eğer, c 2-e eşitse Öğrenciler[j]-ye addStudent() fonksiyonunun cevabını atıyoruz, j-yi 1 arttırıp döngünün başına gidiyoruz, değilse devam et. eğer, c 3-e veya 4-e veya 6-ya eşitse cr ve s = 0 değişkeni yaratılıyor, kullanıcıdan cr alınıyor, findCourse(Dersler, cr)-nin değeri s-ye atanıyor ve eğer, s -1'e eşitse ekrana "Aranan ders bulunamadı" yazdırıyor ve döngünün başına gidiyoruz eğer, c 3-e eşitse prinClass(s-inci Ders) çağırıyoruz ve döngünün başına gidiyoruz, eğer, c 4-e eşitse displayCourseInfo(s-inci Ders), değilse calculateCourseAverage(s-inci Ders) çağırıyor ve döngünün başına gidiyoruz. Değilse eğer, c 5-e eşitse st ve s = 0 değişkeni yaratılıyor, kullanıcıdan st alınıyor, findStudent(Öğrenciler, st)-nin değeri s-ye atanıyor eğer, s -1'e eşitse ekrana "Aranan Öğrenci bulunamadı" yazdır değilse displayStudentInfo(s-inci Öğrenci) çağırıyor ve döngünün başına gidiyoruz, değilse eğer, c 7-e eşitse s, c, st ve cr değişkenleri yaratılıyor, değilse ekrana "Yanlış Girdi" yazdırıp ve döngünün başına gidiyoruz kullanıcıdan st ve cr alınıyor ve findStudent(Öğrenciler, st)-nin değeri s-ye atanıyor findCourse(Dersler, cr)-nin değeri c-ye atanıyor. Eğer s -1'e eşitse ekrana "Aranan Öğrenci bulunamadı" yazdırıyor ve döngünün başına gidiyoruz, Eğer c -1'e eşitse ekrana "Aranan Ders bulunamadı" yazdırıyor ve döngünün başına gidiyoruz,değilse k değişkeni yarat ve 0-a eşitle. Eğer, c-inci Dersin k-ıncı Öğrencisi varsa k-yı 1 arttır ve yeniden kontrol et, yoksa. c-inci dersin k-ıncı öğrencisi olarak s-inci öğrenciyi kayıt ediyoruz ve k-yı 0-a eşitliyoruz. s-inci öğrencinin kayıtlı olduğu k-ıncı kursu varsa k-yı 1 arttır ve yeniden kontrol et, değilse s-inci öğrencinin k-ıncı kursuna c-inci kursu kayıt ediyoruz ve c-inci kursun ortalama değerini öğrencinin bu dersten aldığı puan kadar arttırıyoruz ve döngünün başına gidiyoruz.

findStudent():

findStudent() fonksiyonu paremetre olarak, öğrencilerin bilgilerini içeren s işaretçisini ve aranan öğrencinin numarasını içeren st işaretcisini alıyor. Geriye ise , öğrencilerin bilgilerini içeren s işaretçisinin kaçıncı elemanı ile aradığımız numara eşleşmişse, o elemanın işaretçideki yerini dönderiyor. "for" döngümüzün parametreleri içerisinde integer i değişkenini tanımlayıp 0-a eşitliyoruz. Her döngüde i değişkenini 1 arttırmak sureti ile i MAXSTUDENTSIZE (400 , toplam öğrenci kayıt kapasitesi) 'dan küçük olduğu müddetce dögümüz dönmeye devam edecektir. Döngünün içerisinde bir "if" bloğu yer almaktadır. "if" kontrolcümüze parametre olaraktan "strcmp(s[i].stNo, st) == 0" verilmiş. strcmp(string1, string2) – string1 ile string2-yi karşılaştırıyor. String1 büyükse integer 1, string2 büyükse integer -1, eşitlerse integer 0 değeri dönderiyor. Biz de strcmp(s[i].stNo, st) içerisinde öğrencilerin bilgilerini içeren s işaretcisinin i-ci elemanının numarası ile aradığımız numarayı karşılaştırıyoruz. Eğer 0 dönerse bu eşit oldukları anlamına gelir ve "if" bloğuna gireriz. 0 dışında her hangi bir şey bizim şartımızı ödemez. Eğer eşitlik saplanmış ve "if" bloğu içerisine girilmişse demekki aradığımız elemanı bulmuşuz. Bu elemanın öğrencilerin bilgilerini içeren s işaretcisindeki yerini fonksiyonun cevabı olarak dönderiyoruz ve fonksiyon böylelikle bitiyor. Eğer ki, "for" döngüsü bitene kadar "if" bloğuna girilmemişse "for" döngüsü bittikten sonra integer -1 dönderiyoruz. Bu da değerin bulunamadığı anlamına geliyor ve fonksiyon bitiyor.

```
int findStudent(student_t *s, char *st)
{
    for(int i = 0 ; i < MAXSTUDENTSIZE; i++)
    {
        if(strcmp(s[i].stNo, st) == 0)
        {
            return i;
        }
    }

    return -1;
}
```



1. Başla
2. Öğrenci bilgilerini içeren yapı dizisi alınır (student_t *s)
3. Aranan öğrenci numarası alınır (char *st)
4. Integer i değişkeni tanımlayıp 0-a eşitliyoruz.
5. 2-ci adımda alınan dizideki i-ci öğrencinin numarası ile aradığımız numarayı karşılaştırıyoruz
6. Eğer, eşitse i-yi cevap olarak döndürüyoruz ve 9-cu adıma gidiyoruz, değilse devam ediyoruz.
7. i-yi 1 arttırıyoruz.
8. Eğer, i maksimum öğrenci sayısından küçükse 5-ci adıma gidiyoruz, değilse cevap olarak -1 döndürüyoruz ve devam ediyoruz.
9. Son

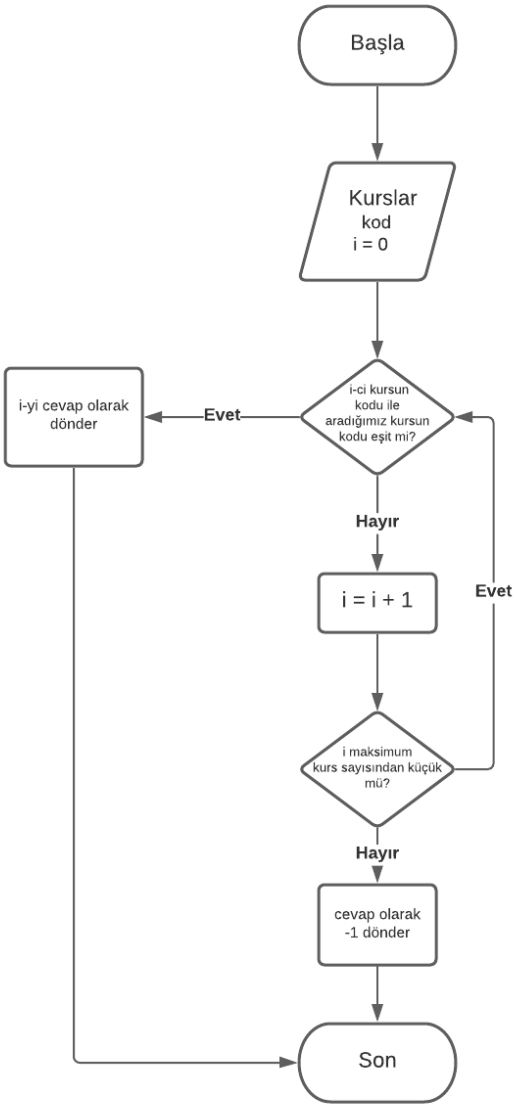
Şema ve Sözde Kodun Açıklaması:

Koda başlarken başla komutu verildikten sonra öğrenci belgesini içeren yapı dizisi alınacak bir yapı oluşturuldu. Aranan öğrenci numaralarımız alıp i-ci öğrencinin numarası ile aradığımız numara eşitse i-yi cevap olarak dönderiyoruz, değilse i-yi 1 arttırıyoruz eğer i maksimum öğrenci sayısından küçükse eşitliği yeniden kontrol ediyoruz, büyükse cevap olarak -1 dönderiyoruz. Sonrasın da programı bitiriyoruz.

findCourse():

findStudent() fonksiyonu paremetre olarak, derslerin bilgilerini içeren c işaretçisini ve aranan dersin kodunu içeren code işaretcisini alıyor. Geriye ise , derslerin bilgilerini içeren c işaretçisinin kaçıncı elemanı ile aradığımız kod eşleşmişse, o elemanın işaretçideki yerini dönderiyor. "for" döngümüzün parametreleri içerisinde integer i değişkenini tanımlayıp 0-a eşitliyoruz. Her döngüde i değişkenini 1 arttırmak sureti ile i MAXCOURSESIZE (20 , toplam ders kayıt kapasitesi) 'dan küçük olduğu müddetce döngümüz dönmeye devam edecektir. Döngünün içerisinde bir "if" bloğu yer almaktadır. "if" kontrolcümüze parametre olaraktan "strcmp(c[i].courseCode, code) == 0" verilmiş. strcmp(string1, string2) – string1 ile string2-yi karşılaştırıyor. String1 büyükse integer 1, string2 büyükse integer -1, eşitlerse integer 0 değeri dönderiyor. Biz de strcmp(c[i].courseCode, code) içerisinde derslerin bilgilerini içeren c işaretcisinin i-ci elemanının kodu ile aradığımız kodu karşılaştırıyoruz. Eğer 0 dönerse bu eşit oldukları anlamına gelir ve "if" bloğuna gireriz. 0 dışında her hangi bir şey bizim şartımızı ödemez. Eğer eşitlik saplanmış ve "if" bloğu içerisine girilmişse demekki aradığımız elemanı bulmuşuz. Bu elemanın öğrencilerin bilgilerini içeren c işaretcisindeki yerini fonksiyonun cevabı olarak dönderiyoruz ve fonksiyon böylelikle bitiyor. Eğer ki, "for" döngüsü bitene kadar "if" bloğuna girilmemişse "fo"r döngüsü bittikten sonra integer -1 dönderiyoruz. Bu da değerin bulunamadığı anlamına geliyor ve fonksiyon bitiyor.

```
int findCourse(courses_t *c, char *code)
{
    for(int i = 0 ; i < MAXCOURSESIZE; i++)
    {
        if(strcmp(c[i].courseCode, code) == 0)
        {
            return i;
        }
    }

    return -1;
}
```



1. Başla
2. Kurs bilgilerini içeren yapı dizisi alınır (courses_t *c)
3. Aranan kursun kodu alınır (char *code)
4. Integer i değişkeni tanımlayıp 0-a eşitliyoruz.
5. 2-ci adımda alınan dizideki i-ci kursun kodu ile aradığımız kodu karşılaştırıyoruz.
6. Eğer, eşitse i-yi cevap olarak döndürüyoruz ve 9-cu adıma gidiyoruz, değilse devam ediyoruz.
7. i-yi 1 arttırıyoruz.
8. Eğer, i maksimum kurs sayısından küçükse 5-ci adıma gidiyoruz, değilse cevap olarak -1 döndürüyoruz ve devam ediyoruz.
9. Son.

Şema ve Sözde Kodun Açıklaması:

Koda başlarken başla komutu verildikten sonra program başlıyor,daha sonra kurs bilgilerini içeren yapı dizisini alıp aranan kursumuzun kodunuz istiyoruz.Değişkenimizi tanımlayıp sıfıra eşitliyoruz.2.adımımızda aldığımız dizideki kursun kodu ile aradığımız kodu karşılaştırıyoruz.Daha sonra eşitliğimiz var ise y mizi cevap olarak döndürüyoruz ve 9 .adıma geçiyoruz değil ise devam ediyoruz .Eğer i değişkenimiz max.kurs sayısından küçük ise 5. Adıma dönüyoruzdeğil ise cevabımız -1 döndürüp sonlandırıyoruz.Özet geçmek gerekirse dersler dizisi, aranan code ve i değişkenimizi 0 olarak alınıyor. i-ci

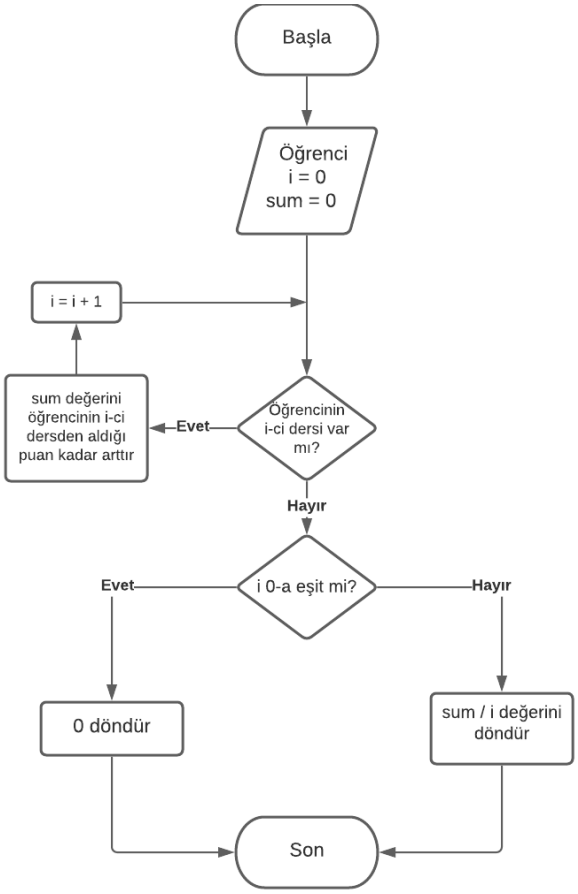
dersin kodu ile aradığımız kod eşitse i-yi cevap olarak döndürüyoruz, değilse i-yi 1 arttırıyoruz eğer i maksimum ders sayısından küçükse eşitliği yeniden kontrol ediyoruz, büyükse cevap olarak -1 döndürüyoruz. Sonrasın da programı bitiriyoruz.

calculateStudentGPA():

İlk başta öğrencinin bütün derslerinin notlarının ortalamasının oluşturulması için fonksiyon oluşturuyoruz."int i "ile notların toplamını tutacak değişkeni atıyoruz.Daha sonra

s.courses[i].courseName[0] != 42 fonksiyonuna parametre olarak gelen öğrencinin (s) kayıtlı olduğu i-inci kursun (s.courses[i]) isminin birinci harfi (s.courses[i].courseName[0]) 42-ye yani ""-e eşitse bu dizinin o elemanının boş olduğunu gösteriyor. Biz notların toplamını bulmak için boş elemana kadar bütün elemanları topluyoruz.Daha sonra i değişkenimiz sıfır değişkenimiz sıfıra eşit ise döndürüyoruz aksi halde sıfıra bölünme sorunu çıkartacaktı. En sonda ortalamayı bulmak için toplamı dizinin dolu eleman sayısına bölüyoruz. Bu bölmede dizide eğer dolu eleman yoksa sıfıra a bölme durumu olacaktı. Bu kontrol yapısı ile bunun karşısına geçmiş oluyoruz.*

```
double calculateStudentGPA(student_t s)
{
    int i = 0;
    double sum = 0;

    for(i = 0; s.courses[i].courseName[0] != 42; i++)
        sum += s.courses[i].numericGrade;

    if(i == 0) return 0;
    else return sum / i;
}
```



1. Başla
2. Öğrenci bilgilerini içeren yapı alınır (student_t s)
3. Integer i ve double sum değişkenlerini tanımlayıp 0-a eşitliyoruz.
4. Eğer, öğrencinin i-ci dersi yoksa 7-ci adıma gidiyoruz,
   varsa devam ediyoruz.
5. sum değerini öğrencinin i-ci dersden aldığı puan kadar arttırıyoruz.
6. i-yi 1 arttırıyoruz ve 4-cü adıma gidiyoruz.
7. Eğer, i 0-a eşitse 0 döndürüyoruz,
   değilse sum / i değerini döndürüyoruz.
8. Son.

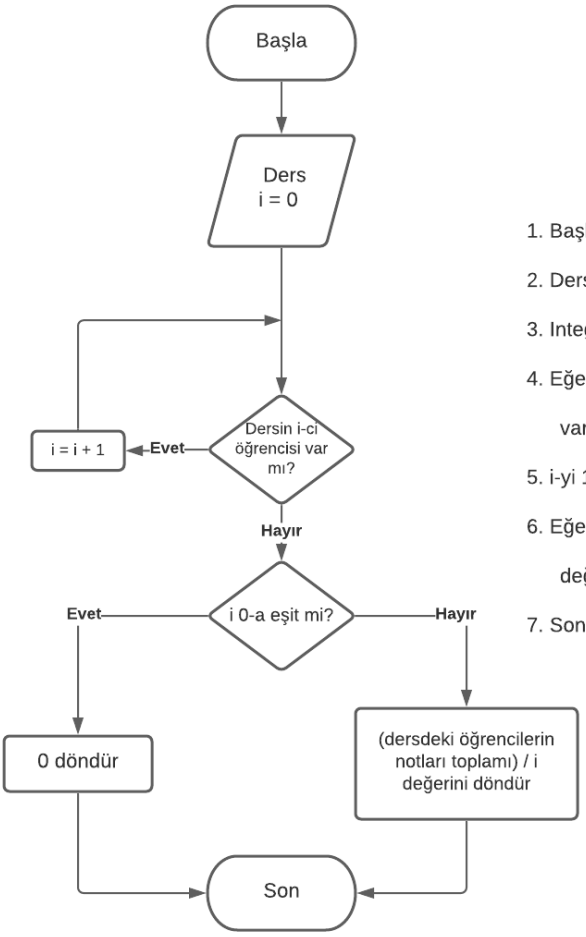
Şema ve Sözde Kodun Açıklaması:

Koda başlarken başla komutu verdikten sonra program başlıyor ve öğrenci bilgilerini içeren yapıyı alıyoruz değişkenlerimizi tanımlayıp sıfıra eşitliyoruz Eğer öğrencimizin i dersi yoksa 7.adımımıza geçiyoruz varsa devam ediyoruz.”sum”değerimi i nin aldığı değer puan kadar arttırıyoruz. 6.adımda i yi bir arttırıyoruz 4.adıma dönüyoruz Eğer değişkenimiz sıfıra eşitse sıfır döndürüyoruz değilse sum değişkenimiziin değerini döndürüyorum.Özetlemek gerekirse Program başlıyor, Öğrenciler dizisi, s ve i 0 olarak alınıyor. Öğrencinin i-ci dersi varsa o dersden aldığı puanı sum-a ekleyip i-yi 1 arttırıyoruz ve yeniden kontrol ediyoruz. Yoksa i-nin 0-a eşit olup olmadığına bakıyoruz. Eşitse 0 döndürüyoruz, değilse sum-ı i-ye bölüyoruz ve bu değeri döndürüyoruz. Sonrasın da programı bitiriyoruz.

calculateCourseAverage():

Burada c.students[i].stNo != NULL - calculateStudentGPA()-dakı kontrolle aynı mantıkta çalışıyor. Tek farkı "stNo"-un array değil pointer olması bu yüzden eğer değer atanmamışsa NULL olacaktır.İlk başta verilen dersin bütün öğrencilerin notlarının ortlamasını alıyoruz.Derse kayıtlı öğrencilerin sayısını bulmak için öğrenci numarasını NULL olana kadar değişkenimiz olan i yi arttırıyoruz.Eğer değişkenimiz i sıfıra eşit ise sıfır döndürüyoruz yoksa sıfıra bölme sorunu çıkartacaktı.Main fonksiyonumuzda öğrenci kayıt oldukça notunu da "courseAverage" değişkenine ekliyoruz.Yani "courseAverage" toplamı tutuyor.

```
double calculateCourseAverage(courses_t c)
{
    int i = 0;

    while(c.students[i].stNo != NULL) i++;

    if(i == 0) return 0;
    else return c.courseAverage / i;
}
```



1. Başla
2. Ders bilgilerini içeren yapı alınır (courses_t c)
3. Integer i değişkeni tanımlayıp 0-a eşitliyoruz.
4. Eğer, derse kayıtlı i-ci öyrenci yoksa 6-cı adıma gidiyoruz,
   varsa idevam ediyoruz.
5. i-yi 1 arttırıyoruz ve 4-cü adıma gidiyoruz.
6. Eğer, i 0-a eşitse 0 döndürüyoruz,
   değilse (dersdeki öğrencilerin notları toplamı) / i değerini döndürüyoruz.
7. Son.

Şema ve Sözde Kodun Açıklaması:

Program başlıyor, Dersler dizisi ve i 0 olarak alınıyor. Dersin i-ci Öğrencisi varsa i-yi 1 arttırıyoruz ve yeniden kontrol ediyoruz. Yoksa i-nin 0-a eşit olup olmadığına bakıyoruz. Eşitse 0 döndürüyoruz, değilse dersteki öğrencilerin notları toplamını i-ye bölüyoruz ve bu değeri döndürüyoruz. Sonrasın da programı bitiriyoruz.

calculateStudentCourseGrade():

Verilen kayita gore ogrencinin kayitli oldugu dersin notunun hesaplanmasi basit matematik ile son notu buluyoruz ve bu nota gore puan veriyoruz.

$$grade = \frac{\sum_{vize}^{quiz} sınav\ puanı * sınav\ ağırlığı)/100\ )}{100}$$

100 ≥ grade > 89 : AA, 4.00

90 > grade > 84 : BA, 3.50

85 > grade > 79 : BB, 3.00

80 > grade > 74 : CB, 2.50

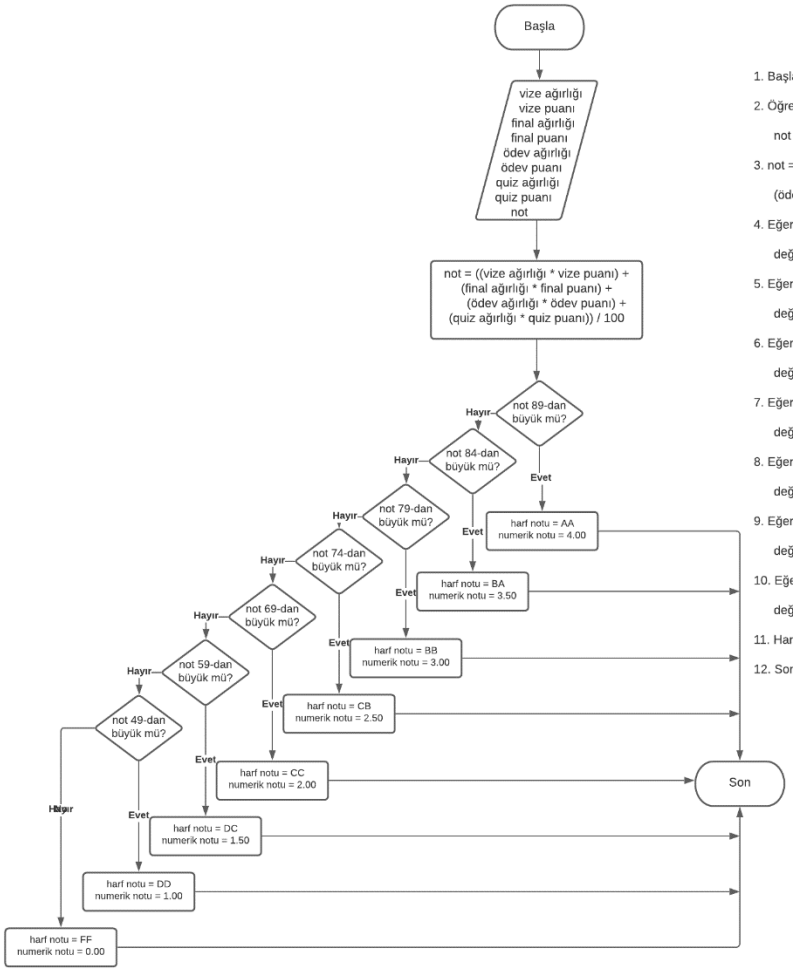
75 > grade > 69 : CC, 2.00

70 > grade > 59 : DC, 1.50

60 > grade > 49 : DD, 1.00

50 > grade ≥ 0 : FF, 0.00

```c
void calculateStudentCourseGrade(courseReg_t *r)
{
    double grade = (MIDTERMWEIGHT * r->grades.midterm + FINALWEIGHT * r->grades.final
                    + HOMEWORKWEIGHT * r->grades.homework + QUIZWEIGHT * r->grades.quiz) / 100.00;

    if(grade > 89)
    {
        strcpy(r->letterGrade, "AA");
        r->numericGrade = 4.00;
    } else if(grade > 84)
    {
        strcpy(r->letterGrade, "BA");
        r->numericGrade = 3.50;
    } else if(grade > 79)
    {
        strcpy(r->letterGrade, "BB");
        r->numericGrade = 3.00;
    } else if(grade > 74)
    {
        strcpy(r->letterGrade, "CB");
        r->numericGrade = 2.50;
    } else if(grade > 69)
    {
        strcpy(r->letterGrade, "CC");
        r->numericGrade = 2.00;
    } else if(grade > 59)
    {
        strcpy(r->letterGrade, "DC");
        r->numericGrade = 1.50;
    } else if(grade > 49)
    {
        strcpy(r->letterGrade, "DD");
        r->numericGrade = 1.00;
    } else
    {
        strcpy(r->letterGrade, "FF");
        r->numericGrade = 0.00;
    }
}
```



1. Başla
2. Öğrencinin vize, final, ödev ve quiz puanları ve ağırlıkları alınıyor ve not değişkeni yaratılıyor.
3. not = ((vize ağırlığı * vize puanı) + (final ağırlığı * final puanı) + (ödev ağırlığı * ödev puanı) + (quiz ağırlığı * quiz puanı)) / 100
4. Eğer, not 89-dan büyükse harf notunu AA, numerik notunu 4.00 yapıp 12.cu adıma gidiyoruz, değilse devam ediyoruz.
5. Eğer, not 84-dan büyükse harf notunu BA, numerik notunu 3.50 yapıp 12.cu adıma gidiyoruz, değilse devam ediyoruz.
6. Eğer, not 79-dan büyükse harf notunu BB, numerik notunu 3.00 yapıp 12.cu adıma gidiyoruz, değilse devam ediyoruz.
7. Eğer, not 74-dan büyükse harf notunu CB, numerik notunu 2.50 yapıp 12.cu adıma gidiyoruz, değilse devam ediyoruz.
8. Eğer, not 69-dan büyükse harf notunu CC, numerik notunu 2.00 yapıp 12.cu adıma gidiyoruz, değilse devam ediyoruz.
9. Eğer, not 59-dan büyükse harf notunu DC, numerik notunu 1.50 yapıp 12.cu adıma gidiyoruz, değilse devam ediyoruz.
10. Eğer, not 49-dan büyükse harf notunu DD, numerik notunu 1.00 yapıp 12.cu adıma gidiyoruz, değilse devam ediyoruz.
11. Harf notunu FF, numerik notunu 0.00 yapıp devam ediyoruz.
12. Son.

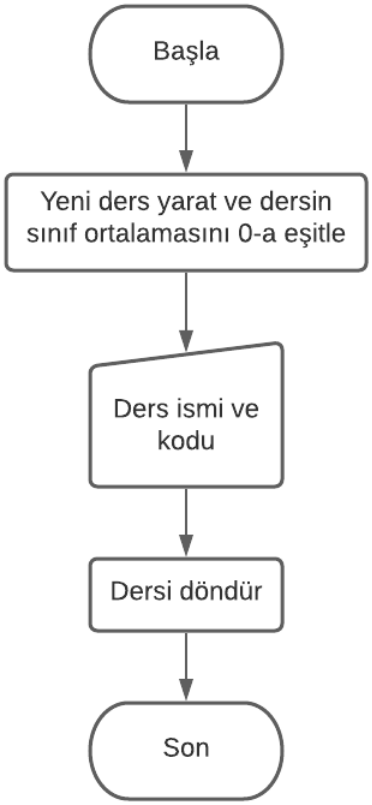
Şema ve Sözde Kodun Açıklaması:

Program başlıyor, vize, final, ödev, quiz sınav notları ve ağırlıkları alınıyor.
$grade = \frac{\sum_{vize}^{quiz} sınav\ puanı * sınav\ ağırlığı)/100\ )}{100}$ *formülü ile nihai not hesaplanıyor.*
Bu nota göre harf notu ve numerik notu kaydedip programı bitiriyoruz.

addCourse():

Bu fonksiyonda dersimizi ekliyoruz . c.courseAvarage = 0; - başlangıç olarak 0 a eşitliyoruz."courses_t c ile yeni ders ekliyoruz .Daha sonra scanf ile kullanıcıdan ders ismi alıyorum.Daha sonra scanf ile yine kullanıcıdan dersin kodunu alıyorum."c.courseAverage" başlangıç olarak sıfıra eşitliyoruz.Daha sonra printf ile ekran a başarı mesajı bastırıyoruz.

```
courses_t addCourse()
{
    courses_t c;

    printf("Course Name: ");
    scanf(" %s", c.courseName);

    printf("Course Code: ");
    scanf(" %s", c.courseCode);

    c.courseAverage = 0;

    printf("\n\n___Course Added!___\n\n");
    return c;
}
```



1. Başla
2. Yeni ders yarat ve dersin sınıf ortalamasını 0-a eşitle
3. Kullanıcıdan ders ismi ve kodunu al
4. Yeni yaratılmış dersi döndür
5. Son

Şema ve Sözde Kodun Açıklaması:

Program başlıyor, yeni ders yaratıp sınıf ortalamasını 0 yapıyoruz. Kullanıcıdan aldığımız ders ismi ve kodunu kaydedip dersi döndürüyoruz. Sonrasında programı bitiriyoruz.

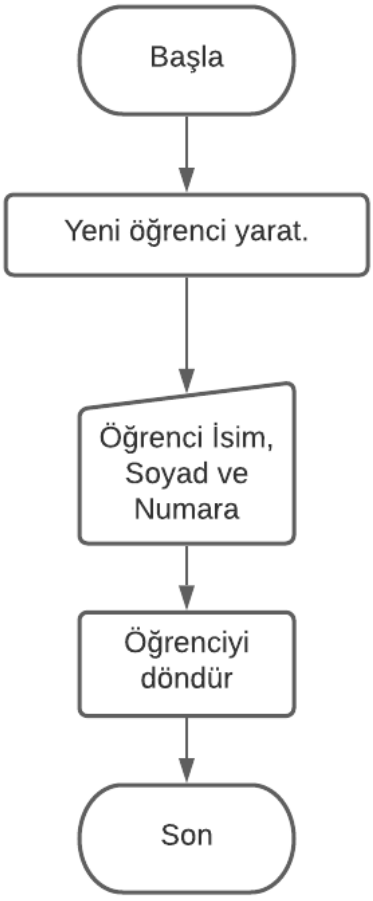
addStudent():

Burada öğrenci ekleme işlemini gerçekleştiriyoruz.”student_t s “ile yeni öğrenci ekliyoruz . stNo, studentName.name ve studentName.surname degiskenleri char array degilde char pointer verildiginden malloc fonksiyonu ile bellekte yer aciyoruz. Bütün ders isimlerin ilk harfini "" yapiyoruz*

*malloc(sayı, boyut) – bellekte sayı * boyut kadar yer açıyor. stNo, name ve surname pointer olduğundan önce yerlerini ayırıp sonra atama yapıyoruz.*

s.courses[i].courseName[0] = 42; - sonra hangi kayıt yeri hangisi boş anlamak için bütün kayıt yerlerine öğrenci yaratılırken “” yazdırıyoruz.*

```c
student_t addStudent()
{
    student_t s;

    s.stNo = malloc(STUDENTNUMBERLENGTH * (sizeof(*s.stNo)));
    s.studentName.name = malloc(NAMELENGTH * (sizeof(*s.studentName.name)));
    s.studentName.surname = malloc(NAMELENGTH * (sizeof(*s.studentName.surname)));

    for(int i = 0; i < COURSESREGISTERED; i++) s.courses[i].courseName[0] = 42;

    printf("\n\nStudent Name, Surname, No (etc. Serife Tan st1): ");
    scanf("%s %s %s", s.studentName.name, s.studentName.surname, s.stNo);
    printf("\n\n___Student Added!___\n\n");
    return s;
}
```



1. Başla
2. Yeni öğrenci yarat.
3. Kullanıcıdan öğrencinin ismini, soyadı ve numarasını al
4. Yeni yaratılmış öğrenciyi döndür
5. Son

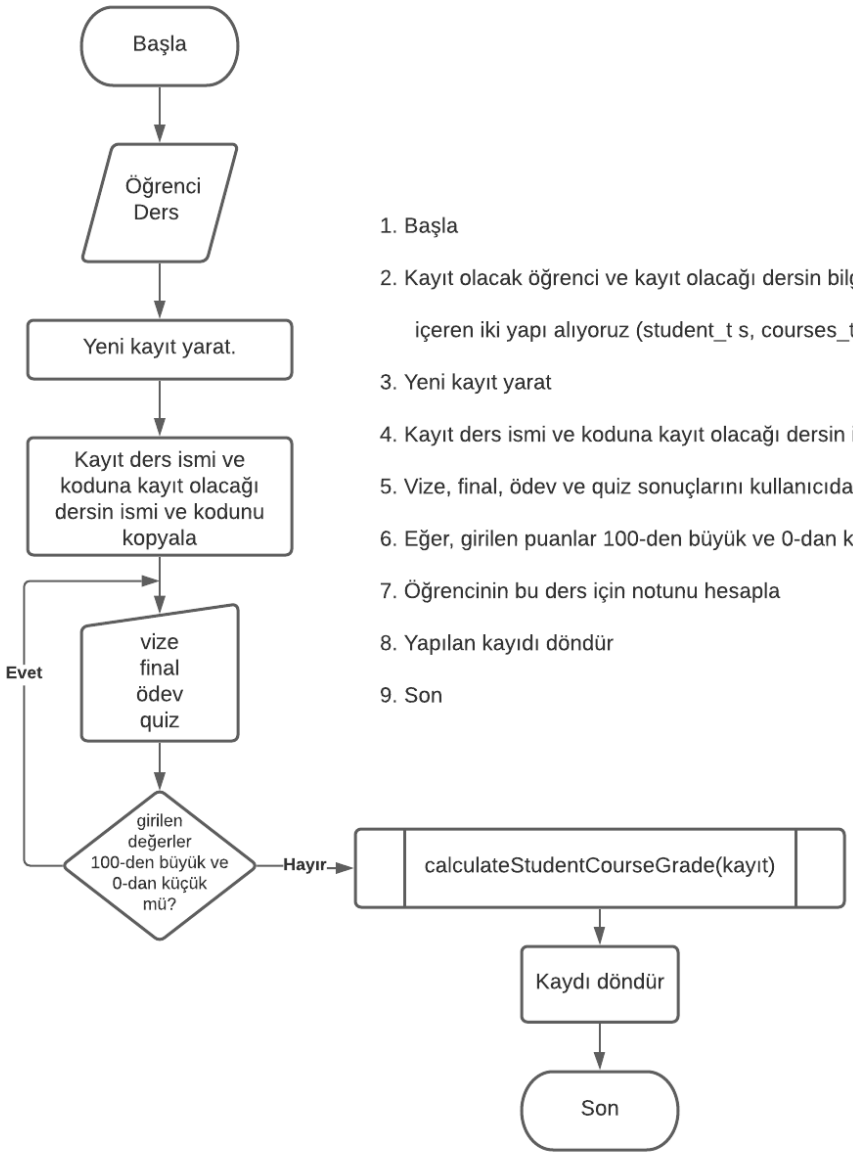
Şema ve Sözde Kodun Açıklaması:

Program başlıyor, yeni öğrenci yaratıyoruz. Kullanıcıdan aldığımız öğrenci ismi, soyismi ve numarasını kaydedip öğrenciyi döndürüyoruz. Sonrasında programı bitiriyoruz.

registerCourse():

registerCourse() fonksiyonu parametre olarak student_t s ve courses_t c yapılarınnı alıyor. Fonksiyonun başlangıcında r isimli bir courseReg_t yapısı oluşturuluyor. Ve integer i oluşturulup 0 a eşitleniyor. Sonraki 2 satırda parametre olarak aldığımız c-nin ders ismi ve kodunu strcpy(r.courseName, c.courseName), strcpy(r.courseCode, c.courseCode) ile r-ye kopyalıyoruz. strcpy(hedef, kaynak) – kaynağı hedefe kopyalayan fonksiyondur. . Ardından programımız "while" döngüsüne giriyor. "while"-a parametre olarak 1 veriyoruz ki, döngü biz durdurmadan bitmesin. Döngü bloğuna giriyoruz ve içerde kullanıcıdan tek tek vize, final, quiz ve homework notları alınıyor. Alınan notlar diğer if bloğunda her girilen not için 100-den küçük ve ya eşit ve 0-dan büyük ya da eşit ise o zaman döngüden çıkıyoruz değilse ekrana "Wrong Input" yazdırıp döngünün başına gidiyoruz. Döngüden çıktıktan sonra calculateStudentCourseGrade(&r) fonksiyonu çağırılıyor. Kaydın referansı fonksiyona öğrencinin nihai notunun hesaplanıp direk fonkiyonun içinde atama yapılması üzere gönderiliyor. Ardından program ilk başta yaratılan r değişkenini dönderiyor.

```
courseReg_t registerCourse(student_t s, courses_t c)
{
    courseReg_t r;
    int i = 0;

    strcpy(r.courseCode, c.courseCode);
    strcpy(r.courseName, c.courseName);

    while(1)
    {
        printf("\nEnter Grades\n \nMidterm: ");
        scanf("%lf", &r.grades.midterm);

        printf("\nFinal: ");
        scanf("%lf", &r.grades.final);

        printf("\nQuiz: ");
        scanf("%lf", &r.grades.quiz);

        printf("\nHomework: ");
        scanf("%lf", &r.grades.homework);

        if ((r.grades.midterm <= 100.00 && r.grades.midterm >= 0.00) &&
            (r.grades.final <= 100.00 && r.grades.final >= 0.00) &&
            (r.grades.quiz <= 100.00 && r.grades.quiz >= 0.00) &&
            (r.grades.homework <= 100.00 && r.grades.quiz >= 0.00))
            break;
        else printf("\n\nWrong Input!\n\n");
    }
    calculateStudentCourseGrade(&r);

    return r;
}
```



1. Başla
2. Kayıt olacak öğrenci ve kayıt olacağı dersin bilgilerini içeren iki yapı alıyoruz (student_t s, courses_t c)
3. Yeni kayıt yarat
4. Kayıt ders ismi ve koduna kayıt olacağı dersin ismi ve kodunu kopyalıyoruz
5. Vize, final, ödev ve quiz sonuçlarını kullanıcıdan alıyoruz.
6. Eğer, girilen puanlar 100-den büyük ve 0-dan küçük ise 5-ci adıma geri dön.
7. Öğrencinin bu ders için notunu hesapla
8. Yapılan kayıdı döndür
9. Son

Şema ve Sözde Kodun Açıklaması:

Program başlıyor, kayıt olacak öğrenci ve kayıt olacağı dersin bilgilerini içeren iki yapı alıyoruz. Yeni kayıt yaratıp kayıt ders ismi ve koduna kayıt olacağı dersin ismi ve kodunu kopyalıyoruz. Vize, final, ödev ve quiz sonuçlarını kullanıcıdan alıyoruz eğer, girilen puanlar 100-den büyük ve 0-dan küçük ise yeniden alıyoruz. Öğrencinin bu ders için notunu hesaplayıp yapılan kayıdı döndürüyoruz. Sonrasında programı bitiriyoruz.