Ders 49

Mülakat sorusu arası.

```
std::string reverse(std::string)

void reverse(char *p)
{

}


ila23y..?da983t++op--rak
ila23y..?da983t++op--rak
1
```

Yazı ters çevirilcek ancak karakterlere dokunulmayacak.

İ k değişsin

A k değişsin ama diğer karakterler yerinde kalsın.

```c
#include "date.h"

#define  PRIVATE            static
#define  PUBLIC
#define  YEARBASE           1900

#define  isleap(y)          ((y) % 4 == 0 && ((y) % 100 != 0 || (y) % 400 == 0))
#define  mdays(m, y)        (daytabs[isleap(y)][m])

PRIVATE const int daytabs[][13] = {
    {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
    {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
};



PRIVATE int is_valid_date(int d, int m, int y);



//private functions

PRIVATE int is_valid_date(int d, int m, int y)
{
    return y >= YEARBASE &&
        m >= 0 && m <= 12 &&
        d > 0 && d <= mdays(m, y);
}
```

Oluşturduğumuz date.c dosyasının başı

Makrolar

Private opp ye benzetmek için

Publıc sadece bize gözükücek benzetme amacı ile

Yearbase struct tm +1900 diye yaptık.

İsleap artık yıl makrosu

Mday makrosu iç içe makrodan oluşuyor 2 boyutlu bir char diziyi look up table olarak kullandık.

Ayın günleri geçerlimi diye bakıcaz.

Fonksiyonlar

İs valid date günler geçerlimi diye bakıyor bunu çok fazla kullanacağımız için fonksiyon olarak yazdık kod tekrarına düşmeyi engelleme amacındayız.

Privete sadece .date içinde kullanılacak du-ışarıya açılmayacak demek

Burda static olarak kullanılacak bir set fonksiyonu yazdık.

```
PRIVATE Date* set(Date* p, int d, int m, int y)
{
    if (!is_valid_date(d, m, y)) {
        fprintf(stderr, "gecersiz tarih!!!\n");
        exit(EXIT_FAILURE);
    }
    p->md = d;
    p->mm = m;
    p->my = y;
```

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

İlk public yani extern fonsiyonumuz

Static set fonksiyonunu kullandım. Değerleri set ettik .

```
// public functions

PUBLIC Date* set_date(Date* p, int day, int mon, int year)
{
    return set(p, day, mon, year);
}
```

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

İlk test fonkisoyunuz.

```
int main(void)
{
    Date mydate;

    set_date(p:&mydate, day:12, 4, year:1987);

}
```

Bu gerçerliydi

```
int main(void)
{
    Date mydate;

    set_date(&mydate, 29, 2, 2019);

}
```

Bu gercersiz ekrana geçersizdir yazıp exit ile çıktı.

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

Bugunu set eden fonksiyon

```
PUBLIC Date* set_today(Date* p)
{
    time_t seconds;
    time(_Time:&seconds);
    struct tm* tptr = localtime(_Time:&seconds);
    int day = tptr->tm_mday;
    int mon = tptr->tm_mon + 1;
    int year = tptr->tm_year + 1900;

    return set(p, day, mon, year);
}
```

Ben olsaydım

p->my=tptr->tm_year +1900 şeklinde yazardın sonuç olarak date türünün adresi var elimde ama hoca daha iyi anlayalım diye yaptı dene.

//

Sazıyı tarihe çevirelim

```
PUBLIC Date* set_date_from_str(Date* p, const char* pstr)
{
    int day = atoi(pstr);
    int mon = atoi(pstr + 3);
    int year = atoi(pstr + 6);

    return set(p, day, mon, year);
}
```

Atoi fonksiyonunu kullandım yazı değerini tarihe çevirdi formatınıda düzgün olarak tahmin ediyorum.

//

Timer değeri ile set edelim

```
Date* set_date_time_t(Date* p, time_t sec)
{
    struct tm* tptr = localtime(&sec);
    int day = tptr->tm_mday;
    int mon = tptr->tm_mon + 1;
    int year = tptr->tm_year + 1900;

    return set(p, day, mon, year);
}
```

Burada bir sey dikkatimi çekti kodun bir kısmı set_today ileaynı o zaman bu fonksiyonu set today ile kullanayım kod tekrarından kaçınalım.

```
PUBLIC Date* set_today(Date* p)
{
    time_t seconds;
    time(&seconds);

    return set_date_time_t(p, seconds);
}
```

Artık set today içi değişti bu güzel bir örnek kullanılması gerekir.

//

Random tarih oluşturma

```
#define  YEARBASE          1900
#define  RANDOM_MIN_YEAR   1940
#define  RANDOM_MAX_YEAR   2010
```

```
#define  mdays(m, y)       (daytabs[isleap(y)][m])
#define  ran_int(low, high)  (rand() % ((high) - (low) + 1) + (low))
```

```
PUBLIC Date* set_date_random(Date* p)
{
    int year = ran_int(RANDOM_MIN_YEAR, RANDOM_MAX_YEAR);
    int mon = ran_int(1, 12);
    int day = rand() % mdays(mon, year) + 1;

    return set(p, day, mon, year);
}
```

Yukardaki makroları anlaşılması kolay olsun diye yazdık.

Rand ifadesinide 2 yerde birden kullanıcaktık o yüzden makro yazdım.

Random olarak yıl ve ayı aldık.

Ama gün farklı günler için önceden oluşturduğumuz mday makrosunu kullandık bu sayede geçersiz gün girme ihtimalimiz ortadan kalktı.

//

Set year

```
Date* set_year(Date* p, int year)
{
    return set(p, p->md, p->mm, year);
}
```

Elemanları değişince bu sefer diğer kodlarıda değiştirmem lazım çünkü bağlantılı olacak.

```
Date* set_year(Date* p, int year)
{
    return set(p, get_month_day(p), get_month(p), year);
}
```

Kendi oluşturduğumuz ger fonksiyonlarını yazdık işin espirisi elemanları değiştirdiğimde diğer kodları değiştirmek yerine get fonksiyonlarını değiştiricem.

//

```
Date* set_month(Date* p, int mon)
{
    return set(p, get_month_day(p), mon, get_year(p));
}


//------------------------------------------------------------------
//------------------------------------------------------------------

Date* set_month_day(Date* p, int mday)
{
    return set(p, mday, get_month(p), get_year(p));
}
```

Diğer 2 koduda aynı mantıkla yazdık ama neden p-> kullanmadık tam mantığını anlamadım hocaya mutlaka sor.

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

Get fonksiyonlarını aynı mantıkla yazdık

```
PUBLIC int  get_year(const Date* p)
{
    return p->my;
}


//------------------------------------------------------------------
//------------------------------------------------------------------


PUBLIC int  get_month(const Date* p)
{
    return p->mm;
}

PUBLIC int  get_month_day(const Date* p)
{
    return p->md;
}
```

Tam nedenini anlamasamda aynı mantığa geliyor.

//

Yılın gününü hesaplayalım

```
//18 11 1987

PUBLIC int  get_year_day(const Date* p)
{
    int sum = get_month_day(p);
    const int mon = get_month(p);
    const int year = get_year(p);

    for (int i = 1; i < mon; ++i) {
        sum += mdays(i, year);
    }

    return sum;
}
```

//

Hafanın günü için sakamuto algoritması var

```
// array with leading number of days values
int t[] = { 0, 3, 2, 5, 0, 3, 5, 1, 4, 6, 2, 4 };

// if month is less than 3 reduce year by 1
if (m < 3)
    y -= 1;

return ((y + y / 4 - y / 100 + y / 400 + t[m - 1] + d) % 7);
```

hazır algoritma bu nunu düzünleyeceğiz.

```
PRIVATE int day_of_the_week(int d, int m, int y)
{
    static const int t[] = { 0, 3, 2, 5, 0, 3, 5, 1, 4, 6, 2, 4 };

    // if month is less than 3 reduce year by 1
    if (m < 3)
        y -= 1;

    return ((y + y / 4 - y / 100 + y / 400 + t[m - 1] + d) % 7);
}
```

Const staticyaptık düzünledik değişkennleri testini yapıcaz.

Yukardaki static olması içindi

```
PUBLIC int  get_weekday(const Date* p)
{
    const int day = get_month_day(p);
    const int mon = get_month(p);
    const int year = get_year(p);

    return day_of_week(day, mon, year);
}
```

Bu bizim paylaşıcağımız kod public olan

Haftanın hangi gününekarşılık geliyor o tarih onu yazdık.

//

Print date fonksiyonunu yazalım.

```
PUBLIC void print_date(const Date* p)
{
    static const char* const pdays[] = { ... }
    static const char* const pmons[] = { ... }

    const int day = get_month_day(p);
    const int mon = get_month(p);
    const int year = get_year(p);
    const int week_day = get_weekday(p);

    printf("%02d %s %d %s\n", day, pmons[mon], year, pdays[week_day]);
}
```

//

Scan fonksiyonunu yazalım

```
PUBLIC DATE* scan_date(Date* p)
{
    int day, mon, year;

    scanf(_Format: "%d%d%d", &day, &mon, &year);
    return set(p, day, mon, year);
}
```

Basit bir yazım mantığı

//

Tarihler arası fark

```
PRIVATE int totaldays(const Date* p)
{
    int sum = get_year_day(p);
    const int year = get_year(p);

    for (int i = YEARBASE; i < year; ++i) {
        sum += isleap(year) ? 366 : 365;
    }

    return sum;
}
```

Buradaki ana mantık tarihin bu zamana kadar gecen gün sayısını bulayım bu sekilde 2 tarihi bir birinden çıkartabilirim aradaki farkı bulabilirim.

```
int date_diff(const Date* p1, const Date* p2)
{
    return totaldays(p1) - totaldays(p2)
}
```

Hesaplama cok kolay oldu.

//

//

Kütüphanemiz bitti artık test fonksiyonlarına geçebiliriz.

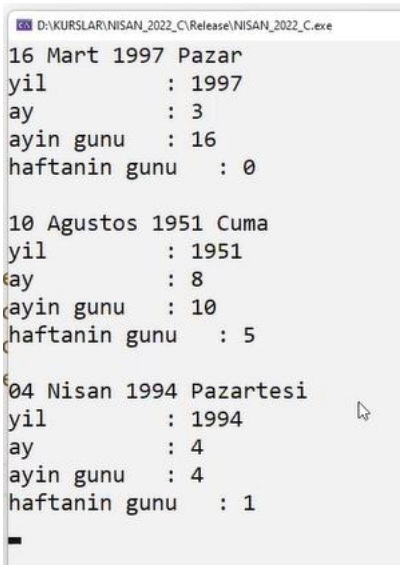
```c
int main(void)
{
    Date d1;

    set_date(&d1, 3, 5, 1992);
    print_date(&d1);

    Date d2;
    set_today(&d2);
    print_date(&d2);

    set_date_from_str(&d2, "24-02-2012");
    print_date(&d2);


    set_year(&d1, 2017);
    set_month(&d2, 1);
    set_month_day(&d1, 25);
    print_date(&d1);
    print_date(&d2);
}
```

```
Microsoft Visual Studio Debug Console
03 Mayis 1992 Pazar
29 Eylul 2022 Persembe
24 Subat 2012 Cuma
25 Mayis 2017 Persembe
24 Ocak 2012 Sali

D:\KURSLAR\NISAN_2022_C\Release\NISAN_202
Press any key to close this window . . .
```

//

Rasgele tarih testi

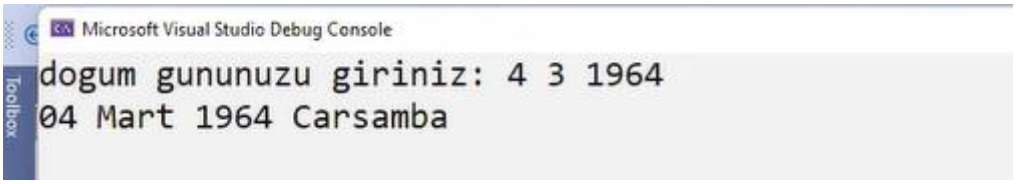
```c
int main(void)
{
    randomize();

    for (int i = 0; i < 10; ++i) {
        Date date;
        set_date_random(&date);
        print_date(&date);
        printf("yil          : %d\n", get_year(&date));
        printf("ay           : %d\n", get_month(&date));
        printf("ayin gunu    : %d\n", get_month_day(&date));
        printf("haftanin gunu    : %d\n", get_weekday(&date));
        (void)getchar();
    }

}
```



//

Hadtanın hangi günü doğduk bakalım test

```
int main(void)
{
    Date birth_date;

    printf(_Format:"dogum gununuzu giriniz: ");
    scan_date(p:&birth_date);
    print_date(p:&birth_date);

}
```



//

2 tarih arasındaki gün farkı bildirip sonra test edelim.

```
PRIVATE int totaldays(const Date* p)
{
    int sum = get_year_day(p);
    const int year = get_year(p);

    for (int i = YEARBASE; i < year; ++i) {
        sum += isleap(year) ? 366 : 365;
    }

    return sum;
}
```

Buradaki ana mantık tarihin bu zamana kadar gecen gün sayısını bulayım bu sekilde 2 tarihi bir birinden çıkartabilirim aradaki farkı bulabilirim.

Yukarda fonksiyon var

Test kodumuz.

```
int main(void)
{
    Date birth_date;

    printf("doğum tarihinizi girin: ");
    scan_date(&birth_date);
    print_date(&birth_date);
    Date todays_date;
    set_today(&todays_date);
    print_date(&todays_date);

    const int n = date_diff(&todays_date, &birth_date);

    printf("Bugun su fani hayatinizin %d gunu\n", n);

}
```

```
#include <stdio.h>
Microsoft Visual Studio Debug Console
do-um tarihinizi girin: 13 4 1997
13 Nisan 1997 Pazar
29 Eylul 2022 Persembe
Bugun su fani hayatinizin 9294 gunu
{
```

//

2 tarihin karşılaştırılması.

```
PUBLIC int cmp_date(const Date* p1, const Date* p2)
{
    const int p1_y = get_year(p1);
    const int p2_y = get_year(p2);

    if (p1_y != p2_y)
        return p1_y - p2_y;

    const int p1_m = get_month(p1);
    const int p2_m = get_month(p2);

    if (p1_m != p2_m)
        return p1_m - p2_m;

    return get_month_day(p1) - get_month_day(p2);

}
```

Aslında kendi tarih cmp fonksiyonumuzu yazdık güzel bir örnek.

Sort date array fonksiyonunuda yazmadık onuda yazmamız lazım

Teste kullanıcaz

```
void sort_date_array(Date* p, size_t n)
{
    qsort(p, n, sizeof(*p), &dcmp);
}
```

```
int dcmp(const void* vp1, const void* vp2)
{
    cmp_date(const Date *)vp1, (const Date*)vp2
}
```

Dcmp sqort ivin void türden yazalım.

Test fonksiyonumuz.

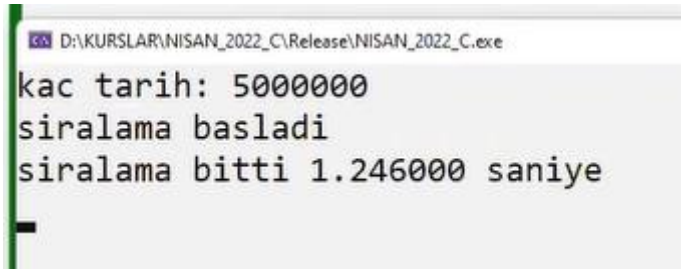
```
int dcmp(const void* vp1, const void* vp2)
{
    cmp_date(const Date *)vp1, (const Date*)vp2
}
```

```
void set_date_array_random(Date* p, size_t size)
{
    while (size--) {
        set_date_random(p++);
    }
}

void print_date_array(const Date* p, size_t size)
{
    while (size--) {
        print_date(p++);
    }
}
```

```
void sort_date_array(Date* p, size_t n)
{
    qsort(p, n, sizeof(*p), &dcmp);
}
```

```
{
    size_t n;

    printf("kac tarih: ");
    scanf("%zu", &n);

    Date* pd = (Date*)malloc(n * sizeof(Date));
    if (pd == NULL) {
        fprintf(stderr, "bellek yetersiz\n");
        return 1;
    }


    set_date_array_random(pd, n);
    printf("siralama basladi\n");
    clock_t start = clock();
    sort_date_array(pd, n);
    clock_t end = clock();
    printf("siralama bitti %f saniye\n", (double)(end - start) / CLOCKS_PER_SEC);
    _getch();
    print_date_array(pd, n);

    free(pd);
```

Test fonksiyonumuz.




```
03 Ocak 1976 Cumartesi
05 Ocak 1976 Pazartesi
05 Ocak 1976 Pazartesi
11 Ocak 1976 Pazar
22 Ocak 1976 Persembe
22 Ocak 1976 Persembe
30 Ocak 1976 Cuma
02 Subat 1976 Pazartesi
03 Subat 1976 Sali
03 Subat 1976 Sali
04 Subat 1976 Carsamba
08 Subat 1976 Pazar
10 Subat 1976 Sali
15 Subat 1976 Pazar
15 Subat 1976 Pazar
20 Subat 1976 Cuma
20 Subat 1976 Cuma
23 Subat 1976 Pazartesi
29 Subat 1976 Pazar
01 Mart 1976 Pazartesi
02 Mart 1976 Sali
02 Mart 1976 Sali
06 Mart 1976 Cumartesi
```

//

1 tarihten sonraki değeri değeri hesaplama n gün sonra ne olur gibi.

Yineyardımcı bir kodyazacağız.

```
PRIVATE int totaldays(const Date* p)
{
    int sum = get_year_day(p);
    const int year = get_year(p);

    for (int i = YEARBASE; i < year; ++i) {
        sum += isleap(year) ? 366 : 365;
    }

    return sum;
}
```

```
PRIVATE Date* to_date(Date* p, int totaldays)
{
    int year = YEARBASE;

    while (totaldays > (isleap(year) ? 366 : 365)) {
        totaldays -= (isleap(year) ? 366 : 365);
        ++year;
    }

    int mon = 1;
    while (totaldays > mdays(mon, year)) {
        totaldays -= mdays(mon, year);
        ++mon;
    }
```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!kodda hata var totaldays fonksiyonu içinde isleap(i) olmalı

Yardımcı olacak fonksiyonumuz.

```
PUBLIC Date* ndays_date(const Date* p, Date* pdest, int n)
{
    return to_date(pdest, totaldays(p) + n);
}
```

Dışarıya açık fonksiyon

```c
#include <conio.h>


int main(void)
{
    Date today;

    set_today(&today);

    printf("bugun:  ");
    print_date(&today);
    Date future_date;

    for (int i = 1; i <= 1'000'000'000; i *= 10) {
        printf("%d fun sonraki tarih\n", i);
        ndays_date(&today, &future_date, i);
        print_date(&future_date);
    }

}
```

Buda testkodum.

//

```
Microsoft Visual Studio Debug Console
bugun:  29 Eylul 2022 Persembe
1 gun sonraki tarih
30 Eylul 2022 Cuma
10 gun sonraki tarih
09 Ekim 2022 Pazar
100 gun sonraki tarih
07 Ocak 2023 Cumartesi
1000 gun sonraki tarih
25 Haziran 2025 Carsamba
10000 gun sonraki tarih
14 Subat 2050 Pazartesi
100000 gun sonraki tarih
14 Temmuz 2296 Sali
1000000 gun sonraki tarih
26 Agustos 4760 Cuma
10000000 gun sonraki tarih
25 Ekim 29401 Pazar
100000000 gun sonraki tarih
12 Haziran 275813 Cumartesi
1000000000 gun sonraki tarih
02 Ekim 2739929 Carsamba
```

Set todayi tekrar yaz unuttum.

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

/

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

Konuyu değiştirelim elemanı struct olan struct

Örnek

```
struct Person {
    char name[40];
    //Date m_bdate;
};


int main(void)
{

    printf("sizeof(struct Person) = %zu\n", sizeof(struct Person));

}
```

Boyutu suan 40

```
struct Person {
    char name[40];
    Date m_bdate;
};

int main(void)
{

    printf("sizeof(struct Person) = %zu\n", sizeof(struct Person));
    printf("sizeof(Date) = %zu\n", sizeof(Date));
```

Date türü 12 byte

52 byte oldu toplam

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

Örnek ilk değer

```
typedef struct {
    int m_id;
    char name[40];
    Date m_bdate;
}Person;


int main(void)
{
    Person per = { 28, "mustafa arslan", {2, 5, 1987} };

}
```

Deziknatır olur

//

Örnek elemana erişim

```
typedef struct {
    int m_id;
    char name[40];
    Date m_bdate;
}Person;


int main(void)
{
    Person per;
    //...

    per.m_bdate.mm

}
```

Sentax aynı . koy struct oldu bidaha nokta koy.

```
typedef struct {
    int m_id;
    char name[40];
    Date m_bdate;
}Person;


struct Team {
    //
    Person m_team_leader;
};

int main(void)
{
    struct Team tm;
    //...
    tm.m_team_leader.m_bdate.my
}
```

Burada 3 nokta kullandık.

//

Örnek person struct adresi olsaydı ne olurdu.

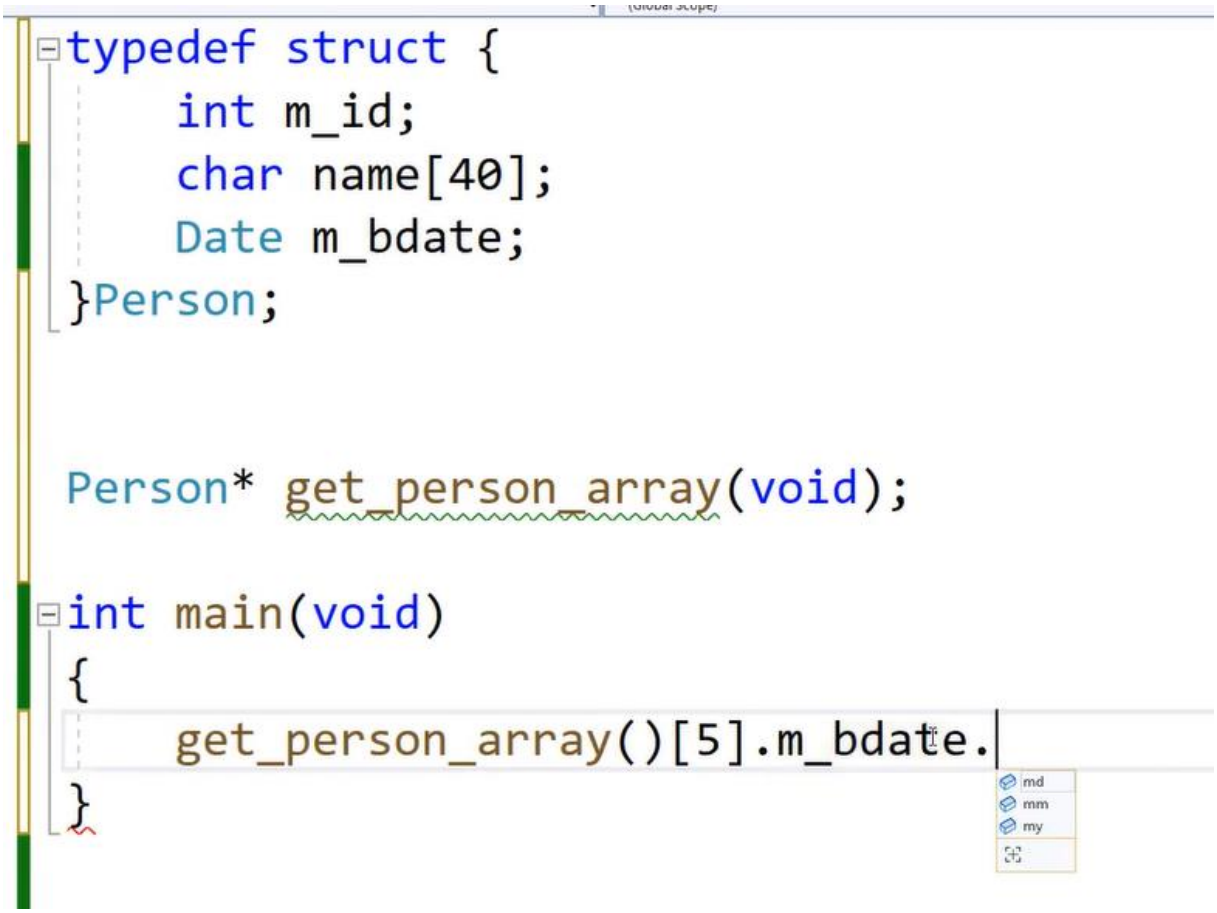
```
typedef struct {
    int m_id;
    char name[40];
    Date m_bdate;
}Person;


int main(void)
{
    Person per;
    //.... code
    Person* p = &per;

    p->m_bdate.my

}
```

Adresten date türüne erişirdim ama sonra . kullanmam lazım olurdu.

*p dizisi olsaydı ne olur.

```
    int m_id;
    char name[40];
    Date m_bdate;
}Person;


int main(void)
{
    Person* p[10];
    //code

    p[5]->m_bdate.my

}
```

Fonksiyonun geri dönüşü *person olsaydı.



Struct nesnesinin adresi var ve içindeki elemanın pointer struct

```
#include "date.h"

typedef struct {
    int m_id;
    char name[40];
    Date *mpdate;
}Person;


int main(void)
{
    Person* ptr;
    //code
    ptr->mpdate->my

}
```

2 tane ok kulandık.

Bir yapı nesnesinde kendi türüne ait pointer yapısı tanımlanabilir ancak kendi türünden yapı nesnesi oluşturulamaz.

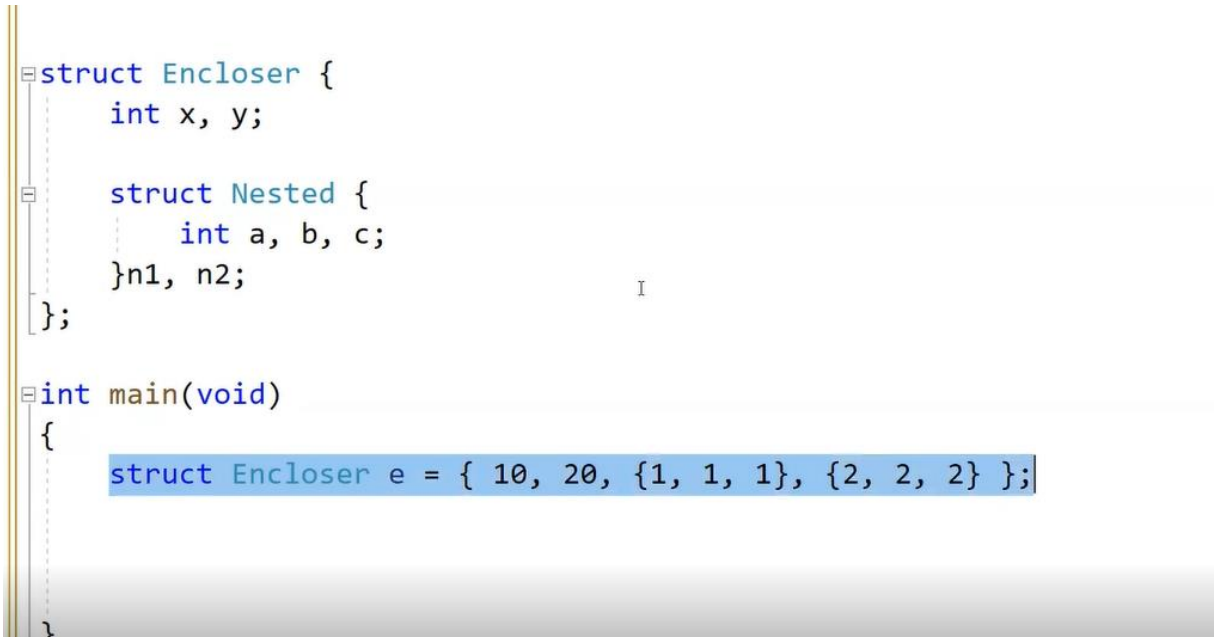
Farklı bir yapı nesneside tanımlanabilir.

```
struct Ertan {
    int mx, my;

    struct Volkan {
        double d1, d2;
    }v1, v2;
};
```

İçeride nesneyi tanımlasamda yapıların scopu olmadığı için bunları yapı dışındada kullanabilirim ilk akla gelen sekilde ömüru sıkıntı değil.

```
struct Encloser {
    int x, y;
    struct Nested {
        int a, b, c;
    }n1, n2;
};

int main(void)
{
    struct Nested a, b;

}
```

09_2022 on Vimeo - Google Chrome 2023-04-05 05-41-33

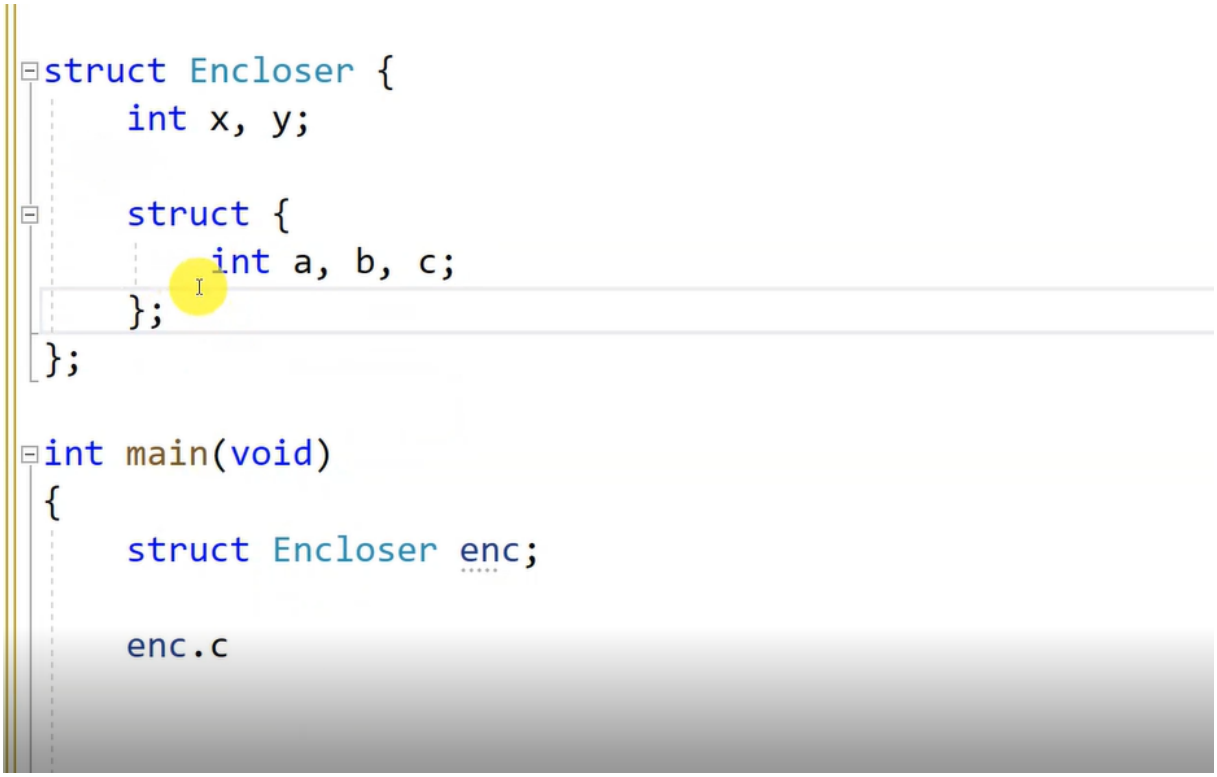


```
struct Encloser {
    int x, y;

    struct Nested {
        int a, b, c;
    }n1, n2;
};

int main(void)
{
    struct Encloser e = { 10, 20, {1, 1, 1}, {2, 2, 2} };
```

Yapının içinde ilk değer verememam ama dışında verebilirim .

Encloser nesnesi tanımlasaydık ona ilk değer verebilirdik.

//



```
struct Encloser {
    int x, y;

    struct {
        int a, b, c;
    };
};

int main(void)
{
    struct Encloser enc;

    enc.c
```

Yapı içnde ismi olmayan bir yapı tanımlarsam buna erişirken isimsiz yapının nesnelerinede enc.b gibi erişebilirim peki bunun ne gibi farydası var aşağıdakinden ne farkı var.

```c
struct Encloser {
    int x, y;

    int a, b, c;

};

int main(void)
{
    struct Encloser enc;

```

Yukardaki ile aşıdaki aynı gibi peki bunların farkı nedir.

Structure ile unionların birlikte kullanımında yardımcı oluyor.

```c
typedef struct {
    double d1, d2;
    struct {
        int a, b, c;
    }x, *ptr;

}Encloser;

int main(void)
{
    Encloser enc = { 2.3, 5.6, 10, 20, 30};

}
```

Bu yapı türünden pointerde olabilir c kütüphanelerinde çok sık çıkar.

//

İlginç özellik

```
//flexible array members
/
struct Data {
    int x, y, z;
    double a[];
};
```

Yapı içinde tanımladığımız diziye boyut yazmıyoruz normalde sentax hatası.

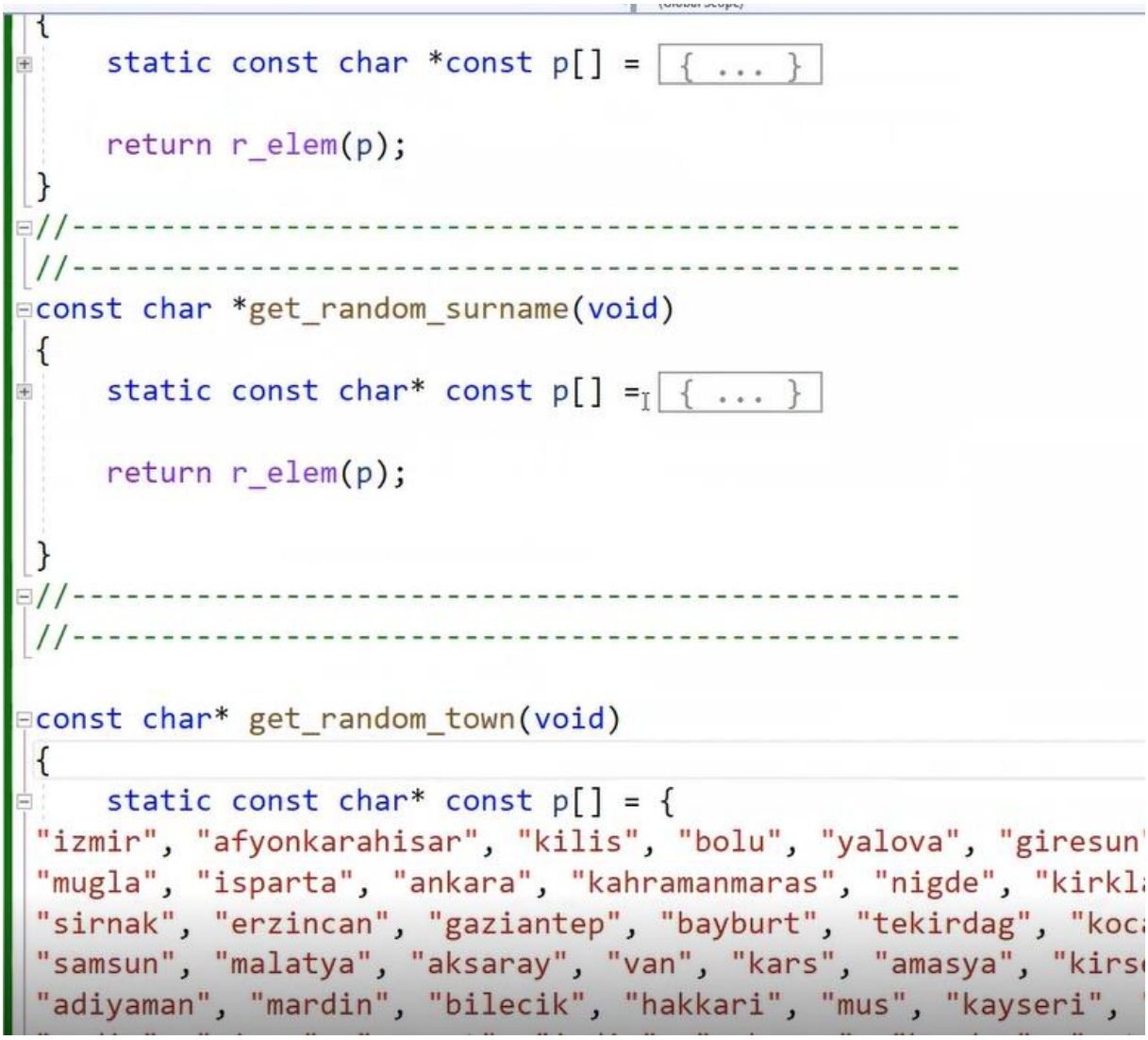
Ama yapı içinde geçerli.

//

Yeni bir kütüphane yazalım employe

```
#include "date.h"

typedef struct {
    int m_id;
    char m_name[20];
    char m_surname[20];
    char m_town[20];
    Date m_bdate;
}Employee;
```

```
Employee* set_employee_random(Employee*);
void print_employee(const Employee*);
int cmp_employee(const Employee*, const Employee*);
```



```
{
    static const char *const p[] = { ... }

    return r_elem(p);
}
//-------------------------------------------------
//-------------------------------------------------
const char *get_random_surname(void)
{
    static const char* const p[] = { ... }

    return r_elem(p);

}
//-------------------------------------------------
//-------------------------------------------------

const char* get_random_town(void)
{
    static const char* const p[] = {
"izmir", "afyonkarahisar", "kilis", "bolu", "yalova", "giresun
"mugla", "isparta", "ankara", "kahramanmaras", "nigde", "kirkl
"sirnak", "erzincan", "gaziantep", "bayburt", "tekirdag", "koc
"samsun", "malatya", "aksaray", "van", "kars", "amasya", "kirs
"adiyaman", "mardin", "bilecik", "hakkari", "mus", "kayseri",
```

İsimleri soy isimleri şehirleri rasgele veren fonksiyonlar oluşturdum.

Bu fonksiyonların geri dönüş değeri r_elem adında bir makro ile yazdım.

```
#define asize(x)      (sizeof(x) / sizeof(*x))
#define r_elem(a)     (a[rand() % asize(a)])
```

Rasgele dizinin bir adresini verecek.

```
{
    static const char* const ptowns[] = {
"izmir", "afyonkarahisar", "kilis", "bolu", "yalova", "giresun", "tunceli", "manisa",
"mugla", "isparta", "ankara", "kahramanmaras", "nigde", "kirklareli", "artvin", "kirik
"sirnak", "erzincan", "gaziantep", "bayburt", "tekirdag", "kocaeli", "trabzon", "ardah
"samsun", "malatya", "aksaray", "van", "kars", "amasya", "kirsehir", "balikesir", "esk
"adiyaman", "mardin", "bilecik", "hakkari", "mus", "kayseri", "agri", "sinop", "istanb
"aydin", "sivas", "yozgat", "igdir", "sakarya", "burdur", "antalya", "osmaniye", "kony
"zonguldak", "corum", "batman", "adana", "usak", "denizli", "edirne", "karaman", "ordu
"siirt", "kutahya", "bitlis", "bartin", "nevsehir", "rize", "kastamonu", "duzce", "erz
"hatay", };
    return ptowns[Irand(0, std::size(ptowns) - 1)()];
```

Static nesne adresi döndürüyor.

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

set random employe fonksiyonu yazalım yukardaki rasgele fonksiyonları bu yüzden tanımladık.

```
#include "employee.h"
#include "nutility.h"
#include <stdlib.h>
#include <string.h>

Employee* set_employee_random(Employee* p)
{
    p->m_id = rand();
    strcpy(p->m_name, get_random_name());
    strcpy(p->m_surname, get_random_surname());
    strcpy(p->m_town, get_random_town());
    set_date_random(&p->m_bdate);
}
```

Numara rasgele

İsim soyisim sehir şimdi tanımladık.

Date için zaten önceden tanımladığımız rasgele bir tarih tanımlayan fonksiyon vardı.

//

Print employe fonksiyonunu yazalım.

```
void print_employee(const Employee* p)
{
    printf("%-8d %-16s %-24s %-16s ", p->m_id, p->m_name, p->m_surname, p->m_town);
    print_date(&p->m_bdate);
}
```

Test kodu.

```
#include <conio.h>

int main(void)
{
    Employee e;
    randomize();
    for (int i = 0; i < 100; ++i) {
        set_employee_random(&e);
        print_employee(&e);
        _getch()
    }

}
```

Herşeyi fonksiyon ve yapılarla yapabiliyoruz çok güzek bişey.

```
28094   cebrail     tokatci     antalya     24 Aralik 1971 Cuma
10669   feraye      merdane     sirnak      25 Haziran 1956 Pazartesi
23281   eda         karayel     kirsehir    11 Agustos 1989 Cuma
16708   gursel      jilet       agri        02 Nisan 1951 Pazartesi
7277    sezen       uslu        kayseri     25 Ekim 1948 Pazartesi
18929   necmettin   akarsu      kirsehir    13 Ocak 1979 Cumartesi
24642   pelin       otaci       tunceli     06 Nisan 2004 Sali
24838   erdem       serinsun    sivas       17 Kasim 2009 Sali
17422   sabriye     saricakir   manisa      21 Haziran 1942 Pazar
14520   cahit       kesman      antalya     17 Mart 1997 Pazartesi
```

//

Şimdi cmp_employe kodunu yazalım.

```
int cmp_employee(const Employee* p1, const Employee* p2)
{
    int cmp_result = strcmp(p1->m_name, p2->m_name);

    if (cmp_result)     return cmp_result;

    cmp_result = strcmp(p1->m_surname, p2->m_surname);
    if (cmp_result)     return cmp_result;

    cmp_result = strcmp(p1->m_town, p2->m_town);
    if (cmp_result)     return cmp_result;

    cmp_result = cmp_date(&p1->m_bdate, &p2->m_bdate);
    if (cmp_result)     return cmp_result;

    return p1->m_id - p2->m_id;
}
```

Eğer sıfırsa yani eşitse devam yok eşit değilse devam et

En sona bir tek numrası kaldı ona baktık sonda.

```
int ecmp(const void* vp1, const void* vp2)
{
    return cmp_employee((const Employee*)vp1, (const Employee*)vp2);
}

int main(void)
{
    size_t n;

    printf("kac calisan: ");
    scanf("%zu", &n);
    Employee* pd = (Employee*)malloc(n * sizeof(Employee));
    if (!pd) {
        fprintf(stderr, "bellek yetersiz\n");
        return 1;
    }

    for (size_t i = 0; i < n; ++i) {
        set_employee_random(pd + i);
    }

    printf("siralama basladi\n");
    qsort(pd, n, sizeof(*pd), &ecmp);
    printf("siralama bitti\n");
    _getch();

    for (size_t i = 0; i < n; ++i) {
        print_employee(pd + i);
    }

    free(pd);
```

```
24573   alev    kalemsiz    istanbul    07 Aralik 2004 Sali
24803   alev    kalemsiz    karabuk     18 Nisan 1984 Carsamba
90      alev    kalinkas    siirt       18 Aralik 1962 Sali
28880   alev    kalpsiz     hatay       11 Haziran 1994 Cumartesi
13052   alev    kalpsiz     kirsehir    15 Mayis 1959 Cuma
4673    alev    kalpsiz     mardin      29 Mayis 1973 Sali
29701   alev    kalpsiz     nevsehir    14 Eylul 1985 Cumartesi
4250    alev    kalpten     cankiri     02 Mart 1959 Pazartesi
9206    alev    kapan       batman      07 Mayis 1942 Persembe
4477    alev    kapici      kastamonu   27 Ekim 1969 Pazartesi
19400   alev    kapici      tekirdag    25 Kasim 1941 Sali
23163   alev    kaplan      aydin       10 Kasim 1967 Cuma
13492   alev    kaplan      denizli     12 Haziran 1989 Pazartesi
28475   alev    kaplan      nigde       16 Mart 2000 Persembe
27790   alev    kaplan      sivas       26 Haziran 2001 Sali
4246    alev    kaplan      tekirdag    03 Ekim 1949 Pazartesi
```

Çok güzel bir örnek kesin tekrar et.