BİLGİSAYAR PROGRAMLAMA DERSİ NOTLARI

12.05.2020

Hamza ÇELİK

43hamzacelik@gmail.com

https://github.com/Bissmit

C Programlama Dili

İçindekiler Tablosu

İçindekiler Tablosu i

1. HAFTA 1

A. C Programlama Diline Giriş 1

a) C'nin Kullanıldığı Alanlar: 1

B. C Dilinin Temel Özellikleri ve Yapısı 1

a) Ön İşlemci Direktifi #include ve Kütüphaneler 2

b) Ana Fonksiyon main() 2

c) Yorum Satırları 3

d) Kod 3

C. C Dilinde Veri Tipleri, Değişkenler ve Sabitler 4

a) Değişkenler 4

b) Veri Tipleri 5

c) Sabitler 6

d) Backslash '\' Sabit Karakterleri 7

D. C Dilinde Operatörler 8

1. HAFTA ÖRNEKLER 11

1.1. Örnek: C dilinin genel yapısı ile ilgili basit kod yaziniz. 11

1.2. Örnek: Yorum satırlarını gösteriniz ve tek satır yorumu, çoklu satır yorumunun farkını gösteriniz. 11

1.3. Örnek: Veri tiplerinin, değişkenlerin, sabitlerin, operatörlerin kullanimini iceren bir kod yaziniz. 12

1.4. Örnek: Kullanicidan aldginiz karakterin ascii tablosundaki degerini gösteriniz. 14

1.5. Örnek: Kullanicinin dogum yilini alarak 2020 senesinde kac yasinda oldugunu bulunuz. 14

1.6. Örnek: Kullanicidan alinan 2 tane sayinin ortalamasini ve modunu hesaplayiniz. 15

1.7. Örnek: ..... **Hata! Yer işareti tanımlanmamış.**

2. HAFTA 16

A. Önişlemci komutları 16

a) #include Önişlemcisi 16

b) #define Önişlemcisi 16

1) Sayi Bildirme 17

2) Karakterleri Bildirme 17
3) Komut, Fonksiyonları Bildirme 17
c) #undef Önişlemcisi 18
d) #if, #else ve #endif Koşullu Derleme Önişlemcisi 18
B. KARŞILAŞTIRMA – KOŞULLLAR 19
a) if-else 19
b) switch-case 20
c) ? 21
C.DÖNGÜLER 21
a) for 21
b) while 22
c) do-while 24
Döngülerde dikkat etmemiz gerekenler: 24
Break: 25
Continue: 25
2. HAFTA ÖRNEKLER 26
2.1. Örnek: Önişlemci komutları ile ilgili genel bir örnek yapiniz. 26
2.2. Örnek: #undef in hakkinda kod yaziniz. 27
2.3. Örnek: if else ile kisinin ehliyet alip alamayacagini gosteren kod. 27
2.4. Örnek: Switch-case yapısına bir ornek veriniz. 28
2.5. Örnek: Kullancidan 1-12 arasi tam sayi alip o sayinin ay olarak karsiligini yazan programı yaziniz. 29
2.6. Örnek: Kullanicidan bir ayin sayisal degerini aliniz ve o ayin hangi mevisme ait oldugunu gosteren kodu switch-case yapisi ile yaziniz 30
2.7. Örnek: Kullanicidan yasini alip koronodan dolayi sokaga cikabilir mi cikamazmi kontolunu yaptırnız.(1-20 cikabilir, 65+ cikamaz) 31
2.8. Örnek: Alinan sayinin mutlagini hesaplayiniz if kosulu ile. 32
2.9. Örnek: Alinan sayinin mutlagini hesaplayiniz ? kosulu ile. 32
2.10. Örnek: Örnek 2.5 teki switch-case kodunu else if koşulu ile yapiniz(ayin ismini yazdirma): 33
2.11. Örnek: Kullanicidan alinan iki sayidan hangisinin kucuk, buyuk olduğunu hesaplayınız. 33
2.12. Örnek: Switch-case ile kare alma, mod alma, toplama, cikarma, bolme, carpma içeren secim menulu bir hesap makinesi yapiniz. 34

2.13. Örnek: Kullanıcıdan aldığınız 3 basamaklı sayının basamaklarındaki sayıları bulunuz. 35

2.14. Örnek: if-else koşulunun koşulunu dışardan alarak farklı bir yapıda oluşturunuz. 36

2.15. Örnek: Kullanıcıdan aldığınız sayi kadar ekrana merhaba yazdiriniz. 37

2.16. Örnek: Kullanıcıdan aldığınız sayinin tek mi çift mi olduğunu bulunuz. 37

2.17. Örnek: Kullanıcıdan aldığınız iki sayı arasında olan tek, çift, asal olan sayıları bulup ekrana yazdırınız ve aralıktaki sayilarin bolenlerini de yaziniz. Ve ikş sayi arasindaki sayilarin toplamini belirtiniz. 38

2.18. Örnek: Kullancıdan aldığınız sayinin faktoriyelini DONGU ile hesaplayiniz. 39

2.19. Örnek: İc ice donguler ile 1x9 carpim tablosunu yazdiriniz. 40

2.20. Örnek: ic ice donguler ile kullanicidan tek sayi alıp o sayi adedince satır içeren bir baklaba dilimi yazdırınız 40

2.21. Örnek: Kullanıcıya bütün ascii tablosundaki değerleri ve karşiliklarini gösteriniz. . 42

2.22. Örnek: Kullanıcıdan aldığınız n adet sayının ortalmasını hesaplayınız..... 43

2.23. Örnek: Kullanıcıdan aldığınız 16bitli 2 lik tabandaki sayı sistemini 10luk tabandaki sayı sistemine ceviriniz. 44

2.24. Örnek: Kullanıcıdan aldığınız bir sayının basamklarını tespit ediniz. Ve sayıın tersten halini bulun. 45

3. HAFTA 47

A. DİZİLER 47

B. DİZİ TANIMAMLAMA 48

C. DİZİLERİ EKRANA BASTIRMA 49

Dizi Boyutu Belirterek - Bütün dizi elmanlarını ekrana yazdırma: 49

Dizinin \0 null geçersiz son indisini arayarak - Bütün dizi elmanlarını ekrana yazdırma: . 49

D. DİZİYE KULLANICIDAN ELEMAN GİRDİRME 49

E. İKİ BOYUTLU DİZİLER 50

3. HAFTA ÖRNEKLER 51

3.1. Örnek: Dizilerin yapısıyla ilgili genel bir kod yazınız(dizi oluşturma,diziye eleman ekleme,gösterme vb). 51

3.2. Örnek: Disardan metin aliniz harf harf ekrana yazdiriniz vede ascii karşılıklarınıda hesaplayınız. 52

3.3. Örnek: Bir ders notu ortalaması hesaplama uygulaması yazınız. Dısardan n tane ders ismini alınız ve tutunuz o derse ait vize ve final notlarını tutunuz ve ortalamasını da ayrıca tutup en son çıktısını verin. 53

3.4. Örnek: Kullanıcıdan aldığınız metinde yine kullanıcıdan aldığınız karkaterin metinde hangi indislerde oldugunu baka dizide tutup ayrıca kaç kere tekrar ettigini bulup belirtin. .. 54

3.5. Örnek: Alınan integer sayının rakamlarının okunusunu yazı ile yazdırma. 55

3.6. Örnek: Dışardan alınan tam sayıları diziye atıp kucukten buyuge buyukten kucuge sıralama. 57

3.7. Örnek: Belirlediginiz bir metinin dışardan aynı şekilde girilmesini bekleyiniz doğru girilmez ise aynı işlem doğru girileseye kadar tekarlansın. 59

3.8. Örnek: Aşağıdaki kare matrisin kodunu yazınız. 60

3.9. Örnek: Disardan alinan metnin tesrten yazdirilmasi. 61

3.10. Örnek: 61

4. HAFTA 62

A. GÖSTERİCİLER (pointer) ve GENEL YAPISI 62

1) Değişkenlerin adreslerini gösterme (&): 62

2) Göstericiler (*): 62

3) Göstericilerin Tanımlanması 63

B. Gösterici ve diziler arasındaki bağlantı. 63

4. HAFTA ÖRNEKLER 65

4.1. Örnek: Pointerların genel yapısıyla ve kullanımı ile alakalı örnekler veriniz. 65

4.2. Örnek: Pointerlarla vize final notu alıp ortalamasını hesaplayın (vize0.40 – final0.60) 66

4.3. Örnek: Belirlenen diziyi iki farklı pointer yolu ile ekrana yazdiriniz. 67

4.4. Örnek: Belirli diziyi pointerla ile tersten yazdiriniz. 68

5. HAFTA 69

A. FONKSİYONLAR 69

B. 69

5. HAFTA ÖRNEKLER 72

5.1. Örnek: 72

5.2. Örnek: 73

5.3. Örnek: 74

5.4. Örnek: 75

6. HAFTA 83

A. DİNAMİK BELLEK KULLANIMI ve YÖNETİMİNİN KAVRANMASI 83

B. DİNAMİK BELLEK FONKSİYONLARI 83

1) malloc(): Bellekte alan ayırma. 83

2) calloc(): Bellekte alan ayırma ve bitlere 0 atama. 84

3) realloc(): Daha önce ayrılan belleğin boyutunu değiştirme 84

4) free(): Daha önce ayrılan bellek alanını boşaltma. 85

C. ARASINDAKİ FARK - (int *)malloc(20) - (int *)malloc(sizeof(int)*5) 85

6. HAFTA ÖRNEKLER 87

6.1. Örnek: malloc(), calloc(), realloc(), free() ile ilgili genel örnek. 87

6.1. Örnek: Disaridan girilen n kadar dinamik char dizisi olusturup, kullanicidan gets() ile icine metin atayip, girilen metin ile ilgili alan yonetiminin yapilmasi disardan dinamik diziye girilen alan dizinin olsutugu alandan kucuk veya fazla ise bunu otomatik duzeltme. 98

6.1. Örnek: Disaridan alinan n boyutluk float tipinde malloc ile bir dinamik dizi olsuturnuz ve diziye deger atayiniz ardindan dizide realloc ile alan degisikligi yapiniz. 99

6.1. Örnek: Disaridan alinan n adet boyutta ineteger dinamik dizi oluşturup, dizi içine dışardan değer atanıp, daha sonra boyutu artırılıp, yeni boyuta gore eski diziyi kaybetmeden eski dizi kayıtlarının yanına yeni integer veri girişi. 101

6.2. Örnek: Malloc ve Calloc Farkı 102

6.3. Örnek: 104

7. HAFTA 105

A. STRİNGLER 105

B. STRİNG.H FONKSİYONLARI 106

7. HAFTA ÖRNEKLER 111

7.2. Örnek: Kütüphane kullanmadan string bir dize tanımlayıp bastırın. 111

7.3. Örnek: strcat(),strcpy(),strcmp() kullanimina ornek. 111

7.4. Örnek: Kelimeleri kucukten buyuge,buyukten kucuge siralama. 112

7.5. Örnek: strlen**,** strchr ve strrchr kullanımı. 113

7.6. Örnek: strlwr,strupr,puts kullanımı. 114

8. HAFTA 115

A. MATEAMTİKSEL İSLEMLER(math.h) 115

8. HAFTA ÖRNEKLER 118

8.1. Örnek: Math kütüphanesi genel örnek, sqrt, pow, floor, ceil, fabs, log, sin, cos 118

8.2. Örnek: 121

9. HAFTA 121

A. STRUCTLAR 121

B. 122

9. HAFTA ÖRNEKLER 123

9.1. Örnek: 123

9.2. Örnek: 123

9.3. Örnek: 123

9.4. Örnek: 123
10. HAFTA 124
A. DOSYA İŞLEMLERİ 124
B. DOSYA TANIMLAMA 125
C. DOSYAYA BİLGİ KAYDETME 125
Kaynakça 127

1. HAFTA

A. C Programlama Diline Giriş

C, 1972 yılında Dennis Ritchie tarafından Bell Laboratuvarlar'ında bir işletim sistemi yazmak için bir sistem programlama dili olarak geliştirilen genel amaçlı programlama dilidir. C, en yaygın kullanılan ve popüler Sistem Programlama Dilidir. En gelişmiş yazılımların çoğu C kullanılarak geliştirilmiştir. C; UNIX, Microsoft, Linux, gibi birçok işletim sisteminin yazımında kullanılmıştır. C programlama, Java, Python vb. Gibi çoğu programlama dilinden daha hızlıdır, öğrenilmesi diğer dillere göre kolaydır.

a) C'nin Kullanıldığı Alanlar:

- Gömülü sistemler.
- İşletim sistemleri geliştirmede.
- Veritabanları geliştirmek için kullanılır. MySQL, 'C' kullanılarak oluşturulan en popüler veritabanı yazılımıdır.
- Masaüstü uygulama geliştirmede kullanılır Adobe uygulamalarının çoğu C ile geliştirilmiştir.
- IOT uygulamaları geliştirmede.
- Tarayıcıları ve uzantılarını geliştirmek için kullanılır. Google'ın Chromium'u 'C' programlama dili kullanılarak oluşturulmuştur.
- Derleyici üretiminde kullanılır.

B. C Dilinin Temel Özellikleri ve Yapısı

C, yapısal dil, zengin kütüphane işlevleri , veri türleri , vb. ile basit bir dildir . C dilinin derleme ve çalıştırma süresi hızlıdır, verimli programlar üretir, çeşitli bilgisayar platformlarında derlenebilir.
Program yazımı belirli kalıpta, bloklar halinde olur. Bloklar, { } parentezleri ile oluşturulur. Komutlar aynı veya alt alta satırlara yazılabilirler. Tüm komutlar, noktalı virgül (;) ile bite. Yalnız blok başlatan ifadelerden sonra noktalı virgül kullanılmaz.

Programda kullanılan tüm değişkenler ve veri tipleri bildirilir, programda kullanılacak olan komutların bulunduğu kütüphaneler çağırılırak. C dilinin genel yapısını ilk örneğimiz üzerinden inceleyelim.

1_1_ornek.c

```
1.      #include <stdio.h> //stdio.h, standart giriş çıkış kütüphanesi
çağırılmıştır.
2.
3.      int main() //program çalışınca başlayan ana fonksiyondur.
4.      {
5.
6.              printf("Merhaba Dünya!"); //Ekrana çıktı veren komuttur,
stdio.h kütüphanesinin içinde barınır.
7.              return 0; //Programı sonlandırma işlevi
8.
9.      }
```

a) Ön İşlemci Direktifi #include ve Kütüphaneler

Ön işlemci direktfileri C dilinde kodun en üstüne yerleştiririz. C de komutlar/fonksiyonlar belirli kütüphanelerde toplanmıştır. C dilinde programda kullancağımız, komutları ve fonksiyonları barındıran Kütüphaneleri dahil etmemizi sağlar ve şu şekilde kullanılır:

```
#include <kütüphaneAdı>
```

İlk başlarda bilmemiz gereken kütüphaneler ise şunlardır:

```
stdio.h = Standart giriş/çıkış kütüphanesidir. Bu kütüphaneyi neredeyse her
kodumuzda kullanırız çok öenmlidir. Mesela ekranda bir metin gosterilerek
kullanıcıya bilgi verilecek ise stdio.h kütüphanesi olmazsa olmazımızdır.

conio.h = DOS destekli giriş/çıkış, genellikle Windows işletim sistemi
kullananların sık sık kullandığı kütüphanesidir. En çok getch(); komutu
kullanılır çünkü windows işletim sistemi kullananlar konsol bu komut olmadan
açılıp kapanır, kapanamasını önlemek için bu kullanıcıdan bir girdi bekleyen
bu komut kullanılır.

math.h = Matematiksel fonksiyonlar kütüphanesidir. Mesela cos, log hesaplama.

string.h = Alfasayısal ve bazı bellek yönetimi kütüphanesidir.
```

b) Ana Fonksiyon main()

main() fonksiyonu her C programının bir parçasıdır. Bir C programında ilk önce bu fonksiyonun içindeki kodlar yürütülür. Ve ... main() nin önündeki veri tipi return komutunda döndürülür.

main fonksiyonu bu şekillerde oluşturabiliriz:

- main()
- int main()
- void main()
- main(void)
- void main(void)
- int main(void)

c) Yorum Satırları

Programda ne yaptığımızı açıklamak için programımıza yorum satırı ekleyebiliriz. Bu yorumlar derleyici tarafından yoksayılır ve yürütülmez. Tek bir satır yorum eklemek için, iki eğik çizgi ve // ardından yorumu ekleyerek tek satırlık yorum satırı eklenir. Çok satırlı yorum eklemek için, /*.....*/ operatörleri arasına alınır satırlarımız.

Tek satırlık yorum şu şekilde kullanılır:

```
//Burası tek satırlık bir yorumdur.
```

Çok satırlı yorum satırı ise en başa **/*** ve en sonra ***/** operatörleri konularak belirtilebilir. Bu iki operatör arasındaki ifadelerin tümü birer yorum satırı olarak değerlendirilir.

```
/*Burası da
                çok satırlı bir
                                yorum satırıdır. */
```

d) Kod

Ana fonksiyon (main()) bildirildikten sonra, açılış ve kapanış parantezlerini belirtmeliyiz. Süslü parantezler {}, bir programın başlangıcını ve sonunu gösterir. Bu parantezler her zaman ana fonksiyondan sonra konulmalıdır. Tüm program kodu ve çalıştırılacak kod bölümü bu parantezlerin içine yazılır. printf(“Merhaba Dünya!”)

fonksiyonu ise ekrana “Merhaba Dünya!” çıktısını verir. return 0 kodu ise main() fonksiyonuna 0 döndürür ve program sonlanır.

C. C Dilinde Veri Tipleri, Değişkenler ve Sabitler

Bilgisayarda işlenen veriler temelde iki tür vardır: Sayısal ve Alfasayısal dır. C dilinde de kullanılacak değişkenler ve veri tipleri programda önceden bildirilmek zorundadır.

a) Değişkenler

Değişkenler bir programlama dilinin en önemli bileşenlerindendir. En basit bir aritmetik işlemin bile kullanıcının girdiği değerleri saklamak için çeşitlik bellek alanlarına ihtiyacı vardır. İşte değişkenler bu bellek adreslerine verilen isimlerdir. Değişkenler bellekte bilginin saklandığı hücreler verilmiş sembolik adlardır. Her değişkenin tuttuğu değerin nasıl bir veri olduğunu gösteren bir tipi vardır. [1]

C gibi yüksek seviyeli programlama dilleri programcıyı bellek adreslerinin karmaşasından kurtararak değişkenler aracılığıyla bellek alanlarına takma isimler vermemize olanak tanırlar.

Değişken isimleri oluşturulurken uyulması gereken kurallar şunlardır:

1. Değişken adı yalnızca karakter, rakam ve alt çizgi içermelidir.
2. Değişken adı bir sayı ile başlamamalıdır.
3. Değişken adı boşluktan oluşmamalıdır.
4. Değişken adı C diline ait tanımlamalardan oluşmamalıdır.
5. 'C', ‘yas’ ve 'YAS' adlı bir değişkenin farklı olduğu anlamına gelen büyük / küçük harfe duyarlı bir dildir.
6. Tükçe karakter karakter kullanılmamalı.

Doğru bir şekilde değişken isimlendirme:

```
height yada HEIGHT
_height
_height1
My_name
Turkce
```

```
int degiskenIsmı = 7; //ŞELİNDE TANIMLANABİLİR.
```

Geçersiz değişken isimlendirme, değişkenlerinizi bu şekilde oluşturmayın:

```
1height
Hei$ght
My name
türkçe
```

[1] https://www.dijitalders.com/icerik/106/5440/c_programlama_dilinde_degiskenler.html

b) Veri Tipleri

C dilinde de kullanılacak değişkenler ve veri tipleri programdan önce bildirilmek zorundadır.

Beş temel veri tipi vardır,

1. Tam sayılar için, int
2. Karakterler için, char
3. Ondalıklı sayılar için, float
4. Daha hassas ondalıklı sayılar için, double
5. void

Dizi, fonksiyonlar, işaretçiler, yapılar türetilmiş veri türleridir. 'C' dili, yukarıda belirtilen temel veri türlerinin daha genişletilmiş sürümlerini sağlar. Her veri tipi boyut ve aralık bakımından birbirinden farklıdır. Aşağıdaki tabloda her veri tipinün boyutu ve aralığı gösterilmektedir.

VERİ TİPİ	BOYUT (BYTES)	ARALIK	BİÇİM
short int	2	-32,768 to 32,767	%hd
unsigned short int	2	0 to 65,535	%hu
unsigned int	4	0 to 4,294,967,295	%u
int	4	-2,147,483,648 to 2,147,483,647	%d
long int	8	-2,147,483,648 to 2,147,483,647	%ld
unsigned long int	8	0 to 4,294,967,295	%lu
long long int	8	-(2^63) to (2^63)-1	%lld
unsigned long long int	8	0 to 18,446,744,073,709,551,615	%llu
signed char	1	-128 to 127	%c
unsigned char	1	0 to 255	%c
float	4	3.4E-38 to 3.4E+38	%f
double	8	1.7E-308 to 1.7E+308	%lf
long double	16	3.4E-4932 to 1.1E+4932	%Lf

Integer veri tipi:

Integer tam sayıları tutar. Bir integer veri tipi için aralık makineden makineye değişir. Bir integer veri tipi için standart aralık -32768 ila 32767'dir.

```
int yas;
```

Şeklide yaşımızı bu veri tipinde tutabiliriz.

```
int yas = 20;
```

Ve bu şeklidede yaş değişkenimize 20 tam sayısını atadık.

Float veri tipi:

Float ondalıklı sayıları tutar.

```
float oran = 0.4;
```

Double veri tipi:

Double veri tipide ondalıklı sayıları tutmamızı sağlar ama float tan daha hassas sayilarida tutarken kullanılır.

Char veri tipi:

Karakter veri türleri, tek tırnak içine alınmış tek bir karakter değerini saklamak için kullanılır. Dizi olarak ta kullanılabilir.

```
char karakter = ‘!’;

char dizi[100] = “karaketer dizisi”;
```

c) Sabitler

C sabitleri, normal değişkenler gibidir. Ama bir kez değer atandıktan sonra program tarafından değiştirilemez.

Sabit değerlerde kullanılır. Bu sayede değiştirilmesine izin verilmez. Örneğin Pi sayısı.

Sabit değerlerin veri tipleri kısıtlaması yoktur. Her veri tipindeki değerleri sabit olarak tanımlayabilirsiniz.

```
char dizi[100] = “karaketer dizisi”;

const float PI_kisa= 3.141
```

[2]**Sabit bildirimi**, başlangıç değeri verilen değişken bildirimi gibi yapılır. Ancak, veri tipinin önüne *const* anahtar sözcüğü konmalıdır.

const float PI = 3.142857;

const double NOT= 12345.8596235489;

const int EOF= -1;

const char[] = “devam etmek için bir tuşa basın…”;

gibi sabit bildirimleri geçerli olup bunların içerikleri program boyunca değiştirilemez. Yalnızca kullanılabilir. Genellikle, sabit olarak bildirilen değişken isimleri büyük harflerle, diğer değişken isimlerinin ise küçük harflerle yazılması (gösterilmesi) C programcıları tarafından geleneksel hale gelmiştir. Birçok C programında sabitler #define önişlemci komutu ile de tanımlandığını görebilirsiniz. Bu komutla sabit bildirimi, bir program parçasına ve makro fonksiyon tanımlaması yapılabilir. Bir program geliştirilirken simgesel sabitlerin kullanılması programın okunurluğunu arttırır ve bazen gerekli de olabilir.

#define MAX 100

#define DATA 0x0378

#define YARICAP 14.22

d) Backslash ‘\’ Sabit Karakterleri

C dilinde özel anlamı olan bazı karakterler vardır. Bu özel anlamı olan fonksiyonlardan faydalanmak için sembollerin önünde BackSlash ‘\’ olmalıdır.

Aşağıda verilen özel karakterler ve onların açıklamaları bulunmaktadır.

BackSlash ‘\’ Karakterler	Açıklama
\b	Bir önceki karakteri siler (Backspace)
\f	Bir sonraki sayfanın başına geçer (form feed)
\n	Bir alt satıra geçer (newline)
\r	Satır başı yapar (carriage return)

[2] https://medium.com/@dijitalgezginler16/c-programlama-dili-veri-tipleri-de%C4%9Fi%C5%9Fkenler-ve-sabitler-1ce8c43003c1

BackSlash '\' Karakterler	Açıklama
\t	Yatay TAB (Horizontal TAB)
\"	Çift tırnak (") karakterini ekrana yazmak için
\'	Tek tırnak (') karakterini ekrana yazmak için
\\	BackSlash '\' karakterini ekrana yazmak için
\v	Dikey TAB (Vertical TAB)
\a	Uyarı sesi üretir (Alert)
\?	Soru işareti '?' karakterini ekrana yazmak için
\N	Sekizlik tabanda sabit (N sekizlik tabanda bir sabittir.)
\XN	Onaltılık tabanda sabit (N onaltılık tabanda bir sabittir.)

D. C Dilinde Operatörler

Birinci öncelikliler	x++, x--
Tek operand alan operatörler	+, -, !, ~, ++x, --x, (Tür)x
Çarpma ve bölme, mod	*, /, %
Toplama ve çıkarma	+, -
Kaydırma operatörleri	<<, >>
İlişkisel ve tür testi operatörleri	<, >, <=, >=, is, as
Eşitlik operatörü	==, !=
Bitsel Ve (AND)	&
Bitsel Özel Veya (XOR)	^
Bitsel Veya (OR)	|
Mantıksal Ve	&&
Mantıksal Veya	||
Koşul operatörü	?:
Atama ve işlemli atama operatörleri	=, *=, /=, %=, +=, -=, <<=, >>=, &=, ^=, |=

Operatörler önceden tanımlanmış birtakım matematiksel ya da mantıksal işlemleri yapmak için kullanılan özel karakterler ya da karakterler topluluğudur.

Bazı ifadelerde birden fazla operatör kullanılmış olabilir. Bu gibi durumlarda operatörlerin belirli bir çalışma sırası olacaktır.

Operatörler Oncelik Sırası

	Operatörlerin Öncelik Sırası	
Yüksek öncelik	Operatörler	Öncelik Yönü
	( ) [ ] -> .	soldan sağa
	! ~ ++ -- + - * & (type) sizeof	sağdan sola
	* / %	soldan sağa
	+ -	soldan sağa
	<< >>	soldan sağa
	< <= > >=	soldan sağa
	== !=	soldan sağa
	&	soldan sağa
	^	soldan sağa
	|	soldan sağa
	&&	soldan sağa
	||	soldan sağa
	?:	soldan sağa
Düşük öncelik	= += -= *= /= %= &= ^= |= <<= >>=	sağdan sola
	,	soldan sağa

[3]

[3] Operatör oncelik sırası kaynak: https://turkmuhendis.net/wp-content/uploads/2019/04/%C4%B0%C5%9Flem-%C3%96ncelik-S%C4%B1ras%C4%B1-.jpg

1. HAFTA ÖRNEKLER

1.1. Örnek: C dilinin genel yapısı ile ilgili basit kod yaziniz.

1_1_ornek.c

```
1. #include <stdio.h> //stdio.h, standart giriş çıkış kütüphanesi çağırılmıştır.
2.
3. int main() //program çalışınca başlayan ana fonksiyondur.
4. {
5.
6. printf("Merhaba Dünya!"); //Ekrana çıktı veren komuttur, stdio.h
   kütüphanesinin içinde barınır.
7. return 0; //Programı sonlandırma işlevi
8.
9. }
```

ÇIKTI:

```
Merhaba Dünya!
```

1.2. Örnek: Yorum satırlarını gösteriniz ve tek satır yorumu, çoklu satır yorumunun farkını gösteriniz.

1_2_ornek.c

```
1. // Bu bir tek satirlik yorumdur
2. // Bu sayede kodlarımızın ne ise yardigi hakkinda aciklama yapabiliriz
3.
4. /*        bu ise coklu yorum
5.                                satiridir daha fazla satiri yorum yapmak icin kullaniriz.*/
6.
7. //Program Aciklma Yorum Satiri Ornekleri Kodu:
8.
9. #include <stdio.h>
10.
11. int main()
12. {
13.    printf("Merhaba Dünya!\n"); //Bu kod calisacaktir ve ekrana Merhaba
   Dunya! basilacaktir.
14. // printf("Merhaba Uzay!"); Ancak bu kod calismayacaktir cunku yorum
   satirlari arasinda kalmistir. Derleyici bu kod yokmus gibi hareket edecektir.
15.    printf("Merhaba Uzay!"); //Bu kod herhangi bir yorum satırı icinde
   olmadigindan calisacktir ve ekrana Merhaba Uzay! basilacaktir.
```

```
16.
17.    return 0;
18. }
```

ÇIKTI:

```
Merhaba Dünya!
Merhaba Uzay!
```

1.3. Örnek: Veri tiplerinin, değişkenlerin, sabitlerin, operatörlerin kullanimini iceren bir kod yaziniz.

1_3_ornek.c

```
1.  #include <stdio.h>
2.
3.  //define sabiti
4.  #define PI 3.141
5.  #define Boyut 10
6.  #define topla(a, b) a+b
7.
8.  int main()
9.  {
10.         //const sabiti
11.         const int boyut = 10;
12.
13.         //tam sayi tanimlama
14.    int tamSayi = 1;
15.
16.    //karakter tanimlama
17.    char karakter ='X';
18.
19.    //karakter dizi tanimlama
20.    char dizi[Boyut] = "Dizi";
21.    //karakter dizisinin alani define sabitine ait olabilir ama const olmaz hata
       verir
22.    //char dizi2[boyut] = "Dizi2"; hata verir const almaz
23.
24.    //ondalikli sayi tanimlama
25.    double ondalikli_1 = 5.48;
26.    float ondalikli_2 = 5.48;
27.
28.    printf("Define sabiti ile PI sayisini tuttuk: %f\n",PI);
29.    printf("integer tipinde tamSayi degiskeninin degeri: %d\n",tamSayi);
30.    printf("const int tipinde boyut degiskeninin degeri: %d\n",boyut);
```

```
31.     printf("Define sabiti ile toplama islemini yaptirdik (int tamSayi + const int) boyut: %d\n",topla(tamSayi,boyut));
32.     printf("double tipinde ondalikli_1 degiskeninin degeri: %f\n",ondalikli_1);
33.         printf("float tipinde ondalikli_1 degiskeninin degeri: %f\n",ondalikli_2);
34.
35.         printf("Bir tab\tbosluk birakma. Tek tirnak \' çift tirnak \" \n Alt satira inme. Uyari sesi \a \n\n");
36.
37.     int a=1;
38.
39.         printf("Atama operatoru kullanim sekilleri: \nint a=1; a degiskenine 1 atadi ve a artik: 1 oldu.\na+=4; veya a=a+1; ile a degiskenini kendisi ve 1 ile topladi ve a artik %d oldu.",a+=4);
40.         a%=2;
41.         printf("\na mod 2 = %d - a mod 2, a nin ikiye bolumunden kalani hesaplar",a);
42.
43.         printf("\n\na 1 iken a++ yapinca önce ekrana a yi %d basar ve bastıktan sonra a yı bir arttırır, a nın yeni degeri %d.",a++,a);
44.         printf("\n\na 2 iken ++a yapinca önce a yı %d  bir arttırır sonra ekrana a yı %d basar, a nın yeni degeri %d.",++a,a,a);
45.
46.         printf("\na-=2 yaparsak a değişkeninin kendisinden 2 çıkartı ve yeni a değeri: %d",a-=2);
47.         printf("\na+= 9-1*4 yaparsak a değişkenini kendisi ile ve işlemin sonucu ile toplar ve sonuc: %d olur cunku operatorlerin onceliğine dikkat edilmeli carpmanin vardir c de",a+= 9-1*4);
48.     return 0;
49. }
```

ÇIKTI:

- Define sabiti ile PI sayisini tuttuk: 3.141000
- integer tipinde tamSayi degiskeninin degeri: 1
- const int tipinde boyut degiskeninin degeri: 10
- Define sabiti ile toplama islemini yaptirdik (int tamSayi + const int) boyut: 11
- double tipinde ondalikli_1 degiskeninin degeri: 5.480000
- float tipinde ondalikli_1 degiskeninin degeri: 5.480000
- Bir tab bosluk birakma. Tek tirnak ' çift tirnak "
- Alt satira inme. Uyari sesi
-
- Atama operatoru kullanim sekilleri:
- int a=1; a degiskenine 1 atadi ve a artik: 1 oldu.
- a+=4; veya a=a+1; ile a degiskenini kendisi ve 1 ile topladi ve a artik 5 oldu.
- a mod 2 = 1 - a mod 2, a nin ikiye bolumunden kalani hesaplar
-
- a 1 iken a++ yapinca önce ekrana a yi 1 basar ve bastıktan sonra a yı bir arttırır, a nın yeni degeri 2.

-
- a 2 iken ++a yapinca önce a yı 3 bir arttırır sonra ekrana a yı 3 basar, a nın yeni degeri 3.
- a-=2 yaparsak a değişkeninin kendisinden 2 çıkartı ve yeni a değeri: 1
- a+= 9-1*4 yaparsak a değişkenini kendisi ile ve işlemin sonucu ile toplar ve sonuc: 6 olur cunku operatorlerin onceliğine dikkat edilmeli carpmanin vardir c de

1.4. Örnek: Kullanicidan aldginiz karakterin ascii tablosundaki degerini gösteriniz.

1_4_ornek.c

```
1. #include <stdio.h>
2. int main(void)
3. {
4.         char ascii;
5.         printf("ascii' tablosundaki degeri gosterilecek karakteri giriniz :\n");
6.         scanf("%c",&ascii);
7.         printf("%d ",ascii);
8.         return 0;
9. }
```

ÇIKTI:

```
ascii' tablosundaki degeri gosterilecek karakteri giriniz :
A
65
```

1.5. Örnek: Kullanicinin dogum yilini alarak 2020 senesinde kac yasinda oldugunu bulunuz.

1_5_ornek.c

```
1. #include <stdio.h>
2. int main(void)
3. {
4.         int sene = 2020;
5.         int dogum;
6.         printf("Dogum yilinizi giriniz :");
7.         scanf("%d",&dogum);
8.         printf("2020 senesinde %d yasindasiniz. ",2020-dogum);
9.         return 0;
10. }
```

ÇIKTI:

```
Dogum yilinizi giriniz :2000
2020 senesinde 20 yasindasiniz.
```

1.6. Örnek: Kullanicidan alinan 2 tane sayinin ortalamasini hesaplayiniz.

1_6_ornek.c

```
1.  #include <stdio.h>
2.  int main(void)
3.  {
4.          int s,s1;
5.
6.    printf("1.Sayiyi giriniz : ");
7.          scanf("%d",&s);
8.    printf("2.Sayiyi giriniz : ");
9.    scanf("%d",&s1);
10.         printf("Ortalama: %.2f",(s+s1)/2.0);
11.         return 0;
12. }
```

ÇIKTI:

```
1.Sayiyi giriniz : 30
2.Sayiyi giriniz : 20
Ortalama: 25.00
```

2. HAFTA

A. Önişlemci komutları

Önişlemci talimatları, kaynak kodunun derlenmesinden önce uygulanır. Tüm önişlemci talimatları bir # simgesi ile başlar. Önişlemci talimatları noktalı virgülle bitmez. Önişmecinin yaptığı işler macro talimatları, koşullu derleme talimatları dosya dahil etme talimatları. 1. Hafta #include ve kutuphane eklemeden kısaca bahsetmiştik.

Talimat	Tanımı
#include	Bir C Programına üsbilgi dosyasını[header file] dahil eder.
#define	İkame makrosudur. Bir ifade vasıtası ile sabit bir değeri programa yerleştirir.
#if	Koşullu ifadenin sonucuna bağlı bir kod bloğu içerir.
#else	#if'in tamamlayıcısıdır.
#elif	#else ve #if ile benzerdir.
#endif	Bu, #if, #elif gibi koşullu yönergelerin sonunu işaretler.
#undef	Tanımsız bir önişlemci makrosu.
#ifdef	Eğer bir sabit değer daha önce #define ile tanımlanmışsa true döndürür.
#ifndef	Eğer sabit değer daha önce #define ile tanımlanmamışsa true döndürür.
#pragma	Derleyiciye özel komutlar verir.
#error	Hata mesajını stderr'da yazdırır.[4]

a) #include Önişlemcisi

#include Önişlemci Bildirimleri, C Programında üstbilgi(*.h) dosyası (kütüphane)eklemek için kullanılır. Yol belirtilmemişse, geçerli dizindeki üstbilgi(*.h) dosyasını denetler. Kullanıcı tanımlı üstbilgi(*.h) dosyasını eklemek için açılı parantez<> yerine çift tırnak"" işareti kullanıyoruz.

```
#include <stdio.h>       // Standart, giris cikis kutuphanesini ekledik.
#include “benimDosyam.h” // Kullanici tanimli ust bilgi dosyaysını ekledik.
```

İlk satır Önişlemciye bu satırı stdio.h başlık dosyasının içeriğiyle değiştirmesini söyler. İkinci satır önişlemciye benimDosyam.h dosyasını geçerli dizinden almasını ve benimDosyam.h dosyasının içeriğini eklemesini söyler.

b) #define Önişlemcisi

#define programimizda bazi kisaltmalari, işimizi kolaylastiracak tanimlamalari oluşturabilmemizi sağlar ve çok esnektir.

[4] https://vdemir.github.io/ceviriler/ceviriler2/2018-01-03-Preprocessor.html

 # define'nın yapabilecekleri:

1) Sayi Bildirme

```
#define Isim_PI 3.141
// Isim_PI adında bir isim verdik ve içeriğine 3.141 pi nin değerini yazdık.
```

Böylece pi sayisina ihtiyacimiz olduğu her an Isim_PI yi çağırdığımızda bize 3.141 değerini getirecektir.

```
#define enSevdigim 11
#define boyut 100
#define ondalik 54.87
#define minimum 50
```

2) Karakterleri Bildirme

```
#define altSatir '\n'
// Isim_PI adında bir isim verdik ve içeriğine 3.141 pi nin değerini yazdık.
```

Böylece her altSatir ı çağırdığımızda \n değerini getirecektir ve geldiği yerde alt satira geçecektir.

3) Komut, Fonksiyonları Bildirme

```
#define topla(x,y) x+y
/* topla() adında bir isim verdik ve içeriğine iki tane x,y değişkenini
verdik ve x ve y yi toplaması için x+y yazdik ve biz topla(5+2) girdiğimizde
7 yi verecektir */
```

Böylece topla(x,y) ı çağırdığımızda bize x+y değerini getirecektir.

```
#define pi 3.141
#define cikar(x,y) x-y
#define kareAl(x) x*x
#define modAl(x,y) x%y
#define ucgenAlani(taban,h) (taban*h)/2
#define dikdortgenCevre(x,y,a,b) x+y+a+b
#define cemberCevre(r) 2*pi*(r)
```

c) #undef Önişlemcisi

Daha önce tanımlanmış bir makronun tanımını kaldırmak için #undef kullanırız.

```
//ÖRNEKLER: 2_2_ornek.c
#include <stdio.h>
#define sayi 33
int main()
{
        printf(“sayi’nin degeri: %d”,sayi);
        //sayiyi ekrana basacaktir

        #undef sayi
        //artik sayi tanimi yok

        printf(“sayi’nin tanimsiz degeri: %d”,sayi);
        //bu satir yuzunden program hata verecek

        return 0;
}
```

Tanimsizlaştırdığımız bir değişkeni kullanmaya calisirsak program undeclared hatasını verecektir:

```
2_2_ornek.c: In function 'main':
2_2_ornek.c: In function 'main': 2_2_ornek.c: In function 'main':
2_2_ornek.c:11:42: error: 'sayi' undeclared (first use in this function)
  printf("sayiГin tanimsiz degeri: %d",sayi);
                                          ^~~~
2_2_ornek.c:11:42: note: each undeclared identifier is reported only once for
each function it appears in
```

d) #if, #else ve #endif Koşullu Derleme Önişlemcisi

Koşullu Derleme Bildirimleri, koşullu ifadenin sonucuna dayalı bir kod bloğu eklememizi sağlar.

```
//ÖRNEKLER: 2_1_ornek.c
#include <stdio.h>
#define sayi 33
int main()
{
        #if((sayi%2)==0)
                printf("Sayi Cifttir\n");
        #else
                printf("Sayi Tektir\n");
        #endif

        return 0;
}
```

```
Cikti: Sayi Cifttir
```

B. KARŞILAŞTIRMA – KOŞULLLAR

Koşulların kontrolünde kullanılan komutlardır. Koşulların doğru olup olmamasına (sağlanıp sağlanmamasına) göre işlem akışını yönlendirir. Karar komutları dört farklı yapıda(biçim/form) olabilirler: (VATANSEVER)

i.Yarım form: Sadece koşul/koşullar doğru olduğunda (doğru ise -evet) yapılacak işlem/işlemler vardır.

ii.Tam form: Koşul/koşullar doğru ve yanlış olduğunda yapılacak işlem/işlemler vardır.

iii.Çok koşullu form: Birçok koşulun durumuna göre yapılacak işlem işlemler vardır.

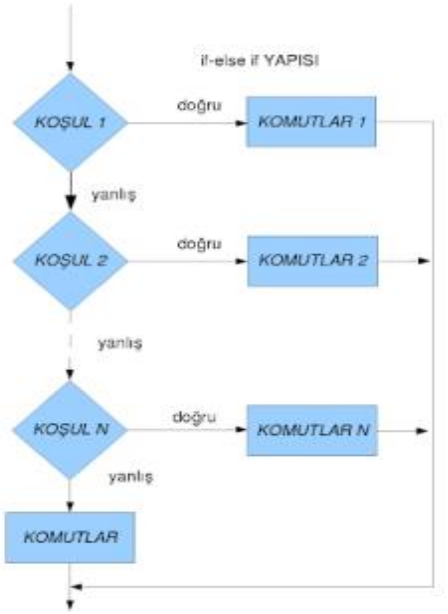
iv.Seçimli form: Seçim/kontrol değişkeninin değerine değerine gore yapılacak işlem işlemler vardır.

İşlem tek komuttan oluşuyorsa blok açmaya gerek yoktur.

a) if-else

```
if (koşul veya koşul grubu) işlem;
if (koşul veya koşul grubu) işlem-1;
else işlem-2;
```

```
if (koşul veya koşul grubu) işlem;
else if (koşul veya koşul grubu) işlem-1;
else if (koşul veya koşul grubu) işlem-2;
else if (koşul veya koşul grubu) işlem-3;
...
else işlem-44;
```




```
if(yas<18)
printf("Yasiniz ehliyet almak cok kucuk :/”);
else
printf("Yasiniz ehliyet almak icin uygun, Hayirli olsun :)...");
```

Eğer yas değiskeni 18 den kucuk ise parantezlerden sonra gelen ilk komut çalıştırılacaltır ve else den sonra gelen ilk komut a uğramadan program devam edecek if-else sonlanacaktır. Eğer kucuk olmasaydı else nin altindaki komutu çalıştırcaktı.

```
if(yas<18)
{
        printf("Yasiniz ehliyet almak cok kucuk :/”);
        printf("Uzulmeyin %d sene sonra alırsınız”,18-yas);
}
else
{
        printf("Yasiniz ehliyet almak icin uygun, Hayirli olsun :)...");
        printf(“Aman, kurallara dikkat ediniz...”);
}
```

Eğer yas değişkeni 18 den kucuk ise ilk suslu parantezli bloğa girip içindeki komutlari çalistiracaktir. Eğer kucuk olmasaydı else nin altindaki bloğun içindeki kodları çalıştıracaktı. Ve if-else sonlanacak programa devam edilecekti.

b) switch-case

Seçimli fonksiyon yapısıdır. “switch ” ten sonra belirtilen “değişken” , “case” ten sonra hangi değeri alırsa, karşılığındaki işlem yapılır. Eğer “değişken ” e karşılık gelen deper “case” lerde yoksa “default” taki islemler gerçekleşecektir.

```
switch(secim)
{
        case 1:
        case 2:
        case 3:
                printf(“Seciminiz 1 – 2 - 3”);
                break;
        case 4:
                printf(“Seciminiz 4”);
                break;
        case 5:
                printf(“Seciminiz 5”);
                break;
        default:
                printf(“Seciminiz 1-5 arasinda olmaliydi gecersiz secim!”);
                break;
}
```

Switch secmeli kontrolumuze integer tipinde `secim` adinda bir değişken koyduk ve `secim` `case` 1-2-3 ten biri ise ekrana `Seciminiz 1 – 2 – 3` ekrana basacak, eğer `case` 4 ise ekrana `Seciminiz 4` basacak, eğer `case` 5 ise ekrana `Seciminiz 5` basacak, eğer secim `case` lerde yoksa default icindeki `gecersiz secim` basacaktir.

Çalışma yapısı ise `secim` `case` lerdekinden hangi sayiyla ayni ise onun altındaki kodu `break`; gorene kadar calistiracaktir yani break koymazsak `case` lerimiz alt alta sırayla ilk sağladığı andan itibaren çalşacaktır Örneğin 1 girerse kullanici o blokta break belirtmediğimiz için altindaki `case` ifadesinide calistiracaktir. `secim` ifadesi `case` lerde yoksa `default` blogu icindeki kodlar calistirilacakti.

c) ?

Koşulun durumuna göre ilgili değeri veya işlem sonucunu belirtilen değişkene aktarır.

```
degisken = (kosul) ? komut1 : komut2;
```

Bu karar yapısında `kosul` doğru ise `komut1` den, yanlış ise `komut2` den üretilen sonuc degiskene aktarılacak.

C.DÖNGÜLER

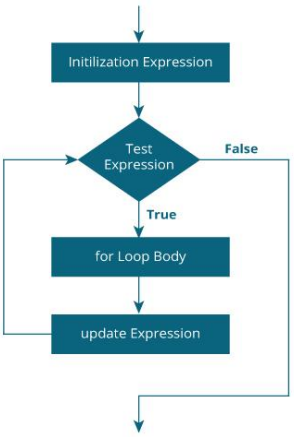
Ardışık veya tekrarlı işlemlerin yapılmasını sağlayan komuttur. Döngüler üçe ayrılır:

i.Sayıcılı Döngüler: Döngü işlemleri bir sayaca bağlı olarak gerçekleştirlir.
ii.Ön Koşullu Döngüler: Döngü içindeki işlemler koşula koşullara bağlı kalarak gerçekleştirlir ve bu koşul/koşullar döngü öncesinde kontrol edilir.
iii. Son Koşullu Dongüler: Döngü içindeki işlemler koşula/koşullara bağlı olarak gerçekleştirilir ve bu koşul/koşullar döngü sonunda kontrol edilir.

Ön koşullu ve son koşullu döngülerde döngü değişkenine döngü öncesinde başlangıç değeri verilmelidir. Döngü değişkeni döngü içerisinde arttırılmalı/azaltılmadır (değiştirilmelidir) yoksa döngü sonsuza kadar devam edecektir.

a) for

For döngüsüdne koşul sağlandığı sürece döngü bloğu işlemleri yapılır ve işlemler bitince döngü tekrar başa döner koşulu kontrol eder sağladığı surece bloktaki işlemleri gerçekleştirir ve tekrar döngü başa döner koşul sağlanmadığı zaman for bloktan çıkar.




```
for (baslangicDegeri; kosul; adim)
{
        /* ISLEMLER*/
}
```

```
for (int i=0; i<3 ; i++)
```

```
{
        printf(“\nTekar %d”,i);
}
```

Döngümüzde başlangic değeri i yi sıfırda baslattik ve i 3 ten kucuk oldugu surece calisacak ve i++ ile her adımda i yi bir artırarak devam edecektir. Ve 3 kere işlemleri tekrarlar.

```
CIKTI:
Tekrar 0
Tekrar 1
Tekrar 2
```

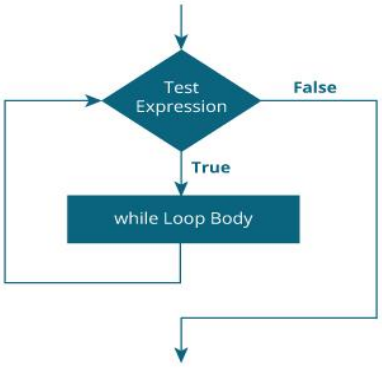
Başlangıç değerine, birden fazla baslangic deger verilebilir. Koşul kısmındada birden falza koşul yazılabilir, adım kısmına da birden falza işlem yazılabilir.

```
for (int i=0, j=10; i < 3 && j<12; i++,j++)
{
        printf(“Tekrar i=%d j=%d\n",i,j);
}
```

Döngümüzde başlangic değerine iki tane dğeişken koyduk i yi 0’dan, j yi 10’dan ve i 3 ten kucuk j 12 den kucuk oldugu surece calisacak ve i++, j++ ile her adımda i,j yi bir artırarak devam edecektir.Ve işlemleri 2 kere tekrarlayacak cunku koşulda j değerinin 10 dan 12 ye kadar sağlanması istendiği için i nin 0dan 3 kere donmeside saglanmadı.

```
CIKTI:
Tekrar i=0 j=10
Tekrar i=1 j=11
```

b) while



While ön koşullu bir döngüdür. While ile verilen koşul sağlandığı sürece döngü içindeki gerçekleştirilir ve tekrarlanır.

```
while (koşul)
{
        /*ISLEMLER*/
}
```

```
for (int i=0; i<3 ; i++)
{
        printf(“\nTekar %d”,i);
}
```

Döngümüzde başlangic değeri i yi sıfırda baslattik ve i 3 ten kucuk oldugu surece calisacak ve i++ ile her adımda i yi bir artırarak devam edecektir. Ve 3 kere işlemleri tekrarlar.

```
CIKTI:
Tekrar 0
Tekrar 1
Tekrar 2
```

```
int i = 0;
while (i < 3)
{
  printf("Tekrar %d\n",i);
}
```

Döngümüzde kontrol edeceğimiz koşul için int i =0 değişkenini oluşturduk, ve while koşulumuzda i<3 oldugu surece tekrarlanacak, ama i değerini dongu içinde değiştirmediğimiz her tekrar i<3 kosulunu sağlayacak ve sonsuz döngüye girecek.

```
CIKTI:
Tekrar 0
Tekrar 0
Tekrar 0
....
```

Bu sorunu duzeltmek için dongu içinde i değerini arttırmalıyız koşulu sağlasın.

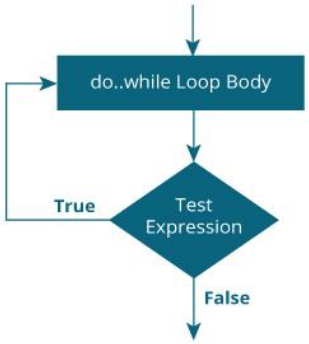
```
int i = 0;
while (i < 3)
{
  printf("Tekrar %d\n",i);
  i++;
}
```

Dongu içinde i yi arttırdığımı kosul sağlandığı için while dongusu 3 kere tekrarlanınca durdu.

```
CIKTI:
Tekrar 0
Tekrar 1
Tekrar 2
```

c) do-while

Son koşullu döngüdür. Bu döngüde once işlemler yapılır sonra koşul kontrol edilir. Ve bu sayede dongu mutlaka bir kere çalıştırılır.




```
do
{
        /*ISLEMLER*/
}
while (koşul)
```

```
int i= 0;
do
{
   printf("Tekrar %d\n",i);
}
while (i != 0);
```

Burada döngümüz kosulu sağlamadığı halde do while dan dolayı mutlaka bir kere gerçekleştirir.

```
CIKTI:
Tekrar 0
```

```
int i= 0;
do
{
   printf("Tekrar %d\n",i);
   i++;
}
while (i < 3);
```

Burada işlemler bir kere gerçekleşir ve kosul kontrol edilir kosul sağlandığından 2 kere daha tekrarlanır işlemler.

```
CIKTI:
Tekrar 0
Tekrar 1
Tekrar 2
```

Döngülerde dikkat etmemiz gerekenler:

- Döngüyü koşulu sağlanmalı,
- Döngü değeri döngüde mutlaka koşula uygun arttırılmalı azaltılmalı veya değiştirmelidir,
- Sonsuz döngüye sokmamak gerek.
- Break ve continue kullanabilirsiniz.

Break:

Break komutu var olduğu bloğu kırmaya yarar. En fazla switch-case de kullanılır. Döngülerde de kullanılır.

```
for (int i=0; i<10 ; i++)
{
        printf("\nTekar %d",i);
        if(i==3) break;
}
```

Burada döngümüzün 10 kere tekrarlanması beklenirken i=3 olduğu döngü break komutu ile kırılmış ve sonlanmıştır. Haliyle döngü 4 kere tekrarlanmıştır.

```
CIKTI:
Tekar 0
Tekar 1
Tekar 2
Tekar 3
```

Continue:

Continue komutu döngülerde yazıldığı anda o anı antlar sonraki adıma geçer.

```
for (int i=0; i<5 ; i++)
{
        printf("\nTekar %d",i);
        if(i==3) break;
}
```

Burada döngümüzün 0 dan 5 e kadar sayilari yazmasi beklenirken 3 ü yazmamıştır çünkü i=3 olduğu anda döngü continue komutu ile atlanmıştır o continue den sonraki işlemler atlanıp sonraki dongü başa döner kaldığı yerden devam eder.

```
CIKTI:
Tekar 0
Tekar 1
Tekar 2
Tekar 4
```

2. HAFTA ÖRNEKLER

2.1. Örnek: Önişlemci komutları ile ilgili genel bir örnek yapiniz.

1_1_ornek.c

```
1.  #include <stdio.h>
2.
3.  #define altagec '\n'
4.  #define tab '\t'
5.  #define unlem '!'
6.  #define pi 3.141
7.  #define sayi 32
8.  #define cemberCevre(r) 2*pi*(r)
9.  #define sayi 32
10. int main()
11. {
12.         printf("Hoooop%cassagidayim Hoooop tab%c aaaa unlem %c
    \n\n",altagec,tab,unlem );
13.
14.         printf("Yaricapi 3 olan dairenin cevresi: %f\n",cemberCevre(3));
15.
16.         #if((sayi%2)==0)
17.     printf("Sayi Cifttir\n");
18.         #else
19.     printf("Sayi Tektir\n");
20.         #endif
21.
22.         return 0;
23. }
```

ÇIKTI:

- Hoooop
- assagidayim Hoooop tab aaaa unlem !

- Yaricapi 3 olan dairenin cevresi: 18.846000
- Sayi Cifttir

2.2. Örnek: #undef in hakkinda kod yaziniz.

2_2_ornek.c (BU KOD BİLEREK HATALI BIRAKILDI)

```
1.  //BU KOD BİLEREK HATALI YAZİLDİ
2.  #include <stdio.h>
3.  #define sayi 33
4.  int main()
5.  {
6.          printf("sayi'nin degeri: %d",sayi);
7.          //sayiyi ekrana basacaktir
8.
9.          #undef sayi
10.         //artik sayi tanimi yok
11.
12.         printf("sayi'nin tanimsiz degeri: %d",sayi);
13.         return 0;
14. }
```

ÇIKTI:

- 2_2_ornek.c: In function 'main':
- 2_2_ornek.c:11:42: error: 'sayi' undeclared (first use in this function)
- printf("sayiΓin tanimsiz degeri: %d",sayi);
- ^~~~
- 2_2_ornek.c:11:42: note: each undeclared identifier is reported only once for each function it appears in

2.3. Örnek: if else ile kisinin ehliyet alip alamayacagini gosteren kod.

2_3_ornek.c

```
1.  #include <stdio.h>
2.
3.  int main()
4.  {
5.          int yas=0;
6.
7.          printf("Yasinizi giriniz: ");
8.          scanf("%d",&yas);
9.          if(yas<18) printf("Yasiniz ehliyet almak cok kucuk :/. Uzulmeyin %d
    sene sonra alabilirsiniz :). ",18-yas);
10.         else printf("Yasiniz ehliyet almak icin uygun, Hayirli olsun :)...");
11.
12.         return 0;
13. }
```

ÇIKTI:

```
Yasinizi giriniz: 14
```

Yasiniz ehliyet almak cok kucuk :/. Uzulmeyin 4 sene sonra alabilirsiniz :).

2.4. Örnek: Switch-case yapısına bir ornek veriniz.

2_4_ornek.c

```
1.  #include <stdio.h>
2.
3.  int main()
4.  {
5.          int secim=0;
6.          printf("1-5 arasinda bir secim yapiniz: ");
7.          scanf("%d",&secim);
8.
9.          switch(secim)
10.         {
11.                 case 1:
12.                 case 2:
13.                 case 3:
14.                         printf("Seciminiz 1, 2, 3");
15.                         break;
16.                 case 4:
17.                         printf("Seciminiz 4");
18.                         break;
19.                 case 5:
20.                         printf("Seciminiz 5");
21.                         break;
22.                 default:
23.                 printf("Seciminiz 1-5 arasinda olmaliydi gecersiz secim!");
24.                 break;
25.         }
26.
27.         return 0;
28. }
```

ÇIKTI:

```
1-5 arasinda bir secim yapiniz: 1
Seciminiz 1, 2, 3
```

2.5. Örnek: Kullancidan 1-12 arasi tam sayi alip o sayinin ay olarak karsiligini yazan programı yaziniz.

2_5_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int ay=0;
5.          printf("Ay'in sayisal degerini yaziniz(1-12):");
6.          scanf("%d",&ay);
7.          switch(ay)
8.          {
9.                  case 1:
10.                         printf("\nOcak");
11.                         break;
12.                 case 2:
13.                         printf("\nSubat");
14.                         break;
15.                 case 3:
16.                         printf("\nMart");
17.                         break;
18.                 case 4:
19.                         printf("\nNisan");
20.                         break;
21.                 case 5:
22.                         printf("\nMayıs");
23.                         break;
24.                 case 6:
25.                         printf("\nHaziran");
26.                         break;
27.                 case 7:
28.                         printf("\nTemmuz");
29.                         break;
30.                 case 8:
31.                         printf("\nAgustos");
32.                         break;
33.                 case 9:
34.                         printf("\nEylul");
35.                         break;
36.                 case 10:
37.                         printf("\nEkim");
38.                         break;
39.                 case 11:
40.                         printf("\nKasim");
```

```
41.                         break;
42.                 case 12:
43.                         printf("\nAraklik");
44.                         break;
45.                 default:
46.                         printf("\nGecersiz giris 1-12 arasi sayi giriniz.");
47.                         break;
48.         }
49.         return 0;
50. }
```

ÇIKTI:

```
Ay'in sayisal degerini yaziniz:11
Kasim
```

2.6. Örnek: Kullanicidan bir ayin sayisal degerini aliniz ve o ayin hangi mevisme ait oldugunu gosteren kodu switch-case yapisi ile yaziniz

2_6_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int ay=0;
5.          printf("Ay'in sayisal degerini yaziniz(1-12):");
6.          scanf("%d",&ay);
7.          switch(ay)
8.          {
9.                  case 12:
10.                 case 1:
11.                 case 2:
12.                         printf("\nKis mevisimindesiniz. Aman kalin giyininde
    usutmeyin :)");
13.                         break;
14.                 case 3:
15.                 case 4:
16.                 case 5:
17.                         printf("\nIlkbahar mevisimindesiniz. Kelebekleeer :)");
18.                         break;
19.                 case 6:
20.                 case 7:
21.                 case 8:
```

```
22.                         printf("\nYaz mevisimindesiniz. Bahamalarda bir tatili
    hakettin hadi gene iyisin :)");
23.                         break;
24.
25.                 case 9:
26.                 case 10:
27.                 case 11:
28.                         printf("\nSonbahar mevisimindesiniz. Uuuu sogumaya
    mi basladi ne?");
29.                         break;
30.                 default:
31.                         printf("\nGecersiz giris 1-12 arasi sayi giriniz.");
32.                         break;
33.         }
34.         return 0;
35. }
```

ÇIKTI:

```
Ay'in sayisal degerini yaziniz(1-12):7
Yaz mevisimindesiniz. Bahamalarda bir tatili hakettin hadi gene iyisin :)
```

2.7. Örnek: Kullanicidan yasini alip koronodan dolayi sokaga cikabilir mi cikamazmi kontolunu yaptırnız.(1-20 cikabilir, 65+ cikamaz)

2_7_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int yas =0;
5.          printf("Yasinizi giriniz: ");
6.          scanf("%d",&yas);
7.          if(yas>0)
8.          {
9.                  if(yas<21) printf("Malesef yasiniz cok kucuk sokaga
    cikamazsiniz. Acil durumlarda izin alabilirsiniz :)");
10.                 else if(yas<65) printf("Sizin icin sokaga cikma yasagi yok. Ama
    cok dikkatli olmalisiniz...");
11.                 else if(yas>64) printf("Malesef disari cikamazsiniz. Sagliginiz ve
    sevdikleriniz icin evde kalin.");
12.         }
13.         else printf("Yasinizi dogru giriniz.");
14.         return 0;
15. }
```

ÇIKTI:

```
Yasinizi giriniz: 56
```

```
Sizin icin sokaga cikma yasagi yok. Ama cok dikkatli olmalisiniz...
```

2.8. Örnek: Alinan sayinin mutlagini hesaplayiniz if kosulu ile.

2_8_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int sayi =0,mutlak=0;
5.          printf("Sayiyi giriniz: ");
6.          scanf("%d",&sayi);
7.
8.          mutlak=sayi;
9.          if(sayi<0) mutlak=sayi*(-1);
10.
11.         printf("%d sayisinin mutlagi: %d",sayi,mutlak);
12.
13.         return 0;
14. }
```

ÇIKTI:

```
Sayiyi giriniz: -54
-54 sayisinin mutlagi: 54
```

2.9. Örnek: Alinan sayinin mutlagini hesaplayiniz ? kosulu ile.

2_9_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int sayi =0;
5.          printf("Sayiyi giriniz: ");
6.          scanf("%d",&sayi);
7.
8.          sayi = (sayi<0) ? sayi*(-1) : sayi;
9.
10.         printf("Sayinin mutlagi: %d",sayi);
11.
12.         return 0;
13. }
```

ÇIKTI:

```
Sayiyi giriniz: -98
Sayinin mutlagi: 98
```

2.10. **Örnek:** Örnek 2.5 teki switch-case kodunu else if koşulu ile yapiniz(ayin ismini yazdirma):

2_10_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int ay=0;
5.          printf("Ay'in sayisal degerini yaziniz(1-12):");
6.          scanf("%d",&ay);
7.          if(ay==1)       printf("\nOcak");
8.          else if(ay==2)  printf("\nSubat");
9.          else if(ay==3)  printf("\nMart");
10.         else if(ay==4)  printf("\nNisan");
11.         else if(ay==5)  printf("\nMayıs");
12.         else if(ay==6)  printf("\nHaziran");
13.         else if(ay==7)  printf("\nTemmuz");
14.         else if(ay==8)  printf("\nAgustos");
15.         else if(ay==9)  printf("\nEylul");
16.         else if(ay==10)         printf("\nEkim");
17.         else if(ay==11)         printf("\nKasim");
18.         else if(ay==12)         printf("\nAraklik");
19.         else    printf("\nGecersiz giris 1-12 arasi sayi giriniz.");
20.         return 0;
21. }
```

ÇIKTI:

```
Ay'in sayisal degerini yaziniz(1-12):8
Agustos
```

2.11. **Örnek:** Kullanicidan alinan iki sayidan hangisinin kucuk, buyuk olduğunu hesaplayınız.

2_11_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int s1 =0;
5.          int s2 =0;
6.          printf("Arada bir bosluk birakarak iki sayiyi giriniz: ");
```

```
7.          scanf("%d %d",&s1,&s2);
8.          if(s1>s2)       printf("\n%d > %d",s1,s2);
9.          else if(s2>s1) printf("%d > %d",s2,s1);
10.         else printf("%d = %d",s1,s2);
11.
12.         return 0;
13. }
```

ÇIKTI:

```
Arada bir bosluk birakarak iki sayiyi giriniz: -8 47
47 > -8
```

2.12. Örnek: Switch-case ile kare alma, mod alma, toplama, cikarma, bolme, carpma içeren secim menulu bir hesap makinesi yapiniz.

2_12_ornek.c

```
1.  #include <stdio.h>
2.  int main(void)
3.  {
4.          float x,y;
5.          int secim=0;
6.          printf("MENU\n[1]Toplama\n[2]Cikarma\n[3]Carpma\n[4]Bolme\n[5]Ka
    re Alma\n[6]Mod Alma\nSeciminiz: ");
7.          scanf("%d",&secim);
8.          switch(secim)
9.          {
10.                 case 1:
11.                         printf("1.Sayiyi Giriniz: ");
12.                         scanf("%f",&x);
13.                         printf("2.Sayiyi Giriniz: ");
14.                         scanf("%f",&y);
15.                         printf("Sonuc: %.2f",x+y);
16.                         break;
17.                 case 2:
18.                         printf("1.Sayiyi Giriniz: ");
19.                         scanf("%f",&x);
20.                         printf("2.Sayiyi Giriniz: ");
21.                         scanf("%f",&y);
22.                         printf("Sonuc: %.2f",x-y);
23.                         break;
24.                 case 3:
```

```
25.                         printf("1.Sayiyi Giriniz: ");
26.                         scanf("%f",&x);
27.                         printf("2.Sayiyi Giriniz: ");
28.                         scanf("%f",&y);
29.                         printf("Sonuc: %.2f",x*y);
30.                         break;
31.                 case 4:
32.                         printf("1.Sayiyi Giriniz: ");
33.                         scanf("%f",&x);
34.                         printf("2.Sayiyi Giriniz: ");
35.                         scanf("%f",&y);
36.                         printf("Sonuc: %.2f",x/y);
37.                         break;
38.                 case 5:
39.                         printf("Karesini almak istediginiz sayiyi giriniz: ");
40.                         scanf("%f",&y);
41.                         printf("Sonuc: %.2f",y*y);
42.                         break;
43.                 case 6:
44.                         printf("Modunu hesaplamak istediginiz sayiyi giriniz: ");
45.                         scanf("%f",&x);
46.                         printf("Modunu yaziniz: ");
47.                         scanf("%f",&y);
48.                         printf("Sonuc: %d",(int)x % (int)y);
49.                         break;
50.                 default:
51.                         printf("Yanlis secim, 1-6 arasi secim yapiniz.");
52.         }
53.         return 0;
54. }
```

ÇIKTI:

```
Seciminiz: 6
Modunu hesaplamak istediginiz sayiyi giriniz: 8
Modunu yaziniz: 3
Sonuc: 2
```

2.13. Örnek: Kullanıcıdan aldığınız 3 basamaklı sayının basamaklarındaki sayıları bulunuz.

2_13_ornek.c

```
1. int main(void)
2. {
```

```
3.          int sayi=0;
4.          int b1,b2,b3;
5.          printf("Uc basamakli sayiyi giriniz: ");
6.          scanf("%d",&sayi);
7.          if(sayi>99 && sayi<1000 || sayi<-99 && sayi>-1000)
8.          {
9.                  b1=sayi%10;
10.                 sayi/=10;
11.                 b2=sayi%10;
12.                 sayi/=10;
13.                 b3=sayi;
14.
15.                 printf("\nBirler basamagi: %d\nOnlar basamagi: %d\nYuzler
    basamagi: %d",b1,b2,b3);
16.
17.         }else printf("Uc basamakli sayi girmeniz gerekiyordu!");
18.
19.         return 0;
20. }
```

ÇIKTI:

```
Uc basamakli sayiyi giriniz: 546
Birler basamagi: 6
Onlar basamagi: 4
Yuzler basamagi: 5
```

2.14. Örnek: if-else koşulunun koşulunu dışardan alarak farklı bir yapıda oluşturunuz.

2_14_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int a=10,b=27,c=34,d=10;
5.          int kosul1= (a==d && b>c); //kosullar dogru olmadigi icin 0 atanir
6.          int kosul2= (a!=d || c>b); //kosullarin sonucu true oldugu icin  1 atanir
7.          int kosul3= kosul1==kosul2; //1==0 olmadigindan 0 atacanak
8.          if(kosul3) printf("kosul true 1");
9.          else printf("kosul false 0");
10.         return 0;
11. }
```

ÇIKTI:

```
kosul false 0
```

2.15. **Örnek:** Kullanıcıdan aldığınız sayi kadar ekrana merhaba yazdiriniz.

2_15_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int s=0;
5.          printf("Sayiyi giriniz: ");
6.          scanf("%d",&s);
7.          int i=0;
8.          for(i;i<s;i++)
9.          {
10.                 printf("Merhaba! ");
11.         }
12.         return 0;
13. }
```

ÇIKTI:

```
Sayiyi giriniz: 4
Merhaba! Merhaba! Merhaba! Merhaba!
```

2.16. **Örnek:** Kullanıcıdan aldığınız sayinin tek mi çift mi olduğunu bulunuz.

2_16_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int s=0;
5.          printf("Bir sayi giriniz: ");
6.          scanf("%d",&s);
7.          if(s%2==0) printf("Cift");
8.          else printf("Tek");
9.      return 0;
10. }
```

ÇIKTI:

```
Bir sayi giriniz: 8
Cift
```

2.17. Örnek: Kullanıcıdan aldığınız iki sayı arasında olan tek, çift, asal olan sayıları bulup ekrana yazdırınız ve aralıktaki sayilarin bolenlerini de yaziniz. Ve ikş sayi arasindaki sayilarin toplamini belirtiniz.

2_17_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.          int s=0,s2=0,i=0,j=0,bolenSayisi=0,buyuk=0,kucuk=0,toplam=0;
5.          printf("Ilk sayiyi giriniz: ");
6.          scanf("%d",&s);
7.          printf("Ikinci sayiyi giriniz: ");
8.          scanf("%d",&s2);
9.
10.         buyuk = (s>s2) ? s:s2; //buyuk sayiyi bulduk
11.         kucuk = (s<s2) ? s:s2; //kucuk sayiyi bulduk
12.         i=kucuk+1;
13.         //i dongude hep bir artilicak ve i buyuk sayiya esit oluncaya kadar
    devam edecek boylecea iki sayi arasindaki sayilari buluyoruz.
14.         while(i!=buyuk)
15.         {
16.                 //cift-tek olup olmadigini tespit ediyoruz
17.                 if(i%2==0) printf("\n%d Cifttir. Bolenleri: 1,",i);
18.                 else printf("\n%d Tekdir. Bolenleri: 1,",i);
19.
20.                 //bolen sayisini tutuyoruz ki kendisi ve birden baska boleni va
    mi anlayalim ve boylece asal olup olmadigini tespit edelim.
21.                 bolenSayisi=0;
22.                 for(j=2;j<=i;j++)//j'yi 2 den baslatarak elimizdeki sayiya kadar
    olan sayilari tespi ediyoruz
23.         {
24.                 if(i%j==0) // ve tespit ettigimiz sayilarin asal olup olmadigini
    bulmak istedigimiz sayiya tam bolunup bolunmedigini sorguluyoruz
25.                 {
26.                         //tam bolerse ekrana boleni ni yazdiriyoruz ve sayaci da
    arttirioruz
27.                         printf(" %d,",j);
28.                         bolenSayisi++;
29.                 }
30.     }
```

```
31.         if(bolenSayisi==1) printf(" - Asaldir");//bolen sayimizi kontrol ediyoruz
    eger kend 1 den farklı ise kendisinde baska da boleni var demektir
32.         else printf(" - Asal degildir.");
33.         toplam+=i; //toplami bulmak icin toplam degiskenine her dongude
    araliktaki sayiyila topluyoruz
34.                     i++;
35.           }
36.           printf("\n%d-%d Arasi sayilarin toplami: %d",kucuk,buyuk,toplam);
37.
38.    return 0;
39. }
```

ÇIKTI:

```
Ilk sayiyi giriniz: 6
Ikinci sayiyi giriniz: 15
7 Tekdir. Bolenleri: 1, 7, - Asaldir
8 Cifttir. Bolenleri: 1, 2, 4, 8, - Asal degildir.
9 Tekdir. Bolenleri: 1, 3, 9, - Asal degildir.
10 Cifttir. Bolenleri: 1, 2, 5, 10, - Asal degildir.
11 Tekdir. Bolenleri: 1, 11, - Asaldir
12 Cifttir. Bolenleri: 1, 2, 3, 4, 6, 12, - Asal degildir.
13 Tekdir. Bolenleri: 1, 13, - Asaldir
14 Cifttir. Bolenleri: 1, 2, 7, 14, - Asal degildir.
6-15 Arasi sayilarin toplami: 84
```

2.18. Örnek: Kullancıdan aldığınız sayinin faktoriyelini DONGU ile hesaplayiniz.

2_18_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int s=0,i=0,carpim=1;
5.    printf("Faktoriyelini hesaplamak istediginiz sayiyi girniz:");
6.    scanf("%d",&s);
7.    printf("%d! = ",s);
8.    for(i=0;s!=i;s--)
9.    {
10.     carpim*=s;
11.   }
12.   printf("%d",carpim);
13.   return 0;
```

```
14. }
```

ÇIKTI:

```
Faktoriyelini hesaplamak istediginiz sayiyi girniz:5
5! = 120
```

2.19. Örnek: İc ice donguler ile 1x9 carpim tablosunu yazdiriniz.

2_19_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int j=0,i=0,carpim=1;
5.
6.    for(i=1;i<10;i++)
7.    {
8.      for(j=1;j<10;j++)
9.        printf("\t%d",i*j);
10.
11.    printf("\n");
12.  }
13.
14.  return 0;
15. }
```

ÇIKTI:

```
1   2   3   4   5   6   7   8   9
2   4   6   8   10  12  14  16  18
3   6   9   12  15  18  21  24  27
4   8   12  16  20  24  28  32  36
5   10  15  20  25  30  35  40  45
6   12  18  24  30  36  42  48  54
7   14  21  28  35  42  49  56  63
8   16  24  32  40  48  56  64  72
9   18  27  36  45  54  63  72  81
```

2.20. Örnek: ic ice donguler ile kullanicidan tek sayi alıp o sayi adedince satır içeren bir baklaba dilimi yazdırınız

2_20_ornek.c

```
1.  #include <stdio.h>
2.  int main (void)
3.  {
4.          int satir=0,girilensayi=0,bosluk=0,yildiz=0,i=0;
5.          int olmasigerekenyild=1;
6.          int y =olmasigerekenyild;
7.          printf("\nTek bir sayi giriniz: ");
8.          scanf("%d",&girilensayi);
9.          printf("\n");
10.         //Kullanici tek bir sayi giresiye kadar 9999 kere hak taniyip tek sayi
    girmesini istioyrum
11.         for (i=0; i<9999;i++)
12.         {
13.                 if(girilensayi%2==0)
14.                 {
15.
16.                         printf("\nTek bir sayi giriniz: ");
17.                         scanf("%d",&girilensayi);
18.                         printf("\n");
19.
20.         }else
21.         {
22.                 //Kullanici tek sayi girince donguden cikariyorum
23.                 i=10000;
24.
25.                 //Satir kismi icin en uste bu for u olusturdum.
26.                 //Artik yazilan sayi kadar altalta satir olusturlucak
27.                 for(satir=1;satir<=girilensayi;satir++)
28.                 {
29.                         //olusturacagim bosluk sayisini bulan bir formul yaptim
    ve atadim
30.                         int olmasigerekenbos = ((int)girilensayi/2+1)-satir;
31.                         //bir formul yaptim
32.                         //simdi buraya kadar formul pozitif cikiyordu ama en
    orta satirdan sonra formul negatif cikiyor bunun icin isaretinin tersini aliyorum
33.                         if (olmasigerekenbos<0)
34.                            olmasigerekenbos=-olmasigerekenbos;
35.                         //Artik bulunulan satira formul ile buldugum eklemem
    gereken bosluklari ekleyiyorum
36.                         for(bosluk=1;bosluk<=olmasigerekenbos;bosluk++)
    printf(" ");
37.                         //Bosluktan sonra gelecek olan yildizlari ilk basta 1i
    atadigim yildiz sayisinj belirtren 'y' degiskeni kadar yildiz ekliyecwk
38.
39.                         for(yildiz =1;yildiz<=y;yildiz++) printf("*");
```

```
40.                             //bu sefer en orta satira kadar yildizlari arttirarak, orta
    satiri gecince yildiz lari azaltarak devam ettirmek soyle bir if yaptim
41.                             if (satir>(int)girilensayi/2)
42.                             y= y-2;
43.                             else y = y+2;
44.                             //buraya kadar bulunulan satirda islem yaptik simdi
    islemler bitti bir alt satira geciriyorum
45.                             printf("\n");
46.                             }
47.                     }
48.             }
49.     return 0;
50. }
```

ÇIKTI:

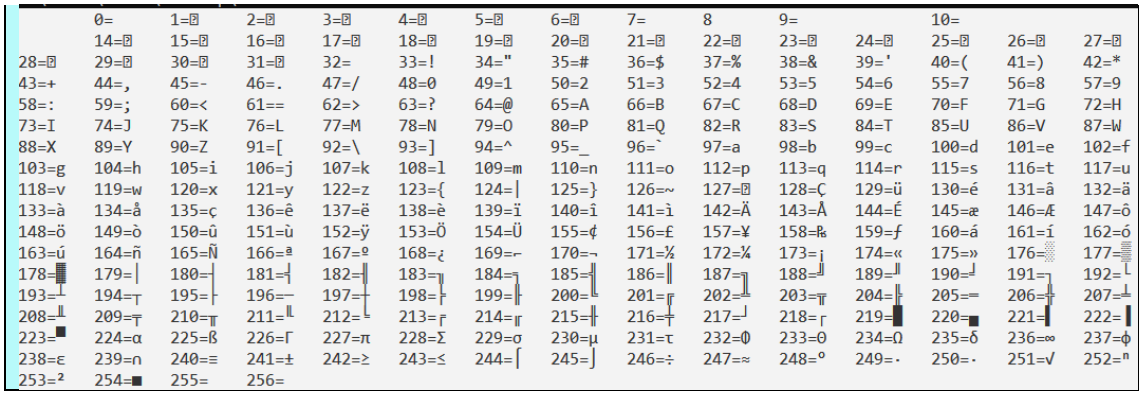
```
Tek bir sayi giriniz: 8
Tek bir sayi giriniz: 6
Tek bir sayi giriniz: 5
  *
 ***
*****
 ***
  *
```

2.21. Örnek: Kullanıcıya bütün ascii tablosundaki değerleri ve karşiliklarini gösteriniz.

2_21_ornek.c

```
1.  #include <stdio.h>
2.  int main(void)
3.  {
4.      for(int i=0; i<257;i++)
5.      printf("\t%d=%c",i,i);
6.      return 0;
7.  }
```

ÇIKTI:



	0=	1=☺	2=☻	3=♥	4=♦	5=♣	6=♠	7=	8	9=		10=		
	14=♫	15=☼	16=►	17=◄	18=↕	19=‼	20=¶	21=§	22=▬	23=↨	24=↑	25=↓	26=→	27=←
28=∟	29=↔	30=▲	31=▼	32=	33=!	34="	35=#	36=$	37=%	38=&	39='	40=(	41=)	42=*
43=+	44=,	45=-	46=.	47=/	48=0	49=1	50=2	51=3	52=4	53=5	54=6	55=7	56=8	57=9
58=:	59=;	60=<	61==	62=>	63=?	64=@	65=A	66=B	67=C	68=D	69=E	70=F	71=G	72=H
73=I	74=J	75=K	76=L	77=M	78=N	79=O	80=P	81=Q	82=R	83=S	84=T	85=U	86=V	87=W
88=X	89=Y	90=Z	91=[	92=\	93=]	94=^	95=_	96=`	97=a	98=b	99=c	100=d	101=e	102=f
103=g	104=h	105=i	106=j	107=k	108=l	109=m	110=n	111=o	112=p	113=q	114=r	115=s	116=t	117=u
118=v	119=w	120=x	121=y	122=z	123={	124=|	125=}	126=~	127=⌂	128=Ç	129=ü	130=é	131=â	132=ä
133=à	134=å	135=ç	136=ê	137=ë	138=è	139=ï	140=î	141=ì	142=Ä	143=Å	144=É	145=æ	146=Æ	147=ô
148=ö	149=ò	150=û	151=ù	152=ÿ	153=Ö	154=Ü	155=¢	156=£	157=¥	158=₧	159=ƒ	160=á	161=í	162=ó
163=ú	164=ñ	165=Ñ	166=ª	167=º	168=¿	169=⌐	170=¬	171=½	172=¼	173=¡	174=«	175=»	176=░	177=▒
178=▓	179=│	180=┤	181=╡	182=╢	183=╖	184=╕	185=╣	186=║	187=╗	188=╝	189=╜	190=╛	191=┐	192=└
193=┴	194=┬	195=├	196=─	197=┼	198=╞	199=╟	200=╚	201=╔	202=╩	203=╦	204=╠	205=═	206=╬	207=╧
208=╨	209=╤	210=╥	211=╙	212=╘	213=╒	214=╓	215=╫	216=╪	217=┘	218=┌	219=█	220=▄	221=▌	222=▐
223=▀	224=α	225=ß	226=Γ	227=π	228=Σ	229=σ	230=µ	231=τ	232=Φ	233=Θ	234=Ω	235=δ	236=∞	237=φ
238=ε	239=∩	240=≡	241=±	242=≥	243=≤	244=⌠	245=⌡	246=÷	247=≈	248=°	249=∙	250=·	251=√	252=ⁿ
253=²	254=■	255=	256=											

2.22. Örnek: Kullanıcıdan aldığınız n adet sayının ortalmasını hesaplayınız.

2_22_ornek.c

```
1.  #include <stdio.h>
2.
3.  int main(void)
4.  {
5.          int s,toplam=0,i,n;
6.
7.          printf("Kac tane sayinin ortalamasini hesaplamak istersiniz: ");
8.          scanf("%d",&n);
9.
10.         for (i = 0; i < n; ++i)
11.         {
12.                 printf("%d. Sayiyi giriniz: ",i+1);
13.                 scanf("%d",&s);
14.                 toplam+=s;
15.         }
16.         printf("Toplam: %d Adet: %d Ortalama:
    %f",toplam,i,((float)toplam/(float)i));
17.
18.
19.         return 0;
20. }
```

ÇIKTI:

```
Kac tane sayinin ortalamasini hesaplamak istersiniz: 2
```

```
1. Sayiyi giriniz: 6
2. Sayiyi giriniz: 7
Toplam: 13 Adet: 2 Ortalama: 6.500000
```

2.23. Örnek: Kullanıcıdan aldığınız 16bitli 2 lik tabandaki sayı sistemini 10luk tabandaki sayı sistemine ceviriniz.

2_23_ornek.c

```
1.  //2lik sayi sistemini 10 luga cevirme
2.  #include <stdio.h>
3.  int main(void)
4.  {
5.          int i=0,n=16,toplam=0,two=1,temp=-1;printf("SAYI SISTEMLERI 10
    TABANINA DONUSTURME\n\tORNEK:\n\n  BIT:
    1.2.3.4.5.6.7.8.9.10.11.12.13.14.15.16.\n\t^ ^ ^ ^ ^ ^ ^ ^ ^  ^  ^  ^  ^  ^
    ^\n\t| | | | | | | | | | | | | | | |\n\n\t0 1 0 1 1 1 0 0 1  0  0  1  0  0  0
    1\n\n\n\nSagdan sola dogru alt alta yaziniz.\n");
6.          n=16;
7.          two=32768;
8.          for(i=0;i<n;i++)
9.          {
10.                 while(temp!=0 && temp!=1)
11.                 {
12.                         printf("\n\t%d.Bit icin 0 veya 1 giriniz: ",i+1);
13.                         scanf("%d",&temp);
14.                 }
15.                 toplam=toplam + (two*temp);
16.                 two=two/2;
17.                 temp=-1;
18.         }
19.         printf("\n10'luk: %d",toplam);
20.         return 0;
21. }
```

ÇIKTI:

```
SAYI SISTEMLERI 10 TABANINA DONUSTURME
      ORNEK:

 BIT:1.  2. 3. 4.5.6.7.8.9.10.11.12. 13. 14. 15.16.
     ^ ^ ^ ^ ^ ^ ^ ^  ^  ^  ^  ^  ^  ^  ^
     |  | | |  | | | |  |  | |  |  | |  | |

     0  1  0  1  1  1  0  0  1  0  0  1  0  0  0  1
```

```
Sagdan sola dogru alt alta yaziniz.
        1.Bit icin 0 veya 1 giriniz: 0
        2.Bit icin 0 veya 1 giriniz: 0
        3.Bit icin 0 veya 1 giriniz: 0
        4.Bit icin 0 veya 1 giriniz: 0
        5.Bit icin 0 veya 1 giriniz: 0
        6.Bit icin 0 veya 1 giriniz: 0
        7.Bit icin 0 veya 1 giriniz: 0
        8.Bit icin 0 veya 1 giriniz: 0
        9.Bit icin 0 veya 1 giriniz: 0
        10.Bit icin 0 veya 1 giriniz: 0
        11.Bit icin 0 veya 1 giriniz: 0
        12.Bit icin 0 veya 1 giriniz: 0
        13.Bit icin 0 veya 1 giriniz: 0
        14.Bit icin 0 veya 1 giriniz: 0
        15.Bit icin 0 veya 1 giriniz: 0
        16.Bit icin 0 veya 1 giriniz: 1
10'luk: 1
```

2.24. **Örnek:** Kullanıcıdan aldığınız bir sayının basamklarını tespit ediniz. Ve sayıın tersten halini bulun.

2_24_ornek.c

```
1.  //Sayinin basmakalarini tespit edip tersten yazdirma
2.  #include <stdio.h>
3.  int main()
4.  {
5.      int i=0,s=0, basamak=0,tersSayi = 0;
6.      printf("Bir tam sayi giriniz: ");
7.      scanf("%d", &s);
8.      while (s != 0)
9.      {
10.         i++;
11.         basamak = s % 10;
12.         printf("\n%d.Basamak: %d",i,basamak);
13.         tersSayi = tersSayi * 10 + basamak;
14.         s /= 10;
15.     }
16.     printf("\nTers sayi = %d", tersSayi);
17.     return 0;
18. }
```

ÇIKTI:

```
Bir tam sayi giriniz: 156
```

```
1.Basamak: 6
2.Basamak: 5
3.Basamak: 1
Ters sayi = 651
```

3. HAFTA

A. DİZİLER

Genel olarak bilgisayarda işlenen verilerin sayısı fazladır. Çok sayıda verinin bilgisayara girilmesi, hızlı ve doğru şekilde işlenebilmesi uygun şekilde saklanabilmesi, için belirili düzende olması gerekmektedir. Belirli bir düzende olan işlenmesi saklanması daha kolaydır.

Bu nedenle biligisayar programlarında; çoklu verileri işlemek için "dizi" sıralı ardışık veri alanları kullanılır. Programda dizi ismi ile tanımlanan bu veri alanları, dizi ismi yanındaki indis numarasıyla çağırılıp işlenirler.

Diziler üç başlıkta incelenebilirler:

Bir boyutlu diziler: Sadece tek satır veya sutundan oluşan yani tek sıradan oluşan veri alanıdır.

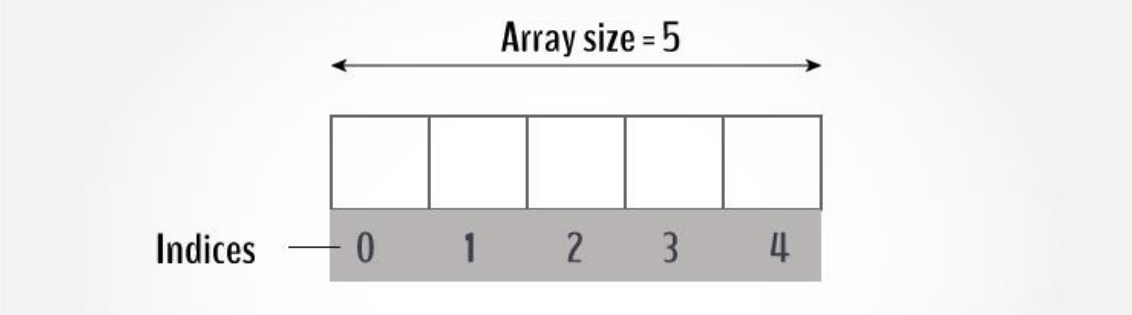
İki boyutlu diziler: "Matris" olarakta adlandırılan ve satır ile sütündan oluşan (diziler dizisi) veri alanıdır.

Çok boyutlu diziler: Üç veya daha fazla boyutlu veri alanlarıdır. Örneğin "uzay", genişlik-uzunluk-yüksekliken oluşan üç boyutlu veri alanıdır.

İster bir ister iki boyutlu dizi olsun, veriler belleğe ardışık olarak yerleştirilir.Ayrıca alfasayısal verilerde dizi şeklindedir ve kelimenin her bir karakteri bir dizi elemanı gibi hareket etmektedir.

```
veriTipi ismi[elemanSayisi];
```

```
int dizi[10];
```



B. DİZİ TANIMAMLAMA

Dizilerin sadece boyutlarını belirterek tanımlayabiliriz.

```
int dizi[5];
float dizi2[5];
char dizi3[5];
```

dizi[0]	dizi [1]	dizi [2]	dizi [3]	dizi [4]	dizi[5]
					\n

Dizilerin sadece boyutlarını ve elemanlarına da belirterek tanımlayabiliriz.

```
int dizi[3] = {5,81,9};
```

dizi [0]	dizi [1]	dizi [2]	dizi [3]
5	81	9	\n

Dizilerin boyutunu belirtmeden elemanları belirterek:

```
int dizi[] = {5,81,9,20,30,56};
```

dizi [0]	dizi [1]	dizi [2]	dizi [3]	dizi [4]	dizi[5]	Dizi[6]
5	81	9	20	30	56	\n

Dizinin elamanlarını sonradan değiştirme:

```
int dizi[] = {5,81,9};
printf("0.Indis: %d 1.Indis: %d 2.Indis: %d",dizi[0], dizi[1], dizi[2]);
dizi[0] = 11;
dizi[1] = 4;
dizi[2] = 52;
printf("0.Indis: %d 1.Indis: %d 2.Indis: %d",dizi[0], dizi[1], dizi[2]);
```

dizi [0]	dizi [1]	dizi [2]	dizi [3]
11	4	52	\n

```
ÇIKTI:
0.Indis: 5  1.Indis: 81 2.Indis: 9
0.Indis: 11 1.Indis: 4  2.Indis: 52
```

C. DİZİLERİ EKRANA BASTIRMA

Dizilerin elemanlarını ekrana bastırma:

```
int dizi[] = {5,81,9};
printf("0.Indis: %d 1.Indis: %d 2.Indis: %d",dizi[0], dizi[1], dizi[2]);
```

```
ÇIKTI: 0.Indis: 5 1.Indis: 81 2.Indis: 9
```

Dizi Boyutu Belirterek - Bütün dizi elmanlarını ekrana yazdırma:

```
int dizi[3] = {5,81,9};
for (int i = 0; i < 3; i++)
        printf("\ndizi[%d] = %d", i, dizi[i]);
```

```
ÇIKTI:
dizi[0] = 5
dizi[1] = 81
dizi[2] = 9
```

Dizinin \0 null geçersiz son indisini arayarak - Bütün dizi elmanlarını ekrana yazdırma:

```
int dizi[3] = {5,81,9};
for (int i = 0; dizi[i] != '\0'; i++)
        printf("\ndizi[%d] = %d", i, dizi[i]);
```

```
ÇIKTI:
dizi[0] = 5
dizi[1] = 81
dizi[2] = 9
```

D. DİZİYE KULLANICIDAN ELEMAN GİRDİRME

Diziye kullanıcıdan veri almak için scanf ile yapabiliriz. Hangi indise eleman eklemek istiyorsak ona gore **scanf("%d",&dizi[indis])** şeklinde ekleyebiliriz. Bütün diziye ise bu şekilde yapabiliriz.

```
int dizi[5];
for (int i = 0; i<5 ; i++)
{
        printf("\ndizi[%d]'ı giriniz:", i);
        scanf("%d",&dizi[i]);
}
```

E. İKİ BOYUTLU DİZİLER

Tablo gibi olan iki boyutlu diziler (dizileirn yatay [satır] ve düşey doğrultuda [sütun] olarak oluşturulması) 'matris' olarak isimlendirilir.

```
int dizi[3][2]; //İKİ BOYUTLU
```

Örneğin oluşturduğumuz dizinin tablosu:

	SÜTÜN 1	SÜTUN 2
SATIR 1	dizi[0][0]	dizi[0][1]
SATIR 2	dizi[1][0]	dizi[1][1]
SATIR 3	dizi[2][0]	dizi[2][1]

```
int dizi[3][2] = { {97,58}, {92,100}, {56,78}};
```

Mesela bir örneğimi inceleyelim:

	VİZE	FİNAL
MATEMATİK	97	58
FİZİK	92	100
PROGRAMLAMA	56	78

```
//örneğin dersler dizisinin satırları derslerimizi ifade etsin, sütünları ise vize ve final notlarımız ifade etsin. 1.sütün vize notumu 2.sütün final notum olsun.
//ilk dersim matematik ikincisi fizik üçüncüsü programlama olsun

//int dersler[3][2] = { {97,58}, {92,100}, {56,78}};

int dersler[3][2];

dersler[0][0]=97; //matematik dersinin vize notu
dersler[0][1]=58; //matematik dersinin final notu
dersler[1][0]=92; //fizik dersinin vize notu
dersler[1][1]=100; //fizik dersinin final notu
dersler[2][0]=56; //programlama dersinin vize notu
dersler[2][1]=78; //programlama dersinin final notu

//simdi derslerin ortalamalarını hesaplayalım(vize %40,final %60)

//mateamtik
float matOrt= (dersler[0][0]*0.40) + (dersler[0][1]*0.60);
//fizik
float fizOrt= (dersler[1][0]*0.40) + (dersler[1][1]*0.60);
//programlama
float progtOrt= (dersler[2][0]*0.40) + (dersler[2][1]*0.60);

printf("\nMatematik dersi ortalamaniz: %0.2f", matOrt);
printf("\nFizik dersi ortalamaniz: %0.2f", fizOrt);
printf("\nProgramla dersi ortalamaniz: %0.2f", progtOrt);
```

```
ÇIKTI:
Matematik dersi ortalamaniz: 73.60
Fizik dersi ortalamaniz: 96.80
Programla dersi ortalamaniz: 69.20
```

3. HAFTA ÖRNEKLER

3.1. Örnek: Dizilerin yapısıyla ilgili genel bir kod yazınız(dizi oluşturma,diziye eleman ekleme,gösterme vb).

3_1_ornek.c

```
1.  //dizilerin genel yapisi
2.  #include <stdio.h>
3.  int main()
4.  {
5.    int dizi2[5] = {5,81,9,-15,7};//dizi tanimlama
6.    printf("\nIlk olsuturulan Dizi: 0.Indis: %d 1.Indis: %d 2.Indis: %d 3.Indis: %d
    4.Indis: %d",dizi2[0], dizi2[1], dizi2[2],dizi2[3],dizi2[4]);//dizi gosterme
7.    //diziyi sonradan düzenleme
8.    dizi2[0] = 11;
9.    dizi2[1] = 4;
10.   dizi2[2] = 52;
11.   printf("\nDegistirilmis Dizi: 0.Indis: %d 1.Indis: %d 2.Indis: %d",dizi2[0],
    dizi2[1], dizi2[2]);//yeni dizi
12.
13.   printf("\nDizinin son hali:");
14.   for (int i = 0; i<5; i++)
15.     printf("\ndizi2[%d] = %d", i, dizi2[i]);
16.
17.   int dizi[3];
18.   printf("\n\nYeni dizinin indislerini giriniz:");
19.
20.   for (int i = 0; i< 3; i++)
21.   {
22.     printf("\ndizi[%d] giriniz:", i);
23.     scanf("%d",&dizi[i]);
24.   }
25.
26.   for (int i = 0; i<3; i++)
27.     printf("\nGirilen dizi[%d] = %d", i, dizi[i]);
28.   return 0;
29. }
```

ÇIKTI:

```
Ilk olsuturulan Dizi: 0.Indis: 5 1.Indis: 81 2.Indis: 9 3.Indis: -15 4.Indis: 7
Degistirilmis Dizi: 0.Indis: 11 1.Indis: 4 2.Indis: 52
Dizinin son hali:
dizi2[0] = 11
dizi2[1] = 4
```

```
dizi2[2] = 52
dizi2[3] = -15
dizi2[4] = 7

Yeni dizinin indislerini giriniz:
dizi[0] giriniz:5
dizi[1] giriniz:4
dizi[2] giriniz:3
Girilen dizi[0] = 5
Girilen dizi[1] = 4
Girilen dizi[2] = 3
```

3.2. Örnek: Disardan metin aliniz harf harf ekrana yazdiriniz vede ascii karşılıklarınıda hesaplayınız.

3_2_ornek.c

```
1.  //METİN ALIP harf harf yazdırma ve ascii lerini gosterme
2.  #include <stdio.h>
3.  int main()
4.  {
5.    char metin[100];
6.    int i;
7.
8.    printf("Metin giriniz:" );
9.    //scanf("%s", metin); BU SEKILDE ALINCA ARALARDAKI BOSLUGA KADAR
      ALIYOR GETS GIBI ALAMIYOR BU YUZDEN ALTTAKI ALTENATIFI BULDUM
10.   scanf("%[^\n]s", metin); //BU IFADE GETS e benzer sekilde ayni islemi
      yapiyor. gets gibi bolsuklar dahil karakterleir tutuyor
11.   printf("\nHarf - Ascii");
12.   for(i=0;metin[i]!='\0';i++)
13.   {
14.     printf("\n%c - %d",metin[i],metin[i]);
15.   }
16.
17.   return 0;
18. }
```

ÇIKTI:

```
Metin giriniz:hamza CELIK !

Harf - Ascii
h - 104
a - 97
```

```
m - 109
z - 122
a - 97
  - 32
C - 67
E - 69
L - 76
I - 73
K - 75
  - 32
! - 33
```

3.3. Örnek: Bir ders notu ortalaması hesaplama uygulaması yazınız. Dısardan n tane ders ismini alınız ve tutunuz o derse ait vize ve final notlarını tutunuz ve ortalamasını da ayrıca tutup en son çıktısını verin.

3_3_ornek.c

```
1.  //Dısardan n tane ders ismini alınız ve tutunuz o derse ait vize ve final
    notlarını tutunuz ve ortalamasını da ayrıca tutup en son çıktısını verin.
2.  #include <stdio.h>
3.  int main()
4.  {
5.
6.    float notlar[30][3]; //notlar dizisi
7.    char dersler[30][10]; // dersler dizisi
8.    int n,i,j;
9.
10. printf( "Kac tane ders girisi yapacaksiniz: ");
11. scanf( "%d", &n );
12. //derslerin isimlerini aldik
13. for (i = 0; i < n; i++)
14. {
15.   printf("\nDers Isim giriniz: ");
16.   scanf( "%s", dersler[i] );
17. }
18.
19. //deslerin not girislerini notlar dizisine atiyoruz
20.   for(i=0;i<n;i++)
21.   {
22.
23.     printf("\n%s Dersinin VIZE notunu giriniz: ",dersler[i]);
24.     scanf("%f",&notlar[i][0]);
```

```
25.     printf("\n%s Dersinin FİNAL notunu giriniz: ",dersler[i]);
26.     scanf("%f",&notlar[i][1]);
27.     notlar[i][2]=notlar[i][0]*0.40 + notlar[i][1]*0.60;
28.   }
29.
30.   for(i=0;i<n;i++)
31.   {
32.     printf("\n%s Vize: %f Final %f Ortalama
      %f",dersler[i],notlar[i][0],notlar[i][1],notlar[i][2]);
33.   }
34.
35.   return 0;
36. }
```

ÇIKTI:

```
Kac tane ders girisi yapacaksiniz: 3
Ders Isim giriniz: Matematik
Ders Isim giriniz: Fizik
Ders Isim giriniz: Kimya
Matematik Dersinin VIZE notunu giriniz: 100
Matematik Dersinin F▒NAL notunu giriniz: 85
Fizik Dersinin VIZE notunu giriniz: 54
Fizik Dersinin F▒NAL notunu giriniz: 68
Kimya Dersinin VIZE notunu giriniz: 97
Kimya Dersinin F▒NAL notunu giriniz: 58
Matematik Vize: 100.000000 Final 85.000000 Ortalama 91.000000
Fizik Vize: 54.000000 Final 68.000000 Ortalama 62.400002
Kimya Vize: 97.000000 Final 58.000000 Ortalama 73.599998
```

3.4. Örnek: Kullanıcıdan aldığınız metinde yine kullanıcıdan aldığınız karkaterin metinde hangi indislerde oldugunu baka dizide tutup ayrıca kaç kere tekrar ettigini bulup belirtin.

3_4_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    char metin[100];
5.    int indis[100];
6.    char aranan;
7.    int i,tekrar=0;
8.
```

```
9.    printf("Metin giriniz:" );
10.   //scanf("%s", metin); BU SEKILDE ALINCA ARALARDAKI BOSLUGA KADAR
      ALIYOR GETS GIBI ALAMIYOR BU YUZDEN ALTTAKI ALTENATIFI BULDUM
11.   scanf("%[^\n]s", metin); //BU IFADE GETS e benzer sekilde ayni islemi
      yapiyor. gets gibi bolsuklar dahil karakterleir tutuyor
12.   scanf("%c",&aranan);
13.   printf("\nAradiginiz karakteri giriniz: ");
14.   scanf("%c",&aranan);
15.
16.   for(i=0;metin[i]!='\0';i++)
17.   {
18.     if(metin[i]==aranan)
19.     {
20.       indis[tekrar]=i;
21.       tekrar++;
22.     }
23.   }
24. printf("Tekrar: %d",tekrar);
25. for(i=0;i<tekrar;i++)
26. {
27.   printf("\n%d.Tekrar: %d.Indis",i+1,indis[i]);
28. }
29.   return 0;
30. }
```

ÇIKTI:

```
Metin giriniz:hamza celik

Aradiginiz karakteri giriniz: a
Tekrar: 2
1.Tekrar: 1.Indis
2.Tekrar: 4.Indis
```

3.5. Örnek: Alınan integer sayının rakamlarının okunusunu yazı ile yazdırma.

3_5_ornek.c

```
1.  #include <stdio.h> //Standart giris cikis kutupahnemizi ekledik
2.  int main()
3.  {
4.          int i; //dongude for da kullanabilmek icin i diye degisken olsutruk
5.
```

```
6.          /* version 1 diziyi biz belirlemiştik
7.
8.          int sayi[] = {5,7,4,3,0}; //sayilarimizi dizimize attık
9.
10.         */
11.
12.         //version 2 sayiyi kullanidan aldik
13.         char sayi[100];
14.         printf("Sayinizi giriniz: ");
15.         scanf("%s",&sayi);
16.
17.         for (int i = 0; sayi[i] != '\0' ; i++)
18.         {
19.                 if (sayi[i]== '0')
20.                 {
21.                         printf("\n sifir");
22.
23.                 }else if (sayi[i]== '1' )
24.                 {
25.                         printf("\n bir");
26.
27.                 }else if (sayi[i]== '2' )
28.                 {
29.                         printf("\n iki");
30.
31.                 }else if (sayi[i]== '3' )
32.                 {
33.                         printf("\n uc");
34.
35.                 }else if (sayi[i]== '4' )
36.                 {
37.                         printf("\n dort");
38.
39.                 }else if (sayi[i]== '5' )
40.                 {
41.                         printf("\n bes");
42.
43.                 }else if (sayi[i]== '6' )
44.                 {
45.                         printf("\n alti");
46.
47.                 }else if (sayi[i]== '7' )
48.                 {
49.                         printf("\n yedi");
50.
51.                 }else if (sayi[i]== '8' )
```

```
52.                     {
53.                             printf("\n sekiz");
54.
55.                     }else if (sayi[i]== '9' )
56.                     {
57.                             printf("\n dokuz");

58.                     }
59.             }
60.
61.             return 0;
62. }
```

ÇIKTI:

```
Sayinizi giriniz: 156
 bir
 bes
 alti
```

3.6. Örnek: Dışardan alınan tam sayıları diziye atıp kucukten buyuge buyukten kucuge sıralama.

3_6_ornek.c

```
1.  #include <stdio.h>
2.  int main(void)
3.  {
4.              int i=0,j=0,x=0,boyut=0;
5.              int dizi[100]; //tamsayıarı tutacagımız dizi
6.     int orj_d[100]; //dizinin kopyasını alıyoruz
7.              int buy_k[100]; //buyukten kucuge sıralamayı tutacagımız dizi
8.              int kuc_b[100]; //kcuukten buyuge sıralamayı tutacagımız dizi
9.                do{
10.     printf("Kac tane sayiyi siralamak istersin(pozitif sayi girin): ");
11.     scanf("%d",&boyut);
12.  }while(boyut<0);
13.
14.             for(i=0;i<boyut;i++)
15.             {
16.              printf("\n%d. Sayiyi giriniz: ",i+1);
17.                     scanf("%d",&dizi[i]);
18.             }
19.
20.
```

```
21.             for(i=0;i<boyut;i++) orj_d[i]=dizi[i];
22.
23.             for(i=0;i<boyut;i++)
24.             {
25.                     for(j=0;j<boyut;j++)
26.                     {
27. if(dizi[i]>dizi[j])
28. {
29.             x=dizi[i];
30.             dizi[i]=dizi[j];
31.             dizi[j]=x;
32. }
33.                     }
34.             }
35.
36.             for(i=0;i<boyut;i++)
37.             {
38.                     buy_k[i]=dizi[i];
39.                     kuc_b[i]=dizi[boyut-1-i];
40.             }
41.
42.   printf("\n\nOrhinal Sira\n");
43.   for(i=0;i<boyut;i++)
44.   {
45.      printf("%4d",orj_d[i]);
46.   }
47.             printf("\n\nKucuktenB  \n");
48.
49.             for(i=0;i<boyut;i++)
50.             {
51.                printf("%4d",kuc_b[i]);
52.             }
53.
54.             printf("\n\nBuyuktenK\n");
55.   for(i=0;i<boyut;i++)
56.   {
57.      printf("%4d",buy_k[i]);
58.   }
59.             return 0;
60. }
```

ÇIKTI:

```
Kac tane sayiyi siralamak istersin(pozitif sayi girin): -5
Kac tane sayiyi siralamak istersin(pozitif sayi girin): 5
1. Sayiyi giriniz: 15
2. Sayiyi giriniz: -7
3. Sayiyi giriniz: 0
```

```
4. Sayiyi giriniz: 1
5. Sayiyi giriniz: 9
Orjinal Sira
  15  -7   0   1   9
KucuktenB
  -7   0   1   9  15
BuyuktenK
  15   9   1   0  -7
```

3.7. Örnek: Belirlediginiz bir metinin dışardan aynı şekilde girilmesini bekleyiniz doğru girilmez ise aynı işlem doğru girileseye kadar tekarlansın.

3_7_ornek.c

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  int main()
4.  {
5.    char sifre[]="HaMza";
6.    char girilen[100];
7.    int i=0,n=0;//i kontrol degiskenimiz i 0 dan kucuk oldugu surece whiledan
      cıkılamyacak, n dogru karkter sayisiise uzunluk
8.    do{
9.      if(i<0)printf("\nHatali giris.Tekrar ");
10.   printf("\"HaMza\" Gordugunuz metini doggru bir sekilde girin: ");
11.   gets(girilen);n=0;
12.   for(int j=0;girilen[j]!='\0';j++)
13.   {
14.     if(girilen[j]==sifre[j]) n++;//iki dizininde belirtilen indisleri eşit ise n i
      arttırıyoruz
15.   }
16.   if(n==strlen(sifre)) i = 1;else i--;//eger n degeri sifre nin uzunlugu ile aynı
      ise whiledan cıkartıyoruz ki dogru girilmil demektir
17.  }while(i<=0);
18.  printf("Tebrikler basarili!");
19.  return 0;
20. }
```

ÇIKTI:

```
"HaMza" Gordugunuz metini doggru bir sekilde girin: Hamza
Hatali giris.Tekrar "HaMza" Gordugunuz metini doggru bir sekilde girin: hamza
Hatali giris.Tekrar "HaMza" Gordugunuz metini doggru bir sekilde girin: haMza
Hatali giris.Tekrar "HaMza" Gordugunuz metini doggru bir sekilde girin: HaMza
Tebrikler basarili!
```

3.8. Örnek: Aşağıdaki kare matrisin kodunu yazınız.

0 0 0 1

0 0 1 0

0 1 0 0

1 0 0 0

3_8_ornek.c

```
1.  //kare matris
2.  #include <stdio.h>
3.  int main (void)
4.  {
5.          int i=0,j=0;
6.          int x = 3;//Kat sayi deger sonradan degisecek
7.          printf("Matrix boyutunu giriniz: ");
8.          scanf("%d",&x);
9.          int a[x][x];//Matrixin kare formati degeri mesela x 3 olsaydi 3 e 3 bir
    kare icine 0 1 yazardi
10.         printf("\n");
11.         printf("\n");
12.         for(i=0;i<x;i++)//Dikey
13.         {
14.                 for(j=0;j<x;j++)//Yatay
15.                 {
16.                   if(i+j==x-1)
17.                   {
18.                    a[i][j]=1;
19.                   }else
20.                   {
21.                    a[i][j]=0;
22.                   }
23.                  printf("%d ",a[i][j]);
24.
25.                   }
26.                   printf("\n");
27.           }
28.         return 0;
29. }
```

ÇIKTI:

```
Matrix boyutunu giriniz: 4
0 0 0 1
0 0 1 0
0 1 0 0
1 0 0 0
```

3.9. Örnek: Disardan alinan metnin tesrten yazdirilmasi.

3_9_ornek.c

```
1.  #include <stdio.h>
2.  #include <string.h>
3.  int main(void)
4.  {
5.          int i=0,j=0,x=0,boyut=0;
6.          char dizi[100]; //tamsayıarı tutacagımız dizi
7.          char orj_d[100]; //dizinin kopyasını alıyoruz
8.
9.   printf("metin giriniz: ");
10. gets(orj_d);
11. boyut=strlen(orj_d);
12.         for(i=0;i<boyut;i++)
13.         {
14.         //buy_k[i]=dizi[i];
15.                 dizi[i]=orj_d[boyut-1-i];
16.         }
17. printf("\n%s Tersten ",orj_d);
18. for(i=0;i<boyut;i++)
19.         {
20.         //buy_k[i]=dizi[i];
21.                 printf("%c",dizi[i]);
22.         }
23.         return 0;
24. }
```

ÇIKTI:

```
metin giriniz: hamza celik
hamza celik Tersten kilec azmah
```

3.10. Örnek:

3_10_ornek.c

```
1.
```

ÇIKTI:

4. HAFTA

A. GÖSTERİCİLER (pointer) ve GENEL YAPISI

Bilindiği üzere her değişken bellekte belli bir alan kaplar. Bu alanların adresleri vardır ve program çalıştırıldığında CPU tanımlanan değişkene ulaşmak için bu adrese gider. Programcının adreslerle uğraşmasına gerek yoktur; çünkü derleyici bunu yapar. Bazı durumlarda ise bu işlere dalmak kaçınılmazdır. Böyle durumlarda pointerla kullanılır.[5]

1) Değişkenlerin adreslerini gösterme (&):

```
int x = 5;
printf(“x degiskeni: %d”, x);
```

Bu şekilde ekrana x değişkeninin değerini yani “x degiskeni: 5” bastırmış olduk

```
int x = 5;
printf("x degiskenin adresi: %p", &x);
```

x değişkenini çağırıken önüne & işareti koyunca x’in adresini döndürmüş olduk ve adresin değerini %p ile gösteriyoruz ve ekranda x değişkenin bellkte tutulu olduğu lanın adresini görürüz “x degiskenin adresi: 0061FF1C”.

2) Göstericiler (*):

Göstericiler (işaretçiler), değişkenleri tutmak yerine onların adreslerini tutarlar veya adreslerindeki değeri işaret eder diyebiliriz.

```
int x = 5;
printf("x degiskenin adresinin içindeki deger: %d", *&x);
```

Öncelikle &x ile x değişkenin adresini aldık ve ardından *&x ile adresini işaret etmiş olduk diyebiliriz ve bu bize x değişkeninin adresindeki değeri döndümüş oldu ve dönen tam sayı tipinde olduğu için %d ile de ekranda göstermiş olduk yani ”x degiskenin adresinin icindeki deger: 5”.

[5] http://yapbenzet.kocaeli.edu.tr/cpp-gostericiler-pointerlar/

3) Göstericilerin Tanımlanması

```
int x  = 5;

int *p;
p=&x;

//int *p = &x;

printf("p nin degeri: %d", *p); //Cıktı: p nin degeri: 5
printf("p nin adresi: %p", p); //Cıktı: p nin adresi: 0061FF1C
```

Adres göstermeyi &x ile yapıyorduk ve bu bize adress döndürüyordu, pointerımızı tanımlarken * ile işaret edeceğiz ve adres int türünde dönecek ve tanımlamış olacağız.

Diğer veritipi tanımlamaları:

```
float f  = 7.8;
char c = '!';
float *p;
char *pc;
p=&f;
pc=&c;
//float *p = &f;

printf("p nin degeri: %f", *p); //Cıktı: p nin degeri: 7.8
printf("p nin adresi: %p", p); //Cıktı: p nin adresi: 0061FF10
```

B. Gösterici ve diziler arasındaki bağlantı.[6]

C dilinde her dizi bir pointer her pointer da doğal bir dizidir.

char str[80], *p1;
p1 = str;

Burada p1, str dizisinin – stringinin – ilk elamaninin adresinin degerini alir. Yani string adi, aslinda o stringin hafizadaki baslangiç adresini = stringin ilk karakterinin adresini tutmaktadir. str dizisinin 5. elemanina erismek için ise;

str[4] veya *(p1+4)

ifadelerini kullaniriz. Her ikisinin de anlami aynidir.

[6] http://bilgisayarkavramlari.sadievrenseker.com/2007/10/16/pointer-gosterici-2/

C’de dizi elemanlarina 2 sekilde ulasilir: pointer kullanimi ile ve indis kullanimi ile. Indis kullanimi gelistirme ve anlama bakimindan bir kolaylik saglasa da, hiz önemli bir konu oldugundan C programcilari genelde dizi erisimini pointer kullanarak yaparlar.

```
int dizi[5]={5,15,20,25,30};
int *p, i,*d;
printf("Dizi adresi %p, dizi[0] adresi %p\n",&dizi, &dizi[0]);

CIKTI: Dizi adresi 0061FF00, dizi[0] adresi 0061FF00
```

Buradaki ornektede gördüğümüz gibi dizilleri pointerlarla kullanırken iki türlü çağırabiliz çünkü dizi indis 0 ‘ın adresi dizi adresine eşittir.

Ve dizinin önüne Yıldız * pointer koymuyoruz çünkü diziler bir pointer, pointerda bir dizi olduğu için.

4. HAFTA ÖRNEKLER

4.1. Örnek: Pointerların genel yapısıyla ve kullanımı ile alakalı örnekler veriniz.

4_1_ornek.c

```
1.  //pointerlar genel
2.  #include <stdio.h>
3.  int main()
4.  {
5.    int x = 5;
6.    //int *p = &x;
7.    int *p;
8.    p = &x;
9.    printf("\nx=5 deiskeni ile yapilanlar:");
10.   printf("\nX degiskeni = %d(%cd -x)",x,'%');
11.   printf("\nx degiskenin adresi: %p(%cp - &x)", &x,'%');
12.   printf("\nx degiskenin adresinin icindeki deger: %d(%cd - *&x)", *&x,'%');
13.   printf("\nx degiskenin adresinin int hali %d(%cd - &x)", &x,'%');
14.
15.   printf("\n\n\nint *p = &x; deiskeni ile yapilanlar:");
16.   printf("\n\np degeri,p nin isaretledigi adresin icindeki deger(x degiskeni) =
      %d (%cd - *p)",*p,'%');
17.   printf("\np adresi = %p(pointerin adresi) - %p(pointerın isaret ettigi adres)
      (%cp - &p - p)", &p,p,'%');
18.   printf("\np degiskenin adresinin int hali %d(%cd - p)", p,'%');
19.
20.   x = 22;
21.   printf("\n\n\n\nint *p = &x; deiskeni ile yapilanlar (X degistirildi 22 yapildi):");
22.   printf("\nAdres x: %p\n", &x);
23.   printf("Deger x: %d\n\n", x);  // 22
24.
25.   printf("Adres pointer p: %p\n", p);
26.   printf("Pointer p nin ici: %d\n\n", *p); // 22
27.
28.   x = 11;
29.   printf("Adres pointer p: %p\n", p);
30.   printf("Pointer p nin ici: %d\n\n", *p); // 11
31.
32.   *p = 2;
33.   printf("Adres x: %p\n", &x);
34.   printf("Deger x: %d\n\n", x); // 2
35.
36.   return 0;
37. }
```

ÇIKTI:

```
x=5 deiskeni ile yapilanlar:
X degiskeni = 5(%d -x)
x degiskenin adresi: 0061FF1C(%p - &x)
x degiskenin adresinin icindeki deger: 5(%d - *&x)
x degiskenin adresinin int hali 6422300(%d - &x)

int *p = &x; deiskeni ile yapilanlar:
p degeri,p nin isaretledigi adresin icindeki deger(x degiskeni) = 5 (%d - *p)
p adresi = 0061FF18(pointerin adresi) - 0061FF1C(pointer─n isaret ettigi adres) (%p
- &p - p)
p degiskenin adresinin int hali 6422300(%d - p)

int *p = &x; deiskeni ile yapilanlar (X degistirildi 22 yapildi):
Adres x: 0061FF1C
Deger x: 22

Adres pointer p: 0061FF1C
Pointer p nin ici: 22
Adres pointer p: 0061FF1C
Pointer p nin ici: 11
Adres x: 0061FF1C
Deger x: 2
```

4.2. Örnek: Pointerlarla vize final notu alıp ortalamasını hesaplayın (vize0.40 – final0.60)

4_2_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int vize=0,final=0, *pv,*pf;
5.    float *ort;
6.    printf("\nVize notu giriniz : ");
7.    scanf("%d", &vize);
8.    printf("\nFinal notu giriniz : ");
9.    scanf("%d", &final);
10.   pv=&vize;
11.   pf=&final;
12.   printf("Ortalama= %0.2f", ((*pv*0.4)+(*pf*0.6)));
13.   return 0;
14. }
```

ÇIKTI:

```
Vize notu giriniz : 100
Final notu giriniz : 100
Ortalama= 100.00
```

4.3. Örnek: Belirlenen diziyi iki farklı pointer yolu ile ekrana yazdiriniz.

4_3_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int dizi[5]={5,15,20,25,30};
5.    int*p, i,*d;
6.
7.    d=dizi;
8.    printf("Pointer'a direk dizi(p=dizi) atayarak dizi yi yazdirma:\n");
9.    for(i=0;i<5;i++)
10.   {
11.     printf("%d. elemani %d - %p\n", i,*d,d);
12.     d++;
13.   }
14.   printf("\n\n\n");
15.   printf("Pointer'a dizi indisi(p=&dizi[0]) atayarak dizi yi yazdirma:\n");
16.   p=&dizi[0];
17.   for(i=0;i<5;i++)
18.     printf("%d. elemani %d - %p\n", i,*(p+i),p+i);
19.   return 0;
20. }
```

ÇIKTI:

```
Pointer'a direk dizi(p=dizi) atayarak dizi yi yazdirma:
0. elemani 5 - 0061FF00
1. elemani 15 - 0061FF04
2. elemani 20 - 0061FF08
3. elemani 25 - 0061FF0C
4. elemani 30 - 0061FF10
Pointer'a dizi indisi(p=&dizi[0]) atayarak dizi yi yazdirma:
0. elemani 5 - 0061FF00
1. elemani 15 - 0061FF04
2. elemani 20 - 0061FF08
3. elemani 25 - 0061FF0C
4. elemani 30 - 0061FF10
```

4.4. Örnek: Belirli diziyi pointerla ile tersten yazdiriniz.

4_4_ornek.c

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.    int dizi[5]={5,15,20,25,30};
5.    int*p, i,*d;
6.
7.    printf("\n\n\n");
8.    printf("duz:\n");
9.    p=&dizi[0];
10.   for(i=0;i<5;i++)
11.   printf("%d. elemani %d - %p\n", i,*(p+i),p+i);
12.
13.   printf("\n\ntersten:\n");
14.   d=dizi+4;
15.
16.   for(i=0;i<5;i++)
17.   {
18.     printf("%d. elemani %d - %p\n", 4-i,*(d-i),(d-i));
19.   }
20.
21.   return 0;
22. }
```

ÇIKTI:

```
duz:
0. elemani 5 - 0061FF00
1. elemani 15 - 0061FF04
2. elemani 20 - 0061FF08
3. elemani 25 - 0061FF0C
4. elemani 30 - 0061FF10

tersten:
4. elemani 30 - 0061FF10
3. elemani 25 - 0061FF0C
2. elemani 20 - 0061FF08
1. elemani 15 - 0061FF04
0. elemani 5 - 0061FF00
```

5. HAFTA

A. FONKSİYONLAR

Fonksiyonlar bizi, kodlarımızda tekrar eden yinelenen kodları tekrar tekrar aynı kodları kopyala yapıştır yapmaktan kurtarıyor. Fonksiyonlara bir kere yazdığımız kodu her seferinde uzun uzun uza o kodu yazmaya ugrasmadan sadece tek satırla fonkisyonu çağırarak istediğimiz yerden ulaşabiliriz. Fonksiyon oluşturduğumuzda dikkat etmemiz gerekenle eğer fonksiyon geir dönüş yapacak ise mutlaka return ile dönüş yaptırılmalı. İki fonksiyonun ismi aynı olamaz.

C'de iki tür fonksyion vardır iteratif fonksiyon yinelemeli ve recursive özyinelemeli. Recursive fonksiyon kendisini çağıran fonksiyondur.

B. FONKSİYON TANIMLAMA

Fonksiyon tanımlarken iki şekilde tanımlayabiliriz birisi main fonksiyonunun üstünde fonksiyonu tanımlarız. İçeriğini ise main fonksiyonunun altında:

5_1_ornek.c

```
#include <stdio.h>

int topla(int,int); //fonksiyonumuzu main'nin ustunde tanimladik

int main()
{
  int a,b,sonuc;
  printf("Birinci sayiyi giriniz: ");
  scanf("%d",&a);
  printf("Ikinci sayiyi giriniz: ");
  scanf("%d",&b);

  sonuc = topla(a,b); //bu sekilde ise fonksiyonumuzu kullandik
  printf("Sonuc %d",sonuc);
  return 0;
}

int topla(int x,int y) //main'nin altinda içini girdik
{
  return x+y;
}
```

Veya direk main fonksiyonunun üstünde oluşturabilirdik.

5_1_ornek.c

```
#include <stdio.h>

int topla(int x, int y) //hem fonksiyonu hem icerigini tek yerde tanimladik
{
  return x + y;
}

int main()
{
  int a,b,sonuc;
  printf("Birinci sayiyi giriniz: ");
  scanf("%d",&a);
  printf("Ikinci sayiyi giriniz: ");
  scanf("%d",&b);

  sonuc = topla(a,b); //bu sekilde ise fonksiyonumuzu kullandik
  printf("Sonuc %d",sonuc);
  return 0;
}
```

Bu yukardaki örnekte olduğu gibi return bize geri değer (x+y)'yi döndürür. Ve main fonksiyonumuzdaki sonuc değişkenine atıyor dönen değeri.

Pointer ve fonksiyon:

5_1_ornek.c

```
#include <stdio.h>

void kareal(int *sayi)
{
  *sayi=(*sayi) * (*sayi);
}

void kareal2(int sayi)
{
  sayi = sayi*sayi;
}

int main()
{
  int s,s1;
  printf("Sayiyi giriniz: ");
  scanf("%d",&s);

  kareal(&s);
  printf("Girilen sayinin karesi: %d",s);

  printf("Sayiyi giriniz: ");
  scanf("%d",&s1);

  kareal2(s1);
  printf("Girilen sayinin karesi: %d",s1);
  return 0;
}
```

Bu örnekte ise fonksiyonda pointer kulllaımı görülmekte. Biz kareal fonksiyonumuza adres gönderdiğimizde fonksiyonda onun gösterdiği adresindeki değeri çarpıyor. Ve hali ile return yapmadan main foksiyonumuzdaki değişken değişiyor. Ama pointer kullanmadan yaparsak kareal2 fonksiyonunda görüldüğü gibi sadece fonksiyonun içindeki değeri çarpıyor ve haliyle main fonksiyonumuzdaki değer değişmiyor.

5. HAFTA ÖRNEKLER

5.1. Örnek: Yinelemeli fonksiyonların yapısıyla ilgili örnek veriniz. Fonksiyonda pointer değeri değiştirince işaret ettiği adresteki değerin değerini gözlemleyiniz.

3_1_ornek.c

```
1.  #include <stdio.h>
2.
3.  int goster(int *x)
4.  {
5.    *x+=10;
6.    return *x;
7.  }
8.
9.  int topla(int x,int y) //hem fonksiyonu hem icerigini tek yerde tanimladik
10. {
11.   return x+y;
12. }
13.
14. int main()
15. {
16.   int a,b,sonuc,*ptr;
17.
18.   printf("Toplama Fonksiyonu Birinci sayiyi giriniz: ");
19.   scanf("%d",&a);
20.   printf("Toplama Fonksiyonu Ikinci sayiyi giriniz: ");
21.   scanf("%d",&b);
22.
23.   sonuc = topla(a,b); //bu sekilde ise fonksiyonumuzu kullandik
24.   printf("\nSonuc %d",sonuc);
25.
26.   printf("\nSayiyi giriniz: ");
27.   scanf("%d",&a);
28.   ptr=&a;
29.   printf("\nDegismeden *ptr nin degeri %d",*ptr);
30.   printf("\nDegismeden a nin degeri %d",a);
31.
32.   printf("\nPointerin degerini degistirdikten sonra *ptr nin degeri
      %d",goster(ptr));
33.   printf("\nPointerin degerini degistirdikten sonra a nin degeri %d",a);
34.   return 0;
35. }
```

ÇIKTI:

```
Toplama Fonksiyonu Birinci sayiyi giriniz: 28
Toplama Fonksiyonu Ikinci sayiyi giriniz: 14

Sonuc 42
Sayiyi giriniz: 9

Degismeden *ptr nin degeri 9
Degismeden a nin degeri 9
Pointerin degerini degistirdikten sonra *ptr nin degeri 19
Pointerin degerini degistirdikten sonra a nin degeri 19
```

5.2. Örnek: Pointer ve fonksiyon ile basit bir kare alma örneği.

3_2_ornek.c

```
1.  //pointer ve fonksiyon
2.  #include <stdio.h>
3.
4.  void kareal(int *sayi)
5.  {
6.    *sayi=(*sayi) * (*sayi);
7.  }
8.
9.  void kareal2(int sayi)
10. {
11.   sayi = sayi*sayi;
12. }
13.
14. int main()
15. {
16.   int s,s1;
17.   printf("Sayiyi giriniz: ");
18.   scanf("%d",&s);
19.
20.   kareal(&s);
21.   printf("Pointer sayesinde girilen sayinin karesi: %d",s);
22.
23.   printf("\nSayiyi giriniz: ");
24.   scanf("%d",&s1);
25.
26.   kareal2(s1);
27.   printf("Pointer kullanmadan girilen sayinin karesi: %d",s1);
28.
29.   return 0;
30. }
```

ÇIKTI:

```
Sayiyi giriniz: 4
Pointer sayesinde girilen sayinin karesi: 16
Sayiyi giriniz: 4
Pointer kullanmadan girilen sayinin karesi: 4
```

5.3. Örnek: Fonksiyon kullanarak bir program menusu yapimi.

5_3_ornek.c

```
1.  //fonksiyonlar program menusu basitrma
2.  #include <stdio.h>
3.
4.  void aciklama()
5.  {
6.          printf("*********                              *********\n");
7.          printf("*****       FONKSIYONLAR ILE MENU YAPIMI       *****\n");
8.          printf("*********                              *********\n");
9.          printf("*****                                  *****");
10.         printf("\n***             ACIKLAMA                  ***\n");
11.         printf("\n    [ 0 ]   Cikis");
12.         printf("\n    [ 1 ]   Bilgi");
13. }
14.
15. int main()
16. {
17.   aciklama();
18.   return 0;
19. }
```

ÇIKTI:

```
*********                                   *********
*****       FONKSIYONLAR ILE MENU YAPIMI       *****
*********                                   *********
*****                                           *****
***                     ACIKLAMA                  ***
   [ 0 ]   Cikis
   [ 1 ]   Bilgi
```

5.4. Örnek: Fonksiyon yazarak us alma işlemi yaptırınız.

5_4_ornek.c

```
1.  //fonksiyon ile us alma
2.  #include <stdio.h>
3.
4.  float usAlma(float x, float y)
5.  {
6.          int i=0; float carpim=1;
7.          if(y!=(int)y)     printf("\nussunu tam sayi girmeliydiniz. us %f
    yuvarlanmistir: %d\n",y,(int)y);
8.          for(i=0;i<(int)y;i++) carpim *=x;
9.
10.         return carpim;
11. }
12.
13. int main()
14. {
15.   float t,u;
16.   printf("Taban: ");
17.   scanf("%f",&t);
18.   printf("Us: ");
19.   scanf("%f",&u);
20.   printf("\nSonuc: %.2f",usAlma(t,u));
21.   return 0;
22. }
```

ÇIKTI:

```
Taban: 2
Us: 5

Sonuc: 32.00
```

5.5. Örnek: Fonksiyon kullanarak menu, us alma, toplama, çarpma, bolme, cikarma,faktoriyel iceren fonksiyonları oluştuurup işlemlerini yaptırınız.

5_5_ornek.c

```
1.  #include <stdio.h>
2.
3.  void aciklama()
4.  {
5.    printf("\n\n\n*********                    *********\n*****       Hesap Makinesi
    *****\n*********                    *********");
```

```
6.    printf("\n[0] Cikis\n[1] Toplama\n[2] Cikarma\n[3] Carpma\n[4] Bolme\n[5]
    Us Alma\n[6] Faktoriyel\n[7] Menu");
7.  }
8.  void topla(float x, float y)
9.  {
10.   if (x == (int)x & y == (int)y)
11.     printf("%d + %d Sonuc: %d",(int)y,(int)x,(int)y + (int)x);
12.   else
13.     printf("%f + %f Sonuc: %.2f",y,x,y+x);
14. }
15. void cikar(float x, float y)
16. {
17.   if (x == (int)x & y == (int)y)
18.     printf("%d - %d Sonuc: %d",(int)y,(int)x,(int)y - (int)x);
19.   else
20.     printf("%f - %f Sonuc: %.2f",y,x,y-x);
21. }
22. void carp(float x, float y)
23. {
24.   printf("(%f) * (%f) Sonuc: %.2f",y,x,y*x);
25. }
26. void bol(float x, float y)
27. {
28.   printf("(%f) / (%f) Sonuc: %.2f",y,x,y/x);
29. }
30. float faktoriyel(float x)
31. {
32.           if(x==0) return 1;
33.           return x*faktoriyel(x-1);
34. }
35. void usAlma(float y, float x)
36. {
37.           int i=0; float carpim=1;
38.           if(y!=(int)y)      printf("\nussunu tam sayi girmeliydiniz. us %f
    yuvarlanmistir: %d\n",y,(int)y);
39.           for(i=0;i<(int)y;i++) carpim *=x;
40.   printf("(%f) ^ (%f) Sonuc: %.2f",x,y,carpim);
41.
42. }
43. int secim()
44. {
45.   int sec=-1;
46.   do{
47.     printf("\nSeciminiz: ");
48.     scanf("%d",&sec);
49.   }while(sec<0 || sec>7);
```

```
50.   return sec;
51. }
52. float sayiAl(int n)
53. {
54.   float s;
55.   printf("\n%d. Sayiyi giriniz: ",n);
56.   scanf("%f",&s);
57.   return s;
58. }
59. int main()
60. {
61.   aciklama();
62.   int cikis=0;
63.   float faktor=0;
64.   while(cikis==0)
65.   {
66.     switch(secim())
67.     {
68.       case 0: cikis = 1; break;
69.       case 1: topla(sayiAl(2),sayiAl(1));break;
70.       case 2: cikar(sayiAl(2),sayiAl(1));break;
71.       case 3: carp(sayiAl(2),sayiAl(1));break;
72.       case 4: bol(sayiAl(2),sayiAl(1));break;
73.       case 5: usAlma(sayiAl(2),sayiAl(1));break;
74.       case 6:
75.         while(faktor<1)
76.         {
77.           printf("\nFaktoriyel icin 1 den buyuk sayi giriniz: ");
78.           scanf("%f",&faktor);
79.         }
80.         printf("\n%f! = %f",faktor,faktoriyel(faktor));
81.         faktor=0;
82.         break;
83.       case 7: aciklama(); secim(); break;
84.         default: break;
85.     }
86.   }
87.   return 0;
88. }
```

ÇIKTI:

```
*********                    *********
*****       Hesap Makinesi       *****
*********                    *********
[0] Cikis
[1] Toplama
[2] Cikarma
```

```
[3] Carpma
[4] Bolme
[5] Us Alma
[6] Faktoriyel
[7] Menu
Seciminiz: 4

1. Sayiyi giriniz: 5

2. Sayiyi giriniz: 2
(5.000000) / (2.000000) Sonuc: 2.50
Seciminiz: 5

1. Sayiyi giriniz: 2

2. Sayiyi giriniz: 5
(2.000000) ^ (5.000000) Sonuc: 32.00
Seciminiz: 6

Faktoriyel icin 1 den buyuk sayi giriniz: 4

4.000000! = 24.000000
Seciminiz: 0
```

5.6. Örnek: Bir dizideki elemanların toplamını ve buyukten kucuge kucukten buyuge siralayan ve diziyi ters ve duz fonksiyonları yazın.

5_6_ornek.c

```
1.  //fnksiyonlar
2.  #include <stdio.h>
3.  void sirala(int *dizi,int n)
4.  {
5.    int i,j, temp;
6.    for(i=1;i<n;i++)
7.    {
8.      for (j=0;j<10 - 1;j++)
9.      {
10.       if (*(dizi+j)>*(dizi+j+1))
11.       {
12.         temp=*(dizi+j);
13.         *(dizi+j)=*(dizi+1+j);
```

```
14.          *(dizi+j+1)=temp;
15.        }
16.     }
17.   }
18. }
19. void yazdir(int *dizi,int n,int secim)
20. {
21.   if(secim==1)
22.   {
23.     printf("\n Duz:");
24.     for(int i=0;i<n;i++)
25.       printf("\n%d. = %d",i,*(dizi+i));
26.   }else
27.   {
28.     printf("\n Ters:");
29.      for(int i=n-1;i>=0;i--)
30.         printf("\n%d. = %d",i,*(dizi+i));}
31.
32. }
33. int toplam(int *dizi,int n)
34. {
35.   int toplam=0;
36.   for(int i=0;i<n;i++)
37.     toplam+=*(dizi+i);
38.   return toplam;
39. }
40. int main()
41. {
42.   int dizilar[5]={4,7,2,10,5};
43.   yazdir(&dizilar[0], 5,1);
44.   yazdir(&dizilar[0], 5,0);
45.   printf("\nToplam %d",toplam(&dizilar[0], 5));
46.   sirala(&dizilar[0], 5);printf("\nSiralandi.");
47.   yazdir(&dizilar[0], 5,1);
48.   yazdir(&dizilar[0], 5,0);
49.   printf("\nToplam %d",toplam(&dizilar[0], 5));
50.   return 0;
51. }
```

ÇIKTI:

```
Duz:
0. = 4
1. = 7
2. = 2
3. = 10
4. = 5
 Ters:
```

```
4. = 5
3. = 10
2. = 2
1. = 7
0. = 4
Toplam 28
Siralandi.
 Duz:
0. = 2
1. = 4
2. = 5
3. = 7
4. = 10
 Ters:
4. = 10
3. = 7
2. = 5
1. = 4
0. = 2
Toplam 28
```

5.7. Örnek: Pointer ve fonksiyon kullanarak iki degerin degerlerinin takasını yapan fonksiyonlarını yazınız.

5_7_ornek.c

```
1.  #include <stdio.h>
2.
3.  void takas(int *x,int *y)
4.  {
5.    int gecici;
6.    gecici = *x;
7.    *x=*y;
8.    *y=gecici;
9.  }
10.
11. int main()
12. {
13.   int x,y;
14.   printf("X degerini girin: ");
15.   scanf("%d",&x);
16.
```

```
17.  printf("Y degerini girin: ");
18.  scanf("%d",&y);
19.
20.  printf("\nX degeriniz: %d Y degeriniz: %d",x,y);
21.
22.  takas(&x,&y);
23.  printf("\n\nTakas Yapildi\n\nX degeriniz: %d Y degeriniz: %d",x,y);
24.  return 0;
25. }
```

ÇIKTI:

```
X degerini girin: 10
Y degerini girin: 20

X degeriniz: 10 Y degeriniz: 20

Takas Yapildi

X degeriniz: 20 Y degeriniz: 10
```

5.8. Örnek:

3_2_ornek.c

```
52.
```

ÇIKTI:

5.9. Örnek:

3_3_ornek.c

```
26.
```

ÇIKTI:

5.10. Örnek:

3_2_ornek.c

53.

ÇIKTI:

5.11. Örnek:

3_3_ornek.c

27.

ÇIKTI:

6. HAFTA

A. DİNAMİK BELLEK KULLANIMI ve YÖNETİMİNİN KAVRANMASI

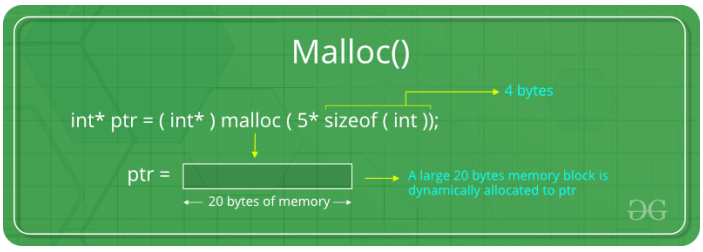
C de diziler oluşturduğumuzda dizilerimizin boyutunu başta belirtiyorduk ve program belirtilen alanı program sonlanıncaya kadar saklı tutup değiştirlmesine izin vermiyordu bu tür dizilere statik dizi deniyor. Statik diziler program çalışırken değiştirilemez. Ancak bazen program çalışırken dizimizin boyutunu artırmamız veya azaltmamız gerekebiliyor, işte bu tür dizilere de dinamik dizi deniyor. Dinamik diziler program çalışırken dizi boyutlarını değiştirebilmemizi sağlıyor.
Alan-hafıza yönetiminin de çok küçük alanlara bile ihtiyaç duyulduğundan dinamik bellek yönetimi ile duruma göre alan ihtiyacına göre alan düşürülür veya artırılır. Eğer alan ayrılmazsa **NULL** döndürür.Bu yüzden dinamik bellek yönetimi çok önemlidir.
C' de dinamik bellek yönetimini kolaylaştırmak için C tarafından **<stdlib.h>** altında tanımlanan 4 kütüphane fonksiyonu vardır . Bunlar:

B. DİNAMİK BELLEK FONKSİYONLARI

1) **malloc():** Bellekte alan ayırma.

Bu fonksiyon bellekten bizim belirlediğimiz miktarda bir alanın ayrılmasını sağlar ve bu alanı başlangıç adresini tutan bir pointer döndürür. Biz bu bellek alanını free fonksiyonu ile boşaltmadığımız sürece işletim sistemi bu bellek alanına dokunmaz.

```
int *ptr;
ptr = (int *) malloc (100 * sizeof (int));
```



Burada ptr isminde bir pointer oluşturduk ve sonrasında, ptr ile dizi oluşturduk yani `(int *)` ile integer türünde bir dizi olacağını belirttik `malloc` ile alan ayrımı olacağını belirttik `(100 * sizeof (int))` ile burada ise 100 elemanlı bir dizi integer bir dizi olacağını söyledik nasıl mı? 100 tane * integerın alan dakapladığı boyut yapmış olduk buda bize 100 tane integer ın kaplaycağı alan 400

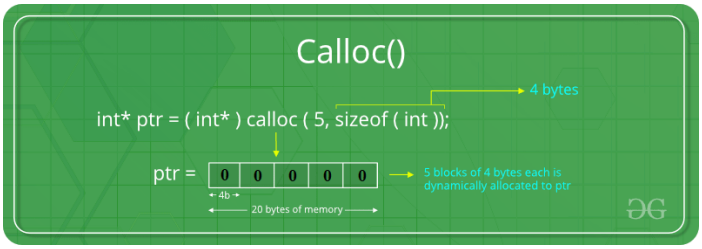
ü verdi. Ve sonuçta 100 indisli bir dizi oluşturmuş olduk. Birde diğer türlerde oluşturalım:

```
char *karakter; float *ptr;
karakter = (char*) malloc(100 * sizeof(char));
ptr = (float*) malloc(100 * sizeof(float));
```

2) calloc(): Bellekte alan ayırma ve bitlere 0 atama.

Calloc fonksiyonu mallco ta olduğu gibi alan ayırır ama malloc tan farklı olarak ayrdığı alanın indislerine(bitlerine/elemanlarına) 0 ı atar.

```
int *ptr;
ptr = (int *) calloc (100, sizeof (int));
```

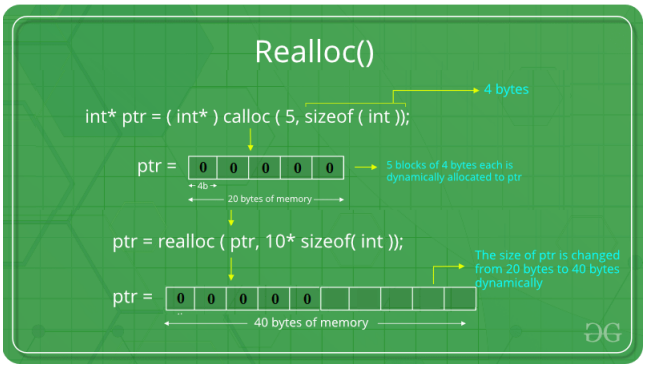


Burada aynı mallocta olduğu gibi dinamik dizimizi oluşturduk ama aynı zamanda bitlere 0 yazdı, malloc ise boştu(random). Malloc performans açısından calloc daha hızlıdır.

3) realloc(): Daha önce ayrılan belleğin boyutunu değiştirme.

Realloc ile daha önceden malloc veya calloc ile ayırdığımız alan yetersiz veya fazla ise sonradan bu alanın boyutunu yeniden boyutlandırmaya yarayan bir fonrksiyondur.

```
int *ptr;
ptr = (int *) calloc (5, sizeof (int));
ptr = realloc(ptr, 10* sizeof(int));
```



Bu kod ile daha onceden 5 elemanlı ve elemanlarında 0 bulunan diznin boyutunu 10 elemanlıya çıkardık. Ve bu yapılırken indislerde girili değerler değişmemiştir.

```
int *ptr;
ptr = (int *) calloc (15, sizeof (int));
ptr = realloc(ptr, 5* sizeof(int));
```

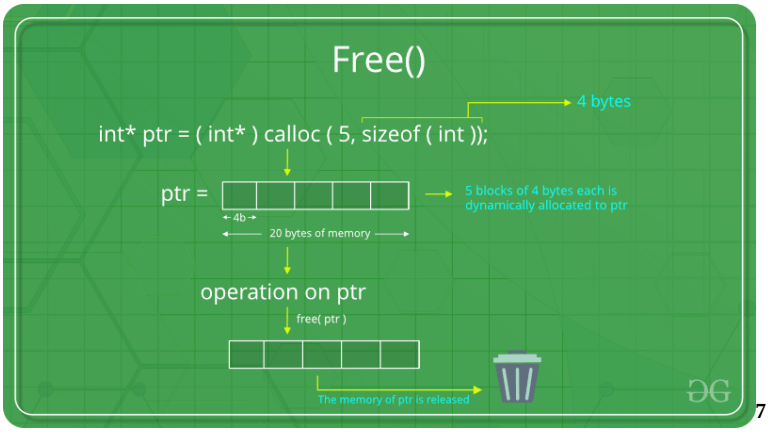
Bu kod ile de 15 elemanlı oluşturulmuş dizinin son 10 elemanı yok olmuştur yani yeni dizi 5elemanlı olarak yendien boyutlandı ve haliyle sonraki elemanlar kaybolmuştur.

4) free(): Daha önce ayrılan bellek alanını boşaltma.

Malloc veya calloc ile oluşturduğumuz alan bellekten kendisi silinmez bellekte hep kalırlar. C de buna karşı free() fonksiyonu ile malloc veya calloc ile ayrılan alanı temizler.

```
int *ptr;
ptr = (int *) malloc (5 * sizeof (int));
free(ptr);
```

Burada malloc ile bellekte alan açtık ve daha sonra bellekte kalmasın diye tasarruf etmek için geri bellekten temizledik. Malloc veya calloc ile oluşturduğumuz alanları mutlaka free() ile temizlemeliyiz yoksa ramimizde boşu boşuna alan işgal edilir.



[7]

C. ARASINDAKİ FARK - (int *)malloc(20) - (int *)malloc(sizeof(int)*5)

- Bunu anlamak için öncelikle **sizeof** ne işe yarar onu bi düşünelim, sizeof(veriTürü) komutu içine girdiğimiz veri türünün bizim sistemimizde kaç baytlık alan kapladığını bize söyler. Bu alan 32 bit sistemden 64 bit sistemlere göre farklılık gösterir.
- Malloc ise dinamik bellek yönetimi yapabilmemizi sağlar. Ve içine girdiğimiz malloc((veri tür'ün sistemde kapladığı alan)*istenilenEleman sayısı) kadar bize bellekte yer açar.

[7] Bu konuya ait görselleri adresinden aldım: https://www.geeksforgeeks.org/dynamic-memory-allocation-in-c-using-malloc-calloc-free-and-realloc/

Benim sistemimde intger 4 baytlık bir alan kaplamaktadır. Bu alan sistemden sisteme değişir. Şimdi benim sistemime göre yaptığımızı varsayalım:
Diyelim biz integer tipinde 5 elemanlık bir alan oluşturmak istiyoruz. İki yöntemi de deniyelim.
(int *)malloc(sizeof(int)*5) > yaparsak bana, benim sistemimde integer türünün kaç bayt ta tutulduğunu öğrenip 5 ile çarpip çıkan sonuc kadar alan oluşturur. Bu da 5*4 den 20 ye eşittir. Ben bu hesaplama yerine direk sonucu içine yazarsam ne olur?
(int *)malloc(20)
Ben bu şekilde içine sonucu yazarsam bu alan yönetimini sadece kendi sistemime göre dizayn etmiş olurum. Çünkü farklı sistemlerde integer 4 baytlık yer kaplamayabilir. Ya 4 ten buyuk ya da 4ten kucuk olabilir e bu durumda istediğim kadar alanı ayıramamış olurum ve dinamik bellek yönetimini gerçekleştiremem.

6. HAFTA ÖRNEKLER

6.1. Örnek: malloc(), calloc(), realloc(), free() ile ilgili genel örnek.

Bu örnek bu konu ile ilgili en genel örnek o yüzden daha fazla örnek koymaycağım. Bu örneğin içinde;

- malloc ve calloc ve realloc ile dinamik dizi oluşturma.
- Oluşturulan dizinin alan yönetimini sağlama fazla alanı silme veya az alanin otomatik duzeltilmesi.
- Ve oluşturulan dinamik dizinin boyutunu değiştirip dizi değerlerini kaybetmeden yeni alana yeni değerler girimesi
- Oluşturulan diziyi pointerlar ile tersten yazdirma
- Vb...

Örnek Ekran Çıktısı: 6_1_ornek.c

```
Hangi turde dinamik dizi olusturmak istersiniz:
[1] malloc
[2] calloc
Seciminiz: 1
Dizinizin tipi nasil olsun:
[1] integer
[2] char
[3] float
Seciminiz(1-3): 2

Boyutu(eleman sayisi) ne kadar olsun: 5

Metin mi girmek istersiniz Yoksa tek tek karakter mi girmek istesiniz? [1]Karakter [2]Metin
Seciminiz: 2
char tipinde dizinizin metninini giriniz: hamzacelik

Diziniz olusturulmustur nasil goruntulemek istersiniz:
[1] Duz
[2] Ters
Seciminiz: 2

Ters:
4. > a
3. > z
2. > m
1. > a
0. > h
Goruldugu uzere diziniz de alan sorunu olusmus!
Dizinizdeki dolu alan: 10 toplam alan: 5, fazlalik alanlari otomatik silinsin, yetersiz alanlar ise acilsin mi? [1]Evet [2]Hayir
Seciminiz: 1

Alan Ayarlandi. Yeni hali:
0. > h
1. > a
2. > m
3. > z
4. > a
5. >
6. > ▯
7. >
8. > i
9. > s


Dizinizin boyutunu degistirmek ister misin? [1] Evet [2]Hayir
Seciminiz: 1
Yeni alan boyutunu(indis/eleman sayisi) giriniz: 3

Yeni Alan Ayarlandi. Yeni hali:
0. > h
1. > a
2. > m
```

6_1_ornek.c

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.
4.
5.  int main()
6.  {
7.    int i,x,*ptrTam,n=0,nYeni=0,secim,secimDizi,secimTuru=0;
8.    char *ptrChr,bosluk;
9.    float *ptrFl;
10.
11.  //malloc veya calloc hangisi olusturacagimzia karar veriyoruz
12.  do
13.  {
14.    printf("\nHangi turde dinamik dizi olusturmak istersiniz:\n[1] malloc\n[2]
     calloc\nSeciminiz: ");
15.    scanf("%d",&secimTuru);
16.  }while(secimTuru!=1 && secimTuru !=2);
17.
18.
19.
20.    printf("Dizinizin tipi nasil olsun:");
21.  //olusturacagimiz dinamik diziyi sectiriyoruz
22.  do
23.  {
24.    printf("\n[1] integer\n[2] char\n[3] float\nSeciminiz(1-3): ");
25.    scanf("%d",&secimDizi);
26.  }while(secimDizi<1 && secimDizi >3);
27.  //dinamik dizinin boyutunu
28.  while(n<1)
29.  {
30.    printf("\nBoyutu(eleman sayisi) ne kadar olsun: ");
31.    scanf("%d",&n);
32.  }
33.  //secilen dinamik dizi turune gore dizi olusturacagiz
34.  switch (secimDizi)
35.  {
36.    case 1://integer bir dinamik dizi olsutuyuyoruz ve kullanıcıya duz veye tersten
     ekrana bastirabiliyor
37.
38.    if(secimTuru==1)//malloc veya calloc tipinde istenmisse ona gore dizilerimizide
     olsuturcagiz
39.    {
40.      ptrTam = (int *) malloc (n * sizeof(int));//dinaik dizimizi aldigimiz degerlere gore
     olusturduk.
```

```
41.    }else
42.    {
43.      ptrTam = (int *) calloc (n, sizeof(int));//dinaik dizimizi aldigimiz degerlere gore
       olusturduk.
44.    }
45.
46.
47.
48.      if(ptrTam!=NULL)//bellekte yeteri kadar alan ayrilirsa dizimizin icini doldurmaya
       basliyoruz
49.      {
50.        for(i=0;i<n;i++)
51.        {
52.          printf("Integer tipinde dizinizin %d. indisini giriniz: ",i);
53.          scanf("%d",ptrTam+i);
54.          //ptrTam++;
55.        }
56.        //diziyi doldurduktan sonra ekrana bastirma seklini alip ona gore dizi ekrana
       basiliyor
57.        printf("\nDiziniz olusturulmustur nasil goruntulemek istersiniz:\n[1] Duz\n[2]
       Ters\nSeciminiz: ");
58.        scanf("%d",&secim);
59.        //diziyi secime gore ekrana bastirma
60.        switch (secim)
61.        {
62.          default:
63.            printf("\nYanlis secim o halde:");
64.          case 1:
65.            printf("\nDuz:");
66.            for(i=0;i<n;i++)
67.            {
68.              printf("\n%d. > %d",i,*(ptrTam+i));
69.            }
70.            break;
71.          case 2:
72.            printf("\nTers:");
73.            for(i=n-1;i>=0;i--)
74.            {
75.              printf("\n%d. > %d",i,*(ptrTam+i));
76.            }
77.          break;
78.        }
79.
80.
81.
82.      }else
83.      {
84.        printf("\nBellekte o kadar alan ayrilamadi!");
```

```
85.       }
86.       break;
87.     case 2://boyutunu almistik, simdi ise char dizisi olusturcagiz
88.       if(secimTuru==1)//malloc veya calloc tipinde istenmisse ona gore dizilerimizide
      olsuturcagiz
89.         {
90.             ptrChr = (char *) malloc (n * sizeof(char));//dinaik dizimizi aldigimiz degerlere
      gore olusturduk.
91.         }else
92.         {
93.           ptrChr = (char *) calloc (n, sizeof(char));//dinaik dizimizi aldigimiz degerlere
      gore olusturduk.
94.         }
95.
96.
97.       if(ptrChr!=NULL)//bellekte yeteri kadar alan ayrilirsa dizimizin icini doldurmaya
      basliyoruz
98.       {
99.
100.                printf("\nMetin mi girmek istersiniz Yoksa tek tek karakter mi girmek
      istesiniz? [1]Karakter [2]Metin\nSeciminiz: ");
101.                scanf("%d",&secim);
102.                scanf("%c",&bosluk);//char oldugu icin ustte bir deger alinca enter
      basinca bir sonraki scanf e enteri(bosluk degerini) basiyor o yuzden enteri boslugu
      oylesine bir char a aldiriyorum her seferinde
103.                if(secim!=2)//tek tek karakter girisi secildiigi icin ona gore alicaz
104.                {
105.                  for(i=0;i<n;i++)
106.                  {
107.                    printf("char tipinde dizinizin %d. indisini giriniz: ",i);
108.                    scanf("%c",ptrChr+i);
109.                    scanf("%c",&bosluk);//char oldugu icin ustte bir deger alinca enter
      basinca bir sonraki scanf e enteri(bosluk degerini) basiyor o yuzden enteri boslugu
      oylesine bir char a aldiriyorum her seferinde
110.                    //ptrTam++;
111.                  }
112.                }else if(secim==2)
113.                {
114.                  printf("char tipinde dizinizin metninini giriniz: ",i);
115.                  gets(ptrChr);
116.                }
117.
118.                  //diziyi doldurduktan sonra ekrana bastirma seklini alip ona gore dizi
      ekrana basiliyor
119.                  printf("\nDiziniz olusturulmustur nasil goruntulemek istersiniz:\n[1]
      Duz\n[2] Ters\nSeciminiz: ");
120.                  scanf("%d",&secim);
121.                  //diziyi secime gore ekrana bastirma
```

```
122.                 switch (secim)
123.                 {
124.                   default:
125.                     printf("\nYanlis secim o halde:");
126.                   case 1:
127.                     printf("\nDuz:");
128.                     for(i=0;i<n;i++)
129.                     {
130.                       printf("\n%d. > %c",i,*(ptrChr+i));
131.                     }
132.                     break;
133.                   case 2:
134.                     printf("\nTers:");
135.                     for(i=n-1;i>=0;i--)
136.                     {
137.                       printf("\n%d. > %c",i,*(ptrChr+i));
138.                     }
139.                   break;
140.                 }
141.                 //simdi ise kullanicidan girilen alanin boyutu ile beasta diziye atadigi
    boyutu kontrol edecegiz ve alan fazla veya az ona gore islem yaptircam
142.                 for(i=0;*(ptrChr+i)!='\0';i++);//kullanicinin doldurdugu alani i ile tespit
    ediyorum
143.                 if(n!=i)//basta dizi icin olsutrudugu alan eger doldurgu alan ile ayni
    degil ise ralloc ile otomatik duzeltipi duzletmek istemedigini soruyorum
144.                 {
145.                   printf("\nGoruldugu uzere diziniz de alan sorunu olusmus!");
146.                   printf("\nDizinizdeki dolu alan: %d toplam alan: %d, fazlalik alanlari
    otomatik silinsin, yetersiz alanlar ise acilsin mi? [1]Evet [2]Hayir\nSeciminiz: ",i,n);
147.                   scanf("%d",&secim);
148.                   if(secim==1)
149.                   {
150.                     n=i;
151.                     ptrChr=realloc(ptrChr,n*sizeof(char));
152.                     if(ptrChr!=NULL)//bellekte yeteri kadar alan ayrilirsa dizimizin icini
    doldurmaya basliyoruz
153.                     {
154.                       printf("\nAlan Ayarlandi. Yeni hali:");
155.                       for(i=0;i<n;i++)
156.                       {
157.                         printf("\n%d. > %c",i,*(ptrChr+i));
158.                       }
159.                     }else{printf("\nBellek hafizasi yetrsiz geldi!");}
160.                   }else{printf("\nAnlasildi islemler bitmistir...");}
161.                 }
162.
163.               }else
164.               {
```

```
165.                 printf("\nBellekte o kadar alan ayrilamadi!");
166.               }
167.               break;
168.
169.
170.           case 3://float bir dinamik dizi olsutuyuyoruz ve kullanıcıya duz veye tersten
    ekrana bastirabiliyor
171.               if(secimTuru==1)//malloc veya calloc tipinde istenmisse ona gore
    dizilerimizide olsuturcagiz
172.               {
173.                 ptrFl = (float *) malloc (n * sizeof(float));//dinaik dizimizi aldigimiz
    degerlere gore olusturduk.
174.               }else
175.               {
176.                 ptrFl = (float *) calloc (n, sizeof(float));//dinaik dizimizi aldigimiz
    degerlere gore olusturduk.
177.               }
178.
179.
180.               if(ptrFl!=NULL)//bellekte yeteri kadar alan ayrilirsa dizimizin icini
    doldurmaya basliyoruz
181.               {
182.                 for(i=0;i<n;i++)
183.                 {
184.                   printf("Float tipinde dizinizin %d. indisini giriniz: ",i);
185.                   scanf("%f",ptrFl+i);
186.
187.                 }
188.                 //diziyi doldurduktan sonra ekrana bastirma seklini alip ona gore dizi
    ekrana basiliyor
189.                 printf("\nDiziniz olusturulmustur nasil goruntulemek istersiniz:\n[1]
    Duz\n[2] Ters\nSeciminiz: ");
190.                 scanf("%d",&secim);
191.                 //diziyi secime gore ekrana bastirma
192.                 switch (secim)
193.                 {
194.                   default:
195.                     printf("\nYanlis secim o halde:");
196.                   case 1:
197.                     printf("\nDuz:");
198.                     for(i=0;i<n;i++)
199.                     {
200.                       printf("\n%d. > %0.2f",i,*(ptrFl+i));
201.                     }
202.                     break;
203.                   case 2:
204.                     printf("\nTers:");
205.                     for(i=n-1;i>=0;i--)
```

```
206.                     {
207.                       printf("\n%d. > %0.2f",i,*(ptrFl+i));
208.                     }
209.                   break;
210.                 }
211.
212.
213.
214.               }else
215.               {
216.                 printf("\nBellekte o kadar alan ayrilamadi!");
217.               }
218.               break;
219.             case 4:
220.             break;
221.             default:
222.             printf("\nGecersiz secim yaptnız 1-4 arası secilmeliydi!");
223.           }
224.
225.
226.           //ekrana basma islemi bittiktekn sonra diziyi kucultmek veya buyutlmek
    isteyip istemedigini soruyoruz
227.           printf("\n\nDizinizin boyutunu degistirmek ister misin? [1] Evet
    [2]Hayir\nSeciminiz: ");
228.           scanf("%d",&secim );
229.           switch (secim)
230.           {
231.             default:
232.               printf("\nYanlis secim o halde:");
233.             case 2:
234.               printf("\nDiziyi yeniden boyutlandirma yapmiyoruz...");
235.               break;
236.             case 1:
237.                 switch (secimDizi)
238.                 {
239.                   case 1://integer tipinde olsuturlmus olan diziye yeni boyut vercegiz
240.                       while(nYeni<1)
241.                       {
242.                         printf("Yeni boyutu(eleman sayisi/indis) ne kadar olsun: ");
243.                         scanf("%d",&nYeni);
244.                       }
245.                       //dinamik diziyi yendiden boyutlandirmamizi yaptik.
246.                       ptrTam = realloc(ptrTam, nYeni* sizeof(int));
247.                       if(ptrTam!=NULL)//bellekte yeteri kadar alan ayrilirsa dizimizin icini
    doldurmaya basliyoruz
248.                       {
249.                         if(nYeni>n)printf("\nEski dizinize %d tane indis daha
    eklenmistir.",nYeni-n);
```

```
250.                    else printf("Eski diziniz kcultulmustur ve son %d indis gitmistir.",n-
    nYeni);
251.                    //yeni boyutlu diziyi kullanicinin nasil dolduracagini ogreniyoruz.
252.                    printf("\nYeni dizinizi nasil doldurmak istesiniz: \n[1]Eski dizi
    degerlerini kaybetmeden yanina\n[2]Eski dizi degerlerininde ustune\n[3]Dizi ye bir
    sey eklemeyecegim oldugu gibi kalacak.");
253.                    printf("\nSeciminiz: ");
254.                    scanf("%d",&secim);
255.                    switch (secim)
256.                    {
257.                      case 1:
258.                        printf("\nEski boyut %d. indisinden eklemeye devam
    ediyorsunuz.",(nYeni<n) ? nYeni:n);
259.                        for(i=n;i<nYeni;i++)
260.                        {
261.                          printf("\nYeni dizinizin %d. indisini giriniz: ",i);
262.                          scanf("%d",ptrTam+i);
263.                        }
264.                        printf("\nYeni diziniz:");
265.                        for(i=0;i<nYeni;i++)
266.                        {
267.                          printf("\n%d. > %d",i,*(ptrTam+i));
268.                        }
269.                        break;
270.                        case 2:
271.                        printf("\nYeni degerleri giriniz.");
272.                        for(i=0;i<nYeni;i++)
273.                        {
274.                          printf("\nYeni dizinizin %d. indisini giriniz: ",i);
275.                          scanf("%d",ptrTam+i);
276.                        }
277.                        printf("\nYeni diziniz:");
278.                        for(i=0;i<nYeni;i++)
279.                        {
280.                          printf("\n%d. > %d",i,*(ptrTam+i));
281.                        }
282.                        break;
283.                        default:
284.                          printf("\nHatali Secim!");
285.                        case 3:
286.                        printf("\nO halde yeni diziniz oldugu gibi kalmistir. Yeni Dizi:");
287.                        for(i=0;i<nYeni;i++)
288.                        {
289.                          printf("\n%d. > %d",i,*(ptrTam+i));
290.                        }
291.                        break;
292.                      }
293.                    }else{printf("Bellekte yeterli alan ayrilamadi!");}
```

```
294.
295.                   break;
296.                 case 2:
297.
298.                   printf("Yeni alan boyutunu(indis/eleman sayisi) giriniz: ");
299.                   scanf("%d",&nYeni);
300.
301.                   ptrChr=realloc(ptrChr,nYeni*sizeof(char));
302.                   if(ptrChr!=NULL)//bellekte yeteri kadar alan ayrilirsa dizimizin icini
    doldurmaya basliyoruz
303.                   {
304.                     printf("\nYeni Alan Ayarlandi. Yeni hali:");
305.                     for(i=0;i<nYeni;i++)
306.                     {
307.                       printf("\n%d. > %c",i,*(ptrChr+i));
308.                     }
309.                   }else{printf("\nBellek hafizasi yetrsiz geldi!");}
310.                   break;
311.                 case 3:
312.                 while(nYeni<1)
313.                 {
314.                   printf("Yeni boyutu(eleman sayisi/indis) ne kadar olsun: ");
315.                   scanf("%d",&nYeni);
316.                 }
317.                 //dinamik diziyi yendiden boyutlandirmamizi yaptik.
318.                 ptrFl = realloc(ptrFl, nYeni* sizeof(float));
319.                 if(ptrFl!=NULL)//bellekte yeteri kadar alan ayrilirsa dizimizin icini
    doldurmaya basliyoruz
320.                 {
321.                   if(nYeni>n)printf("\nEski dizinize %d tane indis daha
    eklenmistir.",nYeni-n);
322.                   else printf("Eski diziniz kcultulmustur ve son %d indis gitmistir.",n-
    nYeni);
323.                   //yeni boyutlu diziyi kullanicinin nasil dolduracagini ogreniyoruz.
324.                   printf("\nYeni dizinizi nasil doldurmak istesiniz: \n[1]Eski dizi
    degerlerini kaybetmeden yanina\n[2]Eski dizi degerlerininde ustune\n[3]Dizi ye bir
    sey eklemeyecegim oldugu gibi kalacak.");
325.                   printf("\nSeciminiz: ");
326.                   scanf("%d",&secim);
327.                   switch (secim)
328.                   {
329.                     case 1:
330.                       printf("\nEski boyut %d. indisinden eklemeye devam
    ediyorsunuz.",(nYeni<n) ? nYeni:n);
331.                       for(i=n;i<nYeni;i++)
332.                       {
333.                         printf("\nYeni dizinizin %d. indisini giriniz: ",i);
334.                         scanf("%f",ptrFl+i);
```

```
335.                    }
336.                    printf("\nYeni diziniz:");
337.                    for(i=0;i<nYeni;i++)
338.                    {
339.                      printf("\n%d. > %0.2f",i,*(ptrFl+i));
340.                    }
341.                    break;
342.                    case 2:
343.                    printf("\nYeni degerleri giriniz.");
344.                    for(i=0;i<nYeni;i++)
345.                    {
346.                      printf("\nYeni dizinizin %d. indisini giriniz: ",i);
347.                      scanf("%f",ptrFl+i);
348.                    }
349.                    printf("\nYeni diziniz:");
350.                    for(i=0;i<nYeni;i++)
351.                    {
352.                      printf("\n%d. > %0.2f",i,*(ptrFl+i));
353.                    }
354.                    break;
355.                    default:
356.                      printf("\nHatali Secim!");
357.                    case 3:
358.                    printf("\nO halde yeni diziniz oldugu gibi kalmistir. Yeni Dizi:");
359.                    for(i=0;i<nYeni;i++)
360.                    {
361.                      printf("\n%d. > %0.2f",i,*(ptrFl+i));
362.                    }
363.                    break;
364.                  }
365.                 }else{printf("Bellekte yeterli alan ayrilamadi!");}
366.
367.                 break;
368.                 case 4: break;
369.                 default: break;
370.               }
371.             }
372.
373.
374.
375.           free(ptrTam);//olusuturlan alanı program isi btince siliyoruz
376.           return 0;
377.         }
```

ÇIKTI:

```
Hangi turde dinamik dizi olusturmak istersiniz:
[1] malloc
[2] calloc
```

```
Seciminiz: 1
Dizinizin tipi nasil olsun:
[1] integer
[2] char
[3] float
Seciminiz(1-3): 1

Boyutu(eleman sayisi) ne kadar olsun: 5
Integer tipinde dizinizin 0. indisini giriniz: 14
Integer tipinde dizinizin 1. indisini giriniz: 13
Integer tipinde dizinizin 2. indisini giriniz: 12
Integer tipinde dizinizin 3. indisini giriniz: 11
Integer tipinde dizinizin 4. indisini giriniz: 10

Diziniz olusturulmustur nasil goruntulemek istersiniz:
[1] Duz
[2] Ters
Seciminiz: 2

Ters:
4. > 10
3. > 11
2. > 12
1. > 13
0. > 14

Dizinizin boyutunu degistirmek ister misin? [1] Evet [2]Hayir
Seciminiz: 1
Yeni boyutu(eleman sayisi/indis) ne kadar olsun: 6

Eski dizinize 1 tane indis daha eklenmistir.
Yeni dizinizi nasil doldurmak istesiniz:
[1]Eski dizi degerlerini kaybetmeden yanina
[2]Eski dizi degerlerininde ustune
[3]Dizi ye bir sey eklemeyecegim oldugu gibi kalacak.
Seciminiz: 1

Eski boyut 5. indisinden eklemeye devam ediyorsunuz.
Yeni dizinizin 5. indisini giriniz: 0

Yeni diziniz:
0. > 14
1. > 13
2. > 12
3. > 11
4. > 10
```

```
5. > 0
```

6.1. Örnek: Disaridan girilen n kadar dinamik char dizisi olusturup, kullanicidan gets() ile icine metin atayip, girilen metin ile ilgili alan yonetiminin yapilmasi disardan dinamik diziye girilen alan dizinin olsutugu alandan kucuk veya fazla ise bunu otomatik duzeltme.

6_1_ornek.c

Ornek 6_1_ornek.c de bu işlev zaten mevcut ekran çıktısı:

ÇIKTI:

```
Hangi turde dinamik dizi olusturmak istersiniz:
[1] malloc
[2] calloc
Seciminiz: 1
Dizinizin tipi nasil olsun:
[1] integer
[2] char
[3] double
[4] float
Seciminiz(1-4): 2

Boyutu(eleman sayisi) ne kadar olsun: 10

Metin mi girmek istersiniz Yoksa tek tek karakter mi girmek istesiniz? [1]Karakter
[2]Metin
Seciminiz: 2
char tipinde malloc dizinizin metninini giriniz: Hamza Celik

Diziniz olusturulmustur nasil goruntulemek istersiniz:
[1] Duz
[2] Ters
Seciminiz: 2

Ters:
9. > i
8. > l
7. > e
6. > C
5. >
```

```
4. > a
3. > z
2. > m
1. > a
0. > H
Goruldugu uzere diziniz de alan sorunu olusmus!
Dizinizdeki dolu alan: 11 toplam alan: 10, fazlalik alanlari otomatik silinsin, yetersiz
alanlar ise acilsin mi? [1]Evet [2]Hayir
Seciminiz: 1

Alan Ayarlandi. Yeni hali:
0. > H
1. > a
2. > m
3. > z
4. > a
5. >
6. > C
7. > e
8. > l
9. > i
10. > k

Dizinizin boyutunu degistirmek ister misin? [1] Evet [2]Hayir
Seciminiz: 1
Yeni alan boyutunu(indis/eleman sayisi) giriniz: 5

Yeni Alan Ayarlandi. Yeni hali:
0. > H
1. > a
2. > m
3. > z
4. > a
```

6.1. Örnek: Disaridan alinan n boyutluk float tipinde malloc ile bir dinamik dizi olsuturnuz ve diziye deger atayiniz ardindan dizide realloc ile alan degisikligi yapiniz.

6_1_ornek.c
Ornek 6_1_ornek.c de bu işlev zaten mevcut ekran çıktısı:

ÇIKTI:

```
Hangi turde dinamik dizi olusturmak istersiniz:
[1] malloc
[2] calloc
Seciminiz: 1
Dizinizin tipi nasil olsun:
[1] integer
[2] char
[3] float
[4] double
Seciminiz(1-4): 3

Boyutu(eleman sayisi) ne kadar olsun: 5
Float tipinde malloc dizinizin 0. indisini giriniz: 5.2
Float tipinde malloc dizinizin 1. indisini giriniz: 7.85
Float tipinde malloc dizinizin 2. indisini giriniz: 3.74
Float tipinde malloc dizinizin 3. indisini giriniz: 1.0
Float tipinde malloc dizinizin 4. indisini giriniz: 99

Diziniz olusturulmustur nasil goruntulemek istersiniz:
[1] Duz
[2] Ters
Seciminiz: 1

Duz:
0. > 5.20
1. > 7.85
2. > 3.74
3. > 1.00
4. > 99.00

Dizinizin boyutunu degistirmek ister misin? [1] Evet [2]Hayir
Seciminiz: 1
Yeni boyutu(eleman sayisi/indis) ne kadar olsun: 3
Eski diziniz kcultulmustur ve son 2 indis gitmistir.
Yeni dizinizi nasil doldurmak istesiniz:
[1]Eski dizi degerlerini kaybetmeden yanina
[2]Eski dizi degerlerininde ustune
[3]Dizi ye bir sey eklemeyecegim oldugu gibi kalacak.
Seciminiz: 2

Yeni degerleri giriniz.
Yeni dizinizin 0. indisini giriniz: 87

Yeni dizinizin 1. indisini giriniz: 1.5
```

```
Yeni dizinizin 2. indisini giriniz: 32.5

Yeni diziniz:
0. > 87.00
1. > 1.50
2. > 32.50
```

6.1. Örnek: Disaridan alinan n adet boyutta ineteger dinamik dizi oluşturup, dizi içine dışardan değer atanıp, daha sonra boyutu artırılıp, yeni boyuta gore eski diziyi kaybetmeden eski dizi kayıtlarının yanına yeni integer veri girişi.

6_1_ornek.c

Ornek 6_1_ornek.c de bu işlev zaten mevcut ekran çıktısı:

ÇIKTI:

```
Hangi turde dinamik dizi olusturmak istersiniz:
[1] malloc
[2] calloc
Seciminiz: 1
Dizinizin tipi nasil olsun:
[1] integer
[2] char
[3] float
Seciminiz(1-3): 1

Boyutu(eleman sayisi) ne kadar olsun: 5
Integer tipinde malloc dizinizin 0. indisini giriniz: 15
Integer tipinde malloc dizinizin 1. indisini giriniz: 17
Integer tipinde malloc dizinizin 2. indisini giriniz: 19
Integer tipinde malloc dizinizin 3. indisini giriniz: 20
Integer tipinde malloc dizinizin 4. indisini giriniz: 14

Diziniz olusturulmustur nasil goruntulemek istersiniz:
[1] Duz
[2] Ters
Seciminiz: 2

Ters:
4. > 14
3. > 20
2. > 19
```

```
1. > 17
0. > 15

Dizinizin boyutunu degistirmek ister misin? [1] Evet [2]Hayir
Seciminiz: 1
Yeni boyutu(eleman sayisi/indis) ne kadar olsun: 10

Eski dizinize 5 tane indis daha eklenmistir.
Yeni dizinizi nasil doldurmak istesiniz:
[1]Eski dizi degerlerini kaybetmeden yanina
[2]Eski dizi degerlerininde ustune
[3]Dizi ye bir sey eklemeyecegim oldugu gibi kalacak.
Seciminiz: 1

Eski boyut 5. indisinden eklemeye devam ediyorsunuz.
Yeni dizinizin 5. indisini giriniz: 1

Yeni dizinizin 6. indisini giriniz: 2

Yeni dizinizin 7. indisini giriniz: 3

Yeni dizinizin 8. indisini giriniz: 4

Yeni dizinizin 9. indisini giriniz: 5

Yeni diziniz:
0. > 15
1. > 17
2. > 19
3. > 20
4. > 14
5. > 1
6. > 2
7. > 3
8. > 4
9. > 5
```

6.2. Örnek: Malloc ve Calloc Farkı

6_2_ornek.c

```
1. #include <stdio.h>
2. #include<stdlib.h>
3. int main()
```

```
4.  {
5.    int *ptrMallok, *ptrCallok, i,n;
6.
7.    printf("Boyutu(eleman/indis miktarini) giriniz: ");
8.    scanf("%d",&n);
9.    ptrCallok = (int *) calloc (n, sizeof(int));
10.   if(ptrCallok!=NULL)
11.   {
12.     printf("Callok alani olusturuldu.");
13.     for(i=0;i<n;i++)
14.     {
15.       printf("\nCallok - %d.Indis degeri %d adresi
      %p",i,*(ptrCallok+i),ptrCallok+i);
16.     }
17.   }else printf("Callok alani olusturulumadi yetersiz bellek.");
18.
19.
20.
21.   ptrMallok = (int *) malloc (n * sizeof(int));
22.   if(ptrMallok!=NULL)
23.   {
24.     printf("\n\n\nMallok alani olusturuldu.");
25.     for(i=0;i<n;i++)
26.     {
27.       printf("\nMalloc - %d.Indis degeri %d adresi
      %p",i,*(ptrMallok+i),ptrMallok+i);
28.     }
29.   }else printf("Mallok alani olusturulumadi yetersiz bellek.");
30.   return 0;
31. }
```

ÇIKTI:

```
Boyutu(eleman/indis miktarini) giriniz: 2
Callok alani olusturuldu.
Callok - 0.Indis degeri 0 adresi 006815F8
Callok - 1.Indis degeri 0 adresi 006815FC

Mallok alani olusturuldu.
Malloc - 0.Indis degeri 6828904 adresi 006815A8
Malloc - 1.Indis degeri 6815936 adresi 006815AC
```

6.3. Örnek:

6_#_ornek.c
32.
ÇIKTI:

7. HAFTA

A. STRİNGLER

C de stringleri char tipinde tanımlabiliyoruz. Karakterler ile stringler arasındaki fark karakterler tektek karakterden oluşuyor ama stringler ise karakterlerin birleşmesi ile oluşur, diğer bir fark ise stringler '\0' karakteri ile sonlanır. '\0' bu özel karakter bir dizenin stringin yani yaznın sonuna getirilir ve o stringin sonunu ifade eder.

Örneğin bir string ifade aşağıdaki şekilde tanımlayabiliriz:

```
char str[]  = "hamza";
char yazi[5] = {'h','a','m','z','a'};
char *strp = "hamza";
```

0	1	2	3	4	5
h	**a**	**m**	**z**	**a**	**\0**

Char dizisine atanmış şekli tabloadaki gibidir. Bir dizinin sonlandığını \0 ile öğreniriz.

```
char str[5] = {'h','a','m','z','a'};
for(int i=0;str[i]!='\0';i++)
        printf("%c",str[i]);
//CIKTI: hamza

char *ogrenciler[3]={"hamza","ahmet","ali"};
  for(int i=0;i<3;i++)
  {

    printf("%d - %s\t",i+1,ogrenciler[i]);

  }
//CIKTI: 1 - hamza       2 - ahmet       3 - ali
```

Seklinde de bastırabiliyorduk. Gelelim string fonksiyonlarına:

B. STRİNG.H FONKSİYONLARI

(BU TABLOLAR HAZIR KAYNAĞI: https://www.bilgigunlugum.net/prog/cprog/c_stdkut/string)

Fonksiyon adı	Açıklama
memccpy	**void* memccpy(void *dest, const void *src, int c, size_t n);** Üçüncü parametre ile gösterilen değer kopyalanana veya dördüncü parametredeki değer kadar, karakter kopyalanana kadar, ikinci parametrede gösterilen bellek alanını ilk parametre ile gösterilen bellek alanına kopyalar.
memchr	**void* memchr(const void *ptr, int c, size_t n);** İlk parametrede gösterilen bellek bölgesinde, üçüncü parametrede gösterilen değer kadar ilk byte değeri içinde ikinci parametrede gösterilen unsigned char değeri arar.
memcmp	**int memcmp(const void *ptr1, const void *ptr2, size_t n);** İlk ve ikinci parametrede gösterilen bellek bölgelerinin, üçüncü parametrede gösterilen değer kadar, ilk byte değerlerini karşılaştırır.
memcpy memcpy_s	**void* memcpy(void *dest, const void *src, size_t n);** İkinci parametredeki bellek bölgesinden, üçüncü parametrede gösterilen değer kadar, byte değerini ilk parametredeki bellek adresine kopyalar.
memmove memmove_s	**void* memmove(void *dest, const void *src, size_t n);** İkinci parametredeki bellek bölgesindeki değerleri, üçüncü parametrede gösterilen değer kadar, ilk parametredeki bellek adresine taşır.
memset memset_s	**void* memset(void *dest, int c, size_t n);** İkinci parametredeki unsigned char değeri ilk parametredeki karakter dizisinin, üçüncü parametrede gösterilen değer kadar, ilk byte değeri üzerine kopyalar.
strcat strcat_s	**char* strcat(char *dest, const char *src);** İkinci parametredeki karakter dizisini ilk parametrede gösterilen karakter dizisinin sonuna ekler.
strchr	**char* strchr(const char *str, int c);** İkinci parametredeki unsigned char değerinin ilk parametrede gösterilen karakter dizisinin içinde bulunduğu ilk yeri arar.
strcmp	**int strcmp(const char *str1, const char *str2);**

	İlk ve ikinci parametredeki karakter dizilerini eşitlik durumunu belirlemek için birbiriyle karşılaştırır.
strcoll	**int strcoll(const char *str1, const char *str2);** İlk ve ikinci parametredeki karakter dizilerini birbiriyle eşitlik durumunu, yerel LC_COLLATE ayarlarına göre, belirlemek için karşılaştırır.
strcpy strcpy_s	**char* strcpy(char *dest, const char *src);** İkinci parametredeki karakter dizisini ilk parametrede gösterilen karakter dizisine kopyalar.
strcspn	**size_t strcspn(const char *str1, const char *str2);** İlk parametrede gösterilen karakter dizisinde, tamamı ikinci parametrede gösterilen karakter dizisi içinde olmayan karakterlerden oluşan, ilk bölümün uzunluğunu hesaplar.
strdup	**char* strdup(const char *str);** Kendisine geçirilen parametrede gösterilen karakter dizisinin aynısını kopyalarak geri döndürür.
strerror strerror_s strerrorlen_s	**char* strerror(int errnum);** Kendisine geçirilen parametredeki hata kodunun metin ifadesini geri döndürür.
strlen strnlen_s	**size_t strlen(const char *str);** Kendisine geçirilen parametredeki karakter dizisinin uzunluğunu geri döndürür.
strncat strncat_s	**char* strncat(char *dest, const char *src, size_t n);** İkinci parametredeki karakter dizisinin, üçüncü parametrede gösterilen değer kadar ilk byte değerini ilk parametrede gösterilen karakter dizisinin sonuna ekler.
strncmp	**int strcmp(const char *str1, const char *str2, size_t n);** İlk ve ikinci parametredeki karakter dizilerinin, üçüncü parametrede gösterilen değer kadar, ilk karakterini birbiriyle eşitlik durumunu belirlemek için karşılaştırır.
strncpy strncpy_s	**char* strncpy(char *dest, const char *src, size_t n);** İkinci parametredeki karakter dizisinin, üçüncü parametrede gösterilen değer kadar, ilk byte değerini ilk parametrede gösterilen karakter dizisine kopyalar.

strndup	**char* strndup(const char *str, size_t n);** İlk parametrede gösterilen karakter dizisinin, ikinci parametrede gösterilen değer kadar, ilk byte değerinin aynısını kopyalarak geri döndürür.
strpbrk	**char* strpbrk(const char *str1, const char *str2);** İkinci parametredeki karakter dizisi içinde yer alan karakterlerden herhangi birinin ilk parametre ile gösterilen karakter dizisi içinde bulunduğu ilk yerin adresini geri döndürür.
strrchr	**char* strrchr(const char *str, int c);** İkinci parametredeki unsigned char değerinin ilk parametrede gösterilen karakter dizisinin içinde bulunduğu son yeri arar.
strspn	**size_t strspn(const char *str1, const char *str2);** Tamamı ikinci parametrede gösterilen karakter dizisi içinde olan karakterlerden oluşan ilk parametrede gösterilen karakter dizisinin ilk bölümünün uzunluğunu hesaplar.
strstr	**char* strstr(const char *str1, const char *str2);** İkinci parametrede gösterilen karakter dizisinin ilk parametrede gösterilen karakter dizisi içindeki ilk yerinin bellek adresini bulur.
strtok strtok_s	**char* strtok(char *str, const char *delim);** İlk parametredeki karakter dizisini ikinci parametredeki ayırıcı karakterlerin bulunduğu yerlerden alt karakter dizilerine ayırır.
strxfrm	**size_t strxfrm(char *dest, const char *src, size_t n);** İkinci parametrede gösterilen karakter dizisini aktif lokal ayarlara göre çevirerek, üçüncü parametredeki değer kadar ilk karakterini, ilk parametre ile gösterilen karakter dizisine kopyalar.

Veriler

Veri türü adı	Açıklama
FILE	Bir dosya ile ilgili giriş veya çıkış işlemleri yapmak için, dosya veya akış hakkında bilgiler içeren bir veridir.
fpos_t	Dosyanın konumunu göstermek için kullanılan long veya long olarak tanımlanan bir veri türüdür.
size_t	Unsigned integer olarak tanımlanan bir veri türüdür.

Makro değişkenler

Değişken adı	Açıklama
stdin	Standart giriş akışını gösteren bir işaretçi tanımlayan bir makrodur.

stdout	Standart çıkış akışını gösteren bir işaretçi tanımlayan bir makrodur.
stdin	Standart hata akışını gösteren bir işaretçi tanımlayan bir makrodur.

Makro sabitler

Sabit adı	Açıklama
BUFSIZ	Setbuf() fonksiyonu tarafından kullanılan arabelleğin boyutunu belirleyen bir int değeridir.
EOF	Dosya sonu durumunu belirten bir negatif int bir değerdir.
FILENAME_MAX	Açılan bir dosya adını yüklemek için kullanılacak bir char dizinin azami boyutunu belirler.
FOPEN_MAX	Eşzamanlı olarak açılabilecek dosya sayısını gösterir.
_IOFBF	Bir kısaltma (Giriş/Çıkış Tamamen Tamponlu - Input/Output Fully Buffered) olup 0x0000 değerini taşır. Açık bir akış için tamamen tamponlanmış giriş ve çıkış talebi için setvbuf() fonksiyonuna geçirilebilecek bir tamsayıdır.
_IOLBF	Bir kısaltma (Giriş/Çıkış Satır Tamponlu - Input/Output Line Buffered) olup 0x0040 değerini taşır. Açık bir akış için satır olarak tamponlanmış giriş ve çıkış talebi için setvbuf() fonksiyonuna geçirilebilecek bir tamsayıdır.
_IONBF	Bir kısaltma (Giriş/Çıkış Tamponlanmamış - Input/Output Not Buffered) olup 0x0004 değerini taşır. Açık bir akış için tamponlanmamış giriş ve çıkış talebi için setvbuf() fonksiyonuna geçirilebilecek bir tamsayıdır.
L_tmpnam	Tmpnam() fonksiyonu tarafından oluşturulan geçici bir dosya adını yüklemek için kullanılan bir char dizinin boyutunu gösterir.
NULL	Boş işaretçi sabiti olan bir makrodur.
SEEK_CUR	Geçerli dosya konumuna göre konumlandırma talebi için fseek() fonksiyonuna geçirilebilecek bir int değerdir.
SEEK_END	Dosya sonuna konumuna göre konumlandırma talebi için fseek() fonksiyonuna geçirilebilecek bir int değerdir.
SEEK_SET	Dosya başına göre konumlandırma talebi için fseek() fonksiyonuna geçirilebilecek bir int değerdir.
TMP_MAX	Tmpnam() fonksiyonu tarafından oluşturulan azami benzersiz dosya adı sayısını gösterir.

7. HAFTA ÖRNEKLER

7.2. Örnek: Kütüphane kullanmadan string bir dize tanımlayıp bastırın.

7_1_ornek.c

```
1.  #include<stdio.h>
2.  #include <string.h>
3.  int main()
4.  {
5.    char *ogrenciler[3]={"hamza","ahmet","ali"};
6.    for(int i=0;i<3;i++)
7.    {
8.      printf("%d - %s\t",i+1,ogrenciler[i]);
9.    }
10.
11.  return 0;
12. }
```

ÇIKTI:

```
1 - hamza       2 - ahmet       3 - ali
```

7.3. Örnek: strcat(),strcpy(),strcmp() kullanimina ornek.

7_2_ornek.c

```
54. #include<stdio.h>
55. #include <string.h>
56. int main()
57. {
58.   char a[50],b[50],c[50],d[50];
59.   char buyuk[]=" > ";
60.   strcpy(a,"hamza");
61.   strcpy(b,"ziya");
62.   if(strcmp(a,b)>0)strcpy(d,a);else strcpy(d,b);
63.   strcat(d,buyuk);
64.   if(strcmp(a,b)<0)strcpy(c,a);else strcpy(c,b);
65.   strcat(d,c);
66.   printf("Alfabetik olarak buyuk olan: %s\n",d);
67.   return 0;
68. }
```

ÇIKTI:

```
Alfabetik olarak buyuk olan: ziya > hamza
```

7.4. Örnek: Kelimeleri kucukten buyuge,buyukten kucuge siralama.

7_3_ornek.c

```
#include<stdio.h>
#include <string.h>
int main()
{
  int i, j,n;
  printf("Kac tane kelime siralamak istiyorsunuz:");
  scanf("%d",&n);
  char kelime[n][100], temp[20];

  for (i = 0; i < n; i++)
  {
    printf("\n%d.Kelime:",i);
    scanf("%s", kelime[i]);
  }

  for (i = 0; i < n-1; i++)
  {
    for (j = i+1; j < n; j++)
    {
      if (strcmp(kelime[i], kelime[j]) > 0)
      {
            strcpy(temp, kelime[i]);
            strcpy(kelime[i], kelime[j]);
            strcpy(kelime[j], temp);
      }
    }
  }
  printf("\nKucukten buyuge : ");
  for (i = 0; i < n; i++)
    printf("%s < ", kelime[i]);
  printf("\nBuyukten kucuge : ");
    for (i = n-1; i >-1; i--)
      printf("%s > ", kelime[i]);
  return 0;
}
```

ÇIKTI:

```
Kac tane kelime siralamak istiyorsunuz:3
0.Kelime:ankara
1.Kelime:istanbul
2.Kelime:edirne
Kucukten buyuge : ankara < edirne < istanbul <
Buyukten kucuge : istanbul > edirne > ankara >
```

7.5. Örnek: strlen**,** strchr ve strrchr kullanımı.

7_4_ornek.c

```
1.  #include<stdio.h>
2.  #include <string.h>
3.  int main()
4.  {
5.    char mesaj[]="Merhaba nasilsin?",*pBas,*pSon;
6.    pBas=strchr(mesaj,'a');
7.    pSon= strrchr(mesaj,'a');
8.
9.    for(int i=0;i<strlen(mesaj);i++)
10.  {
11.    printf("\n%d - %c",i+1,mesaj[i]);
12.  }
13.
14.  printf("\n\na harfi bastan %d, sonda ise %d. sirada bulunuyor.",pBas-
     mesaj+1,pSon-mesaj+1);
15.
16.  return 0;
17. }
```

ÇIKTI:

```
1 - M
2 - e
3 - r
4 - h
5 - a
6 - b
7 - a
8 -
9 - n
10 - a
11 - s
12 - i
13 - l
14 - s
15 - i
16 - n
17 - ?

a harfi bastan 5, sonda ise 10. sirada bulunuyor.
```

7.6. Örnek: strlwr,strupr,puts kullanımı.

7_5_ornek.c

```
1.  #include<stdio.h>
2.  #include <string.h>
3.  int main()
4.  {
5.    char kucuk[]="ben kucuk harf ile yazilmistim",buyuk[]="BEN BUYUK HARF
    ILE YAZILMISTIM";
6.    puts(kucuk);
7.    puts(buyuk);
8.    puts("\n");
9.    puts(strupr(kucuk));
10.   puts(strlwr(buyuk));
11.
12.   return 0;
13. }
```

ÇIKTI:

```
ben kucuk harf ile yazilmistim
BEN BUYUK HARF ILE YAZILMISTIM


BEN KUCUK HARF ILE YAZILMISTIM
ben buyuk harf ile yazilmistim
```

8. HAFTA

A. MATEAMTİKSEL İSLEMLER(math.h)

math.h kutuphanesi matematiksel işlemleri yapmamıza yarayan bir kütüphanedir.

sqrt(sayi): karekok almaya yarar

pow(taban,us): us almaya yarar

floor(sayi): asagi yuvarlama

ceil(sayi): yukarı yuvarlama

fabs(sayi): mutlak

log(sayi): logaritma hesaplar

sin(derece): sinus hesaplar, radyan cinsinde dondurur, derece icin: aci*pi/180 yapilmali

cos(derece): cos hesaplar,radyan cinsinde dondurur, derece icin: aci*pi/180 yapilmali

(BU TABLO HAZIR KAYNAĞI: https://tr.wikipedia.org/wiki/Math.h)

İsim	Tanım
`acos`	Arccosinüs
`asin`	arcsinüs
`atan`	arctanjant
`atan2`	iki parametreli arctanjant
`ceil`	x'i kendinden büyük ilk tam sayıya yuvarlar

`cos`	cosinüs
`cosh`	hiperbolik cosinüs
`exp(double x)`	eksponensiyel fonksiyon, e^x hesaplaması
`fabs`	mutlak değer
`floor`	x'i kendinden küçük ilk tam sayıya yuvarlar
`fmod`	x/y işleminin kalanını bulur
`frexp`	fraction and power of 2.
`ldexp`	scale exponent of floating-point value
`log`	doğal (e tabaninda) logarithma
`log10`	log-10 tabaninda 10 logaritma alir
`pow(x,y)`	*Üs alma*
`sin`	Sinus hesaplar
`sinh`	hyperbolic sin
`sqrt`	kare kök alır

`tan`	tan(x)`i bulur
`tanh`	tanh(x)`i bulur

8. HAFTA ÖRNEKLER

6.2. Örnek: Math kütüphanesi genel örnek, sqrt, pow, floor, ceil, fabs, log, sin, cos

8_1_ornek.c

```
1.  #include <stdio.h>
2.  #include <math.h>
3.  #define pi 3.14159265
4.  int main()
5.  {
6.    int s1,secim=-1;
7.    double taban,us,s,derece;
8.    double sonuc;
9.    while(secim<0 || secim>7)
10.   {
11.
12.   while(secim<0 || secim>7)
13.   {
14.   printf("\n\n[0]Cikis\n[1]Karekok alma\n[2]Us
      alma\n[3]Yuvarlama\n[4]Mutlak\n[6]sin\n[7]cos");
15.   printf("\nIslem seciniz: ");
16.   scanf("%d",&secim);
17.   }
18.   switch (secim)
19.   {
20.     case 0:
21.       return 0;
22.       break;
23.     case 1:
24.       printf("Sayiyi giriniz: ");
25.       scanf("%lf",&s);
26.       printf("Sonuc %.5lf",sqrt(s));
27.       break;
28.     case 2:
29.       printf("Taban: ");
30.       scanf("%lf",&taban);
31.       printf("Us: ");
32.       scanf("%lf",&us);
33.       printf("Sonuc %.5lf",pow(taban,us));
34.     break;
35.     case 3:
36.       printf("Degeri girin: ");
37.       scanf("%lf",&s);
```

```
38.       printf("Asagi yuvarlama: %lf\nYukari yuvarlama %lf",floor(s),ceil(s));
39.     break;
40.     case 4:
41.       printf("Sayiyi girin: ");
42.       scanf("%lf",&s);
43.       printf("Sonuc |%lf| = %lf",s,fabs(s));
44.     break;
45.     case 5:
46.       printf("Sayiyi girin: ");
47.       scanf("%lf",&s);
48.       printf("Sonuc log(%lf) = %lf",s,log(s));
49.     break;
50.     case 6:
51.       printf("Aciyi girin: ");
52.       scanf("%lf",&s);
53.       derece=s*pi/180;
54.       printf("Sonuc sin(%lf) %lf derece ve %lf radyan",s,sin(derece),sin(s));
55.     break;
56.     case 7:
57.       printf("Aciyi girin: ");
58.       scanf("%lf",&s);
59.       derece=s*pi/180;
60.       printf("Sonuc cos(%lf) %lf derece %lf radyan",s,cos(derece),cos(s));
61.     break;
62.       default:
63.           printf("\nGecersiz secim...");
64.
65.   }
66.   secim=-1;
67. }
68.   return 0;
69. }
```

ÇIKTI:

```
[0]Cikis
[1]Karekok alma
[2]Us alma
[3]Yuvarlama
[4]Mutlak
[6]sin
[7]cos
Islem seciniz: 1
Sayiyi giriniz: 81
Sonuc 9.00000

[0]Cikis
[1]Karekok alma
```

```
[2]Us alma
[3]Yuvarlama
[4]Mutlak
[6]sin
[7]cos
Islem seciniz: 2
Taban: 3
Us: 3
Sonuc 27.00000

[0]Cikis
[1]Karekok alma
[2]Us alma
[3]Yuvarlama
[4]Mutlak
[6]sin
[7]cos
Islem seciniz: 3
Degeri girin: 4.5
Asagi yuvarlama: 4.000000
Yukari yuvarlama 5.000000

[0]Cikis
[1]Karekok alma
[2]Us alma
[3]Yuvarlama
[4]Mutlak
[6]sin
[7]cos
Islem seciniz: 4
Sayiyi girin: -4
Sonuc |-4.000000| = 4.000000

[0]Cikis
[1]Karekok alma
[2]Us alma
[3]Yuvarlama
[4]Mutlak
[6]sin
[7]cos
Islem seciniz: 6
Aciyi girin: 30
Sonuc sin(30.000000) 0.500000 derece ve -0.988032 radyan

[0]Cikis
[1]Karekok alma
```

```
[2]Us alma
[3]Yuvarlama
[4]Mutlak
[6]sin
[7]cos
Islem seciniz: 7
Aciyi girin: 30
Sonuc cos(30.000000) 0.866025 derece 0.154251 radyan
```

6.3. Örnek:

8_#_ornek.c
69.
ÇIKTI:

9. HAFTA

A. STRUCTLAR

Bir program yaparken, çoğu kez, belirli ama farklı türden verilerin bir tek kayıt içinde birarada bulunmasının yararlı ve hatta gerekli olduğu durumlarla karşılaşırız. Birden çok verinin (değişkenin) bir araya getirilmesiyle oluşturulan yeni birime (veri türüne) C dilinde bir yapı (structure) adı verilir.[8]

[8] http://www.baskent.edu.tr/~tkaracay/etudio/ders/prg/c/yapi.htm

B.

7. HAFTA ÖRNEKLER

7.1. Örnek:

3_1_ornek.c
70.
ÇIKTI:

7.2. Örnek:

3_2_ornek.c
70.
ÇIKTI:

7.3. Örnek:

3_3_ornek.c
63.
ÇIKTI:

7.4. Örnek:

3_4_ornek.c
14.
ÇIKTI:

8. HAFTA

C. DOSYA İŞLEMLERİ

Dosyalama işlemleri kullanmadan yaptığımız uygulamalarda veriler programımızı sonlandırdığımızda kayboluyor. Aynı veriye tekrar ihtiyacımız olduğu durumarda tek tek bu verileri tekrar girmemize neden oluyor, bu nedenle dosya işlemleri ile sistemimizdeki herhangi bir(txt,bat,sql) dosyaya yazıp verilerimizi saklayabilir ve okuma yazmada yapabiliriz.

C'de dosyalama yapılarına göre ikiye ayrılıyor text ve binary. Text ve binary arasındaki farklar:

	TEXT DOSYALARI	BİNARY DOSYALARI
String bilgilerin sonundaki '\n' karakteri:	'\r'+'\n' karakter çiftlerine dönüştürülerek kaydedilir.	Aynen kayededilir,istenirse kullanılmayabilir.
Kayıtlara erişim:	Sıralı	Rastgele, anahtar alan vasıtası ile erişilir.
Kayıt yazma:	Sıralı, yazılan en son kayıttan sonra dosya sonu işareti konulur.	Anahtar alan vasıtası ile boş bir kayda konumlanarak yapılır.
Kayıt okuma:	İlk kayıttan aranılan kayıta sıra ile okunarak yapılır.	Anahtar alan vasıtası ile doğrudan yapılabilir.
Kayıt silme:	İşleminden sonra diğer kayıtlar otomatik olarak yeniden sıralanırlar.	Anahtar alan vasıtası ile olur ve silinen kaydın yeri yeni kayıt ile doldurulunca kadar boş kalır.

Girilen bilgiler:	Yan bellekte uzumlukları kadar yer kaplarlar.	Yan bellekte verilerin tiplerinin uzunluğu kadar yer kaplarlar ve anahtar alan vasıtasıyla boş yere kaydedilir.

D. DOSYA TANIMLAMA

C'de standart bir dosya tipi tanımlanmamıştır. "stdio.h" başlık dosyası içerisinde Ogrenci yapısal veri tipi tanımlanmıştır. Bu veri tipi şu şekildedir.

```
typedef struct
{
   char *isim[30];
   char cinsiyet;
   int okulNumarasi; }Ogrenci;
```

Dosya tanımlaması: `Ogrenci *dosya_adi` şeklinde tanımlanır.

typedef struct{ char *isim[30]; char cinsiyet; int okulNumarasi; char *okul[30]; }Ogrenci;

E. DOSYAYA BİLGİ KAYDETME

Fonksiyon	Görevi
putc()	Dosyaya karakter kaydeder.
fputc()	Dosyaya bir karakter veri kaydeder.
fputs()	Dosyaya string kaydeder.
fwrite()	Dosyaya bir kayıt dizi veya karakter kaydeder.
fprintf()	Dosyaya biçimlendirilmiş veri kaydeder.

F. DOSYA AÇILIŞ MODLARI

r	Okuma için bir metin dosyası açar.
w	Yazma için bir metin dosyası oluşturur.
a	Bir metin dosyasına ekleme yapar.
rb	Okuma için bir dosyayı ikili sistemde açar.
wb	Yazma için ikili sistemde bir dosya oluşturur.
ab	İkili sistemde bir dosyaya ekleme yapar.
r+	Okuma ve yazma için bir metin dosyası açar.
w+	Okuma ve yazma için bir metin dosyası oluşturur.
a+	Okuma ve yazma için bir metin dosyası oluşturur veya ekleme yapar.
r+b	Okuma ve yazma için bir ikili sistem dosyası açar.
w+b	Okuma ve yazma için bir ikili sistem dosyası oluşturur.
a+b	Okuma ve yazma için bir ikili sistem dosyasına ekleme yapar.

Kaynakça

bilgigunlugum. (tarih yok). bilgigunlugum: https://www.bilgigunlugum.net/prog/cprog/c_stdkut/string adresinden alınmıştır

dijitalders. (tarih yok). dijitalders: https://www.dijitalders.com/icerik/106/5440/c_programlama_dilinde_degiskenler.html adresinden alınmıştır

dijitalgezginler16. (tarih yok). *medium*. medium: https://medium.com/@dijitalgezginler16/c-programlama-dili-veri-tipleri-de%C4%9Fi%C5%9Fkenler-ve-sabitler-1ce8c43003c1 adresinden alınmıştır

geeksforgeeks. (tarih yok). geeksforgeeks: https://www.geeksforgeeks.org/ adresinden alınmıştır

programiz. (tarih yok). programiz: https://www.programiz.com/ adresinden alınmıştır

sadievrenseker. (tarih yok). *bilgisayarkavramlari*. bilgisayarkavramlari: http://bilgisayarkavramlari.sadievrenseker.com/2007/10/16/pointer-gosterici-2/ adresinden alınmıştır

turkmuhendis. (tarih yok). turkmuhendis: https://turkmuhendis.net/wp-content/uploads/2019/04/%C4%B0%C5%9Flem-%C3%96ncelik-S%C4%B1ras%C4%B1-.jpg adresinden alınmıştır

VATANSEVER, F. (tarih yok). *Algoritma ve Programlama Giriş.* seçkin yayınları.

vdemir. (tarih yok). https://vdemir.github.io: https://vdemir.github.io/ceviriler/ceviriler2/2018-01-03-Preprocessor.html adresinden alınmıştır

yapbenzet.kocaeli.edu.tr. (tarih yok). yapbenzet.kocaeli.edu.tr: http://yapbenzet.kocaeli.edu.tr/cpp-gostericiler-pointerlar/ adresinden alınmıştır