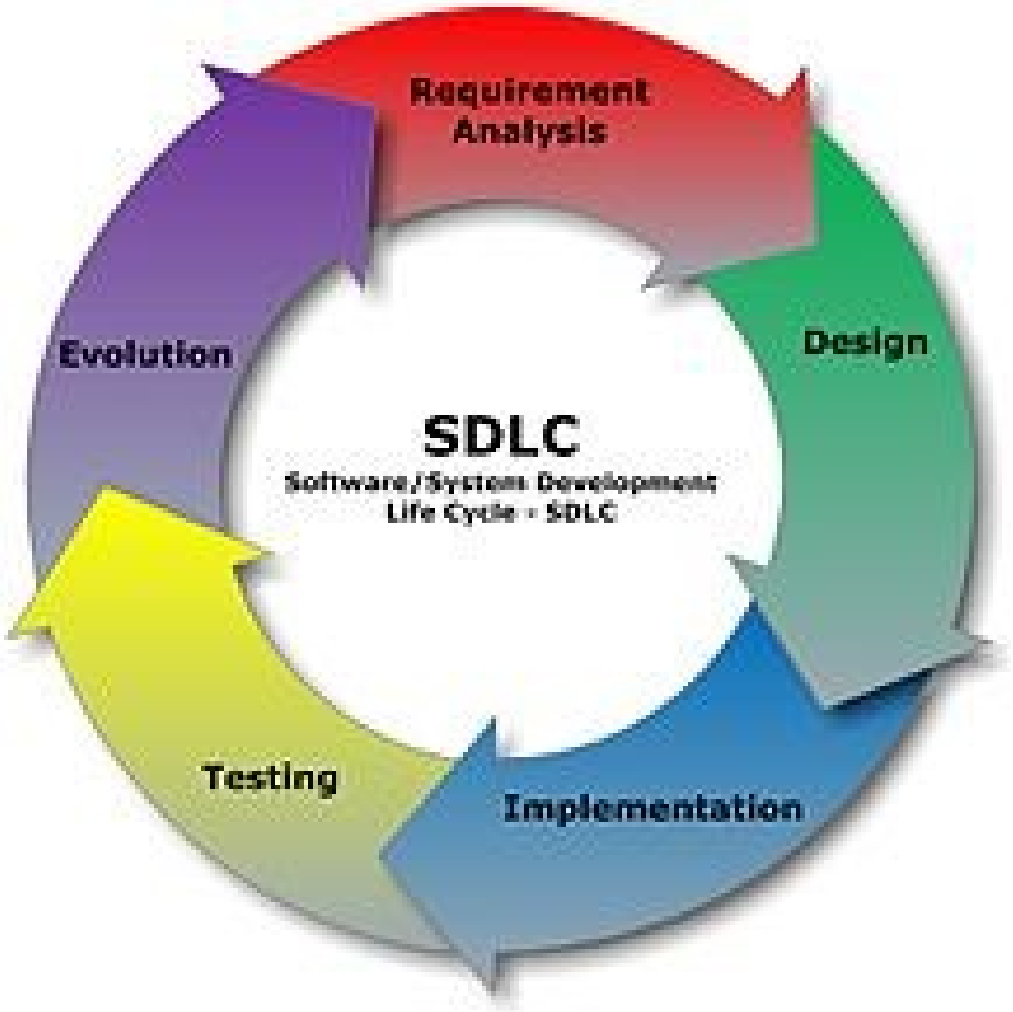
BİL2001 Algoritmalar ve Veri Yapıları

ÖĞR. GÖR. DR. ALPER VAHAPLAR

2018 – 2019

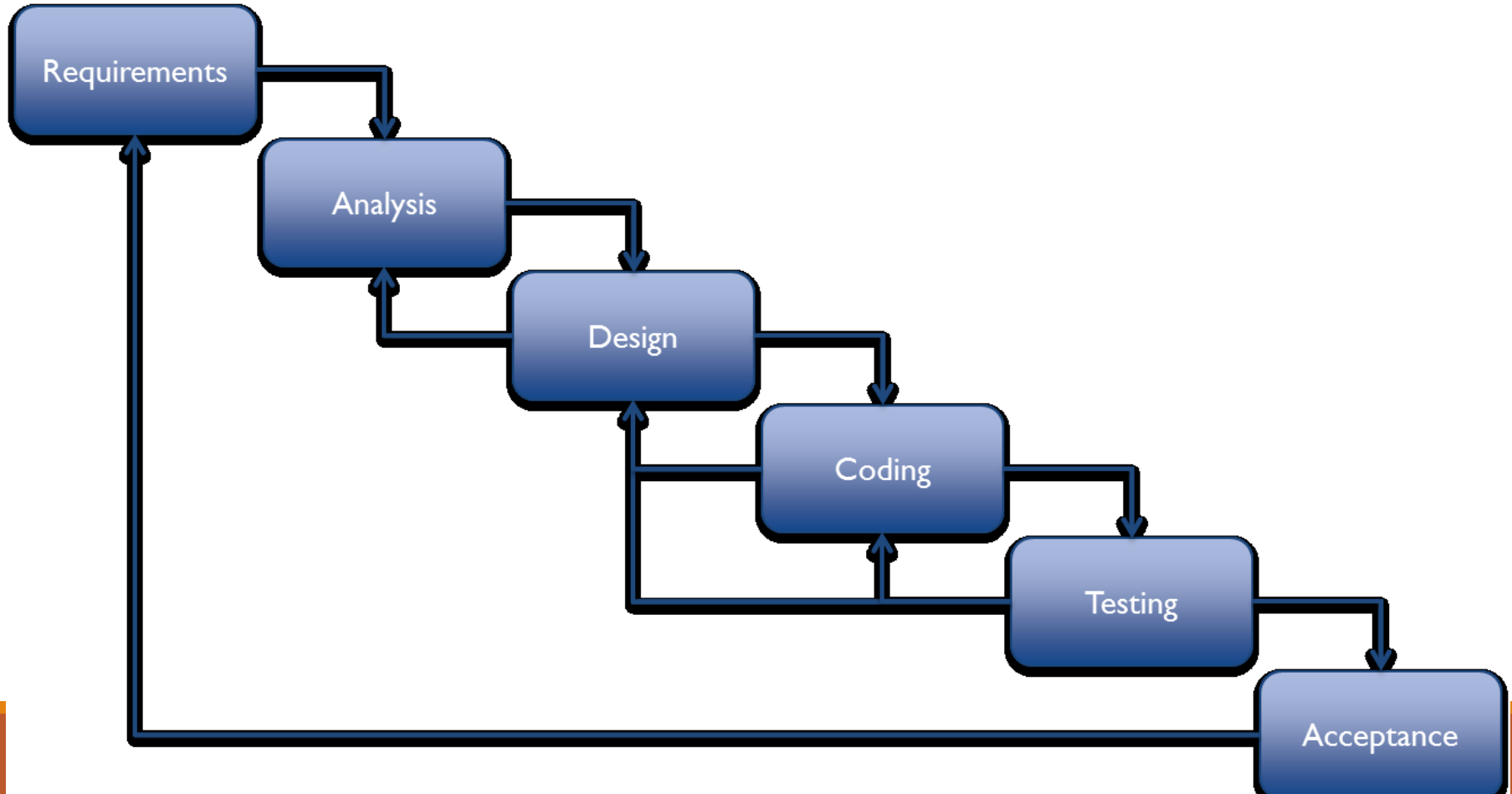
Yazılım Yaşam Döngüsü

- ✓Yazılım Geliştirme Yaşam Döngüsü
- ✓SDLC



Yazılım Yaşam Döngüsü

- ✓ Waterfall Model



Bazı Tanımlar

- ✓ Bazı Tanımlar
- ✓ Algoritma: Bir problemin çözüm tarifi, bir problemin çözümünde kullanılan komutlar dizisi.
- ✓ Program: Bir algoritmanın bir programlama dilinde ifade edilmiş hali.
- ✓ Kaba Kod (Pseudo Code): Belirli bir algoritmanın bir programlama diline benzer şekilde ifadesi.
- ✓ Yazılım (Software): Belirli bir görevi yapmak üzere bir veya birden çok program ve doküman kümesi.

Bazı Tanımlar

- ✓ **Veri (Data):** Bir nesneye ait herhangi bir özelliğinin herhangi bir andaki ölçülen, gözlenen ya da hesaplanan değeri.
- ✓ **Değişken (Variable):** Verilerin tutulduğu bellek adreslerine verilen simgesel isim.
- ✓ **İşlemci (Operator):** Veriler üzerinde tanımlanmış işlemleri gerçekleştirmeye yarayan simgeler.

Bazı Tanımlar

- ✓ **Veri Tipi (Data Type):** Bir değişkenin alabileceği değerler kümesini ve o değişken için ayrılacak bellek miktarını belirleyen ifade.

- ✓ **Veri Yapıları (Data Structures):** Bilgisayar ortamında verilerin etkin olarak saklanması ve işlenmesi için kullanılan yapılardır.

- ✓ **Veri Modeli (Data Model):** Belirli bir veri kümesinin mantıksal ya da dizisel/ilişkisel durumu.

Bazı Tanımlar

- ✓ Veri yapıları ile veri modeli farklı kavramlardır.

- ✓ Fakat iç içe geçmişlerdir.
  - Veri Yapıları verinin saklanma biçimiyle,
  - Veri Modelleri, veriler arasındaki ilişkiler ve bağlantılarla ilgilenir.

Bazı Tanımlar

- ✓ Ör:
  - Veri: Sınıftaki Öğrenciler
  - Veri Yapısı: Sıralar
  - Veri Modeli: Sıraların yerleşimi
- ✓ Ör:
  - Veri: Uçan Kazlar
  - Veri Yapısı: Gökyüzü
  - Veri Modeli: “V” şeklinde uçmak

Veri Modeli – Data Model

- ✓ Veri Modeli:
- ✓ Problem çözümü için kavramsal yaklaşım yöntemidir.

- ✓ Bilgisayar ortamında uygulanacak tüm matematik ve mühendislik problemleri bir veri modeline yaklaştırılarak veya yeni veri modelleri tanımlaması yapılarak çözülebilir.

Veri Yapıları – Veri Modelleri

- ✓ Neden ihtiyaç var?
  - Yazılımlar giderek karmaşıklaşıyor,
  - Yönetimi ve güncellenmesi zorlaşıyor,
  - Daha etkin ve doğru programlama yapabilmek için daha yeni, gelişmiş kavramsal yapılara ihtiyaç duyuluyor.
- ✓ Ör: Google – web sitelerinin ve sayfalarının indekslenmesi

Veri Yapıları – Veri Modelleri

- ✓ İyi bir yazılım için
  - Doğru ve düzgün bir tasarım
  - Kolay bakım ve yönetim
  - Güvenilir, Kolay kullanımlı, Hızlı algoritmalar
  - Kullanışlı Veri Yapıları,
  - Etkin Algoritmalar

  gerekir.

Veri Yapıları – Veri Modelleri

- **Örnek**
- Her biri satır başına ortalama 10 kelimeden ve yine ortalama 20 satırdan oluşan 3000 metin koleksiyonu olduğunu düşünelim.
- 600,000 kelime
- Bu metinler içinde “dünya” kelimesini bulmak isteyelim
- Her karşılaştırma 1 sn. sürsün.
- Ne kadar sürer?

Veri Yapıları – Veri Modelleri

- ✓ **Çözüm. 1:**
- ✓ Sıralı eşleştirme:
- ✓ 1 sn. x 600,000 kelime= 166 saat
- ✓ **Çözüm. 2:**
- ✓ İkili Arama (Binary searching):
  - kelimeler sıralanır
  - sadece tek yarıda arama yapılır
  - toplam adım sayısı $\log_2 N = \log_2 600000$
  - yaklaşık 20 adım (çevrim)
  - 20 sn.
- ✓ 20 saniye veya 166 saat!

Veri Yapıları – Veri Modelleri

- **Örnek :**25 değerini 5,8,12,15,15,17,23,25,27 dizisinde arayalım. Kaç adımda sonuç bulunur?

- 25 ? 15 — 15 17 23 25 27
- 25 ? 23 — 23 25 27
- 25 ? 25

Veri Yapısı ve Önemi

- ✓ Her şey 2'lik düzendeki sayılardan oluşur.
- ✓ Sadece 0'lar ve 1'ler var.
  - Disklerde,
  - Bellekte,
  - işlemcideki registerlarda,
- ✓ "01000001" Nedir?
- ✓ Formatı/yapısı bilinirse cevap verilebilir.

Veri Yapısı ve Bilgi

- ✓ 0100 0010 0100 0001 0100 0010 0100 0001

- ✓ Yukarıdaki bit dizisi;
  - Karakter dizisi (string) ise (ASCII): B A B A
  - BCD (Binary Coded Decimal) ise: 4 2 4 1 4 2 4 1
  - 16-bit tam sayı ise: 16961 16961
  - 32-bit tam sayı ise: 1111573057
  - 32-bit gerçel sayı ise: $(-1)^{0}x0.4276801x2^{66-127}$

Alıştırma

- ✓ Bir öğretmen sınıftaki 100 öğrenciden 24'ünün erkek ve 32'sinin kız olduğunu söylüyor. Sayı sistemini ve 10'luk karşılıklarını bulunuz…

Alıştırma

21	17	24	27
30	26	19	18
23	20	31	25
16	28	22	29

1

13	24	14	15
26	29	31	25
12	27	8	30
9	28	11	10

2

4	13	5	22
28	23	12	20
30	6	21	29
15	31	14	7

3

22	11	23	26
27	15	2	30
3	14	31	7
18	10	6	19

4

5	3	27	15
29	1	23	31
7	25	9	13
19	17	21	11

5

Veri Yapıları – Data Structures

- **Temel Veri Yapıları (Basic – Primitive)**
  - daha çok programlama dilleri tarafından doğrudan değişken veya sabit bildirimi yapılarak kullanılır.

- **Kullanıcı Tanımlı Veri Yapıları (User Defined)**
  - kendisinden önceki tanımlamalı veya temel veri yapıları üzerine kurulurlar; yani, önceden geçerli olan veri yapıları kullanılarak sonradan tanımlanırlar.

Temel Veri Yapıları

- **Karakterler**
  - ASCII — Her karakter 8 bit (2^8 = 256 farklı karakter)
  - Unicode — Her karakter 16 bit (2^{16} = 65536 farklı karakter)
- **Tamsayılar**
  - 8 bit — short, short int, ShortInt, byte
  - 16 bit — integer, int, integer16, Int16
  - 32 bit — long, long int, LongInt, integer32, Int32
- **Ondalıklı (Gerçel) Sayılar**
  - 16 bit — half (IEEE 754-2008)
  - 32 bit — single, float (C)
  - 64 bit — double, real (Pascal)
  - 128 bit — quad
- **Diziler (Arrays)**

Temel Veri Tipleri – C++

Name	Description	Size*	Range*
char	Character or small integer.	1 byte	signed: -128 to 127 unsigned: 0 to 255
short int(short)	Short Integer.	2 bytes	signed: -32768 to 32767 unsigned: 0 to 65535
int	Integer.	4 bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
long int (long)	Long integer.	4 bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
bool	Boolean value. It can take one of two values: true or false.	1 byte	true or false
float	Floating point number.	4 bytes	+/- 3.4e +/- 38 (~7 digits)
double	Double precision floating point number.	8 bytes	+/- 1.7e +/- 308 (~15 digits)
long double	Long double precision floating point number.	8 bytes	+/- 1.7e +/- 308 (~15 digits)
wchar_t	Wide character.	2 or 4 bytes	1 wide character

Temel Veri Tipleri

- ✓ Karakterler
- ✓ ASCII
- ✓ (American Standard Code for Information Interchange)
- ✓ Her karakter 7 bit ile ifade edilir
- ✓ 128 farklı karakter
- ✓ Extended ASCII 8 bit (256 karakter)

KOD	CHAR	KOD	CHAR	KOD	CHAR	KOD	CHAR	KOD	CHAR	KOD	CHAR	KOD	CHAR
000	(nul)	043	+	087	W	138	Ë	185	╣	221	▌		
001	(soh)	044	,	088	X	139	Ô	186	║	222	▐		
002	(stx)	045	-	089	Y	140	Ó	187	╗	223	▀		
003	(etx)	046	.	090	Z	141	ı	188	╝	224	α		
004	(eot)	047	/	091	[	142	ƒ	189	╜	225	β		
005	(enq)	048	0	092	\	143	≈	190	╛	226	Γ		
006	(ack)	049	1	093	]	144	...	191	┐	227	π		
007	(bel)	050	2	094	^	145	È	192	└	228	Σ		
008	(bs)	051	3	095	_	146	Δ	193	┴	229	σ		
009	(tab)	052	4	096	`	147	Ù	194	┬	230	µ		
010	(lf)	053	5	097	a	148	ˆ	195	├	231	τ		
011	(vt)	054	6	098	b	149	Ú	196	─	232	Φ		
012	(np)	055	7	099	c	150	°	197	┼	233	Θ		
013	(cr)	056	8	100	d	151	¨	198	╞	234	Ω		
014	(so)	057	9	101	e	152	0	199	╟	235	δ		
015	(si)	058	:	102	f	153	Ö	200	╚	236	∞		
016	(dle)	059	;	103	g	154	Ü	201	╔	237	Ø		
017	(dc1)	060	<	104	h	155	-	202	╩	238	ε		
018	(dc2)	061	=	105	i	156	£	203	╦	239	∩		
019	(dc3)	062	>	106	j	157		204	╠	240	≡		
020	(dc4)	063	?	107	k	158	Ş	205	═	241	±		
021	(nak)	064	@	108	l	159	Ş	206	╬	242	≥		
022	(syn)	065	A	109	m	160	.	207	╧	243	≤		
023	(etb)	066	B	110	n	161	ı	208	╨	244	⌠		
024	(can)	067	C	111	o	162	Û	209	╤	245	⌡		
025	(em)	068	D	112	p	163	.	210	╥	246	÷		
026	(eof)	069	E	113	q	164	Ò	211	╙	247	≈		
027	(esc)	070	F	114	r	165	‡	212	╘	248	o		
028	(fs)	071	G	115	s	166		213	╒	249	˙		
029	(gs)	072	H	116	t	167	ğ	214	╓	250	·		
030	(rs)	073	I	117	u	168	TL	215	╫	251	√		
031	(us)	074	J	118	v	169	°	216	╪	252	ⁿ		
032	sp	075	K	119	w	170	Ω	217	┘	253	²		
033	!	076	L	120	x	171	O	218	┌	254	■		
034	i	077	M	121	y	172	0	219	█	255			
035	#	078	N	122	z	173	'	220	▄				
036	$	079	O	123	{	174	²						
037	%	080	P	124	|	175	▒						
038	&	081	Q	125	}	176	▓						
039	ë	082	R	126	~	177	│						
040	(	083	S	127		178	┤						
041	)	084	T	128	Ç	179	╡						
042	*	085	U	129	ü	180	╢						
		086	V	130	É	181	╖						
				131	â	182	╕						
				132	‰	183	╣						
				133	‡	184	╗						
				134	Â								
				135	ç								
				136	Ê								
				137	Î								

Temel Veri Tipleri

- ✓ Karakterler
- ✓ Unicode
- ✓ Doğal dillerden bağımsız kodlama sistemi
- ✓ Her karakter 16 bit yer kaplar
- ✓ Toplam 65536 farklı karakter tanımlanabilir.

Temel Veri Tipleri

- ✓Tamsayılar (Integers)
- ✓Tamsayı biçimleri
  - Natural (Doğal, ikilik düzendeki karşılığıyla)
  - One’s Complement (1’e Tümleme)
  - Two’s Complement (2’ye Tümleme)
  - Binary Coded Decimal (BCD) (Her basamak 4 bit ile ifade edilir)

Temel Veri Tipleri

- ✓Örnek: 39 sayısının ifadesi

Sayı	39
Doğal	100111
One’s Complement	011000
Two’s Complement	011001
BCD	00111001

Temel Veri Tipleri

- ✓ Tamsayılar (Integers)
- ✓ Negatif Tamsayıların ifadesi
  - (a) İşaret Biti kullanarak
  - (b) Two's Complement kullanarak
- ✓ Ör: -39 sayısının ifadesi
  - 39 = 100111
  - (a) **1**100111
  - (b) **1**011001

Temel Veri Tipleri

- ✓ Oransal – Gerçel Sayılar
- ✓ (Fractional – Real Numbers)
- ✓ Kayan Noktalı (Floating Point)
- ✓ Sabit Noktalı (Fixed Point)
- ✓ 0.39
- ✓ $3.9x10^{-1}$
- ✓ $39x10^{-2}$
- ✓ $0.039x10^{1}$

Temel Veri Tipleri

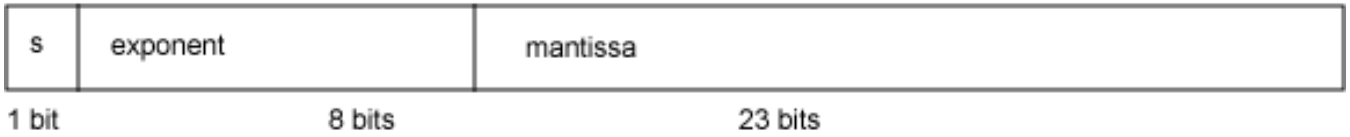
- ✓ Floating Point gösterimi
- ✓ IEEE 754 ($S = \pm M \times B^{\pm E}$)

IEEE Floating Point Representation

IEEE Double Precision Floating Point Representation

Floating Point Numbers

IEEE Floating Point Representation



- ✓ **32 bit**
- ✓ **11000000110110011001100110011010**
- ✓ **[1] [10000001] [10110011001100110011010]**
- ✓ **Sign exponent fraction**
- ✓ **Sign = [1] (yani sayı NEGATİF)**
- ✓ **Exponent = [10000001] = 129**
- ✓ **Fraction = [10110011001100110011010]**
- ✓ $=\mathbf{1*2^{-1}}+ 0*2^{-2} + \mathbf{1*2^{-3}} + \mathbf{1*2^{-4}} + 0*2^{-5} + 0 * 2^{-6} + \ldots$
- ✓ = 0.7000000476837158.

Floating Point Numbers

- ✓ **[1] [10000001] [10110011001100110011010]**
- ✓ **Sign exponent fraction**
- ✓ **Fraction = [10110011001100110011010]**
- ✓ $=\mathbf{1*2^{-1}}+0*2^{-2}+\mathbf{1*2^{-3}}+\mathbf{1*2^{-4}}+0*2^{-5}+0*2^{-6}+\ldots$
- ✓ $= 0.7000000476837158.$
- ✓ $(-1)^{\text{sign}} * (1+\text{fraction}) * 2^{(\text{exponent} - \text{bias})}$
- ✓ $(-1)^{1} *(1.7000000476837158) * 2^{129-127}$
- ✓ **= -6.8**

Floating Point Numbers

- ✓Ör: 0.085'i IEEE 754'e göre gösterin. (32 Bit)
- ✓Sign Bit = 0 (sayı POZİTİF)
- ✓0.085'i scientific notation ile göster
- ✓$0.085 = (-1)^0 * (1+\text{fraction}) * 2^{\text{power}}$
- ✓$0.085 / 2^{\text{power}} = (1+\text{fraction})$
- ✓0.085'i (1+fraction) olana kadar böl

Floating Point Numbers

- ✓ Ör: 0.085'i IEEE 754'e göre gösterin. (32 Bit)
- ✓ 0.085'i (1+fraction) olana kadar böl
- ✓ $0.085 / 2^{-1} = 0.17$

  $0.085 / 2^{-2} = 0.34$

  $0.085 / 2^{-3} = 0.68$

  $0.085 / 2^{-4} = 1.36$
- ✓ Power = -4 = (exponent – bias)
- ✓ Bias = 127
- ✓ Exponent = 123 (01111011)

Floating Point Numbers

- ✓Ör: 0.085'i IEEE 754'e göre gösterin. (32 Bit)
- ✓Fraction = 0.36'yi 2'lik düzende göster
- ✓0.36 = (0/2) + **(1/4)** + (0/8) + **(1/16)** + **(1/32)** +...
- ✓$0.36 = 2^{-2} + 2^{-4} + 2^{-5}$+...
- ✓$(0.36)_{10} \sim (0.01011100001010001111011)_2$

	Sign	Exponent	Fraction
Decimal	0	123	0.36
Binary	**0**	**01111011**	**01011100001010001111011**

Temel Veri Tipleri

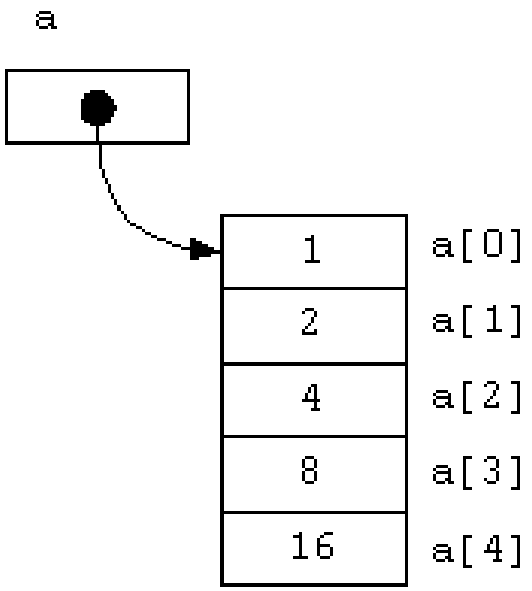
- ✓ Metinsel İfadeler (Strings)
- ✓ Aslında karakter dizisidir.
- ✓ Eleman sayısı ve sonlandırma karakteri
- ✓ Ör: andimiz = “Ne Mutlu Türk’üm Diyene”
- ✓ Bellekte tutulma yöntemleri
  - Karakter sayısını saklamak
  - Sonlandırma karakteri kullanmak

Temel Veri Tipleri

- ✓ Bellekte tutulma yöntemleri
  - Karakter sayısını saklamak
  - Sonlandırma karakteri kullanmak
- ✓ andimiz = “Ne Mutlu Türk’üm Diyene”
- ✓ Pascal
  - andimiz → [**23**,N,e, ,M,u,t,l,u, ,T,ü,r,k,’,ü,m…]
- ✓ C
  - andimiz →[N,e, ,M,u,t,l,u, ,T,ü,r,k,’,ü,m ….e,n,e,**\0**]

Temel Veri Yapıları

- ✓ Diziler (Arrays)
- ✓ Aynı tip değişkenin birden çok sayıda değerinin aynı isim altında saklanabildiği veri yapısıdır.



Temel Veri Yapıları

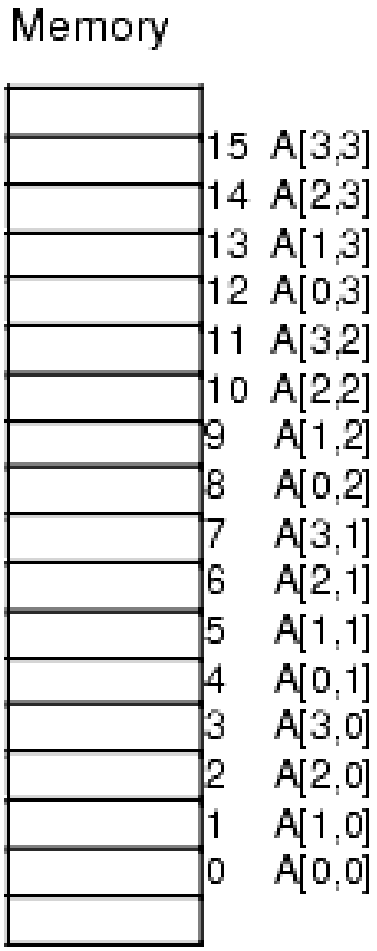
- ✓ Diziler
- ✓ Tek boyutlu dizi = vektör
- ✓ Çok boyutlu dizi = matris
- ✓ C dilinde dizi tanımlama
  - int c[10] //10 elemanlı c dizisi
  - İlk eleman c[0]
  - Son eleman c[9]

Temel Veri Yapıları

- ✓Çok boyutlu di

A :array [0..3,0..3] of char

	0	1	2	3
0	0	1	2	3
1	4	5	6	7
2	8	9	10	11
3	12	13	14	15



Temel Veri Yapıları

- ✓ Çok boyutlu diziler

$$\begin{bmatrix} 3 & 1 & 7 \end{bmatrix}$$

C	A[1][3]
PASCAL	A[1,3]
BASIC	A(1,3)

$$\begin{bmatrix} 4 & -3 & -1 \\ 6 & 2 & 1 \\ 0 & 3 & -2 \end{bmatrix}$$

Kullanıcı Tanımlı Veri Yapıları

- Temel veri yapılarının alt kümelerinden ya da veri yapılarının birleştirilmesi ile oluşturulmuş veri yapılarıdır.

- Ör: *type* tamsayi = integer; (Pascal)

  var i : tamsayi;

- Alt alan (subrange) veri tipleri
- Ör: *type* sinif =1..4; (Pascal)

  var x:sinif;

Kullanıcı Tanımlı Veri Yapıları

```
type    kucukharfler='a' .. 'z';
        buyukharfler='A' .. 'Z';
        birkisim = 'K' .. 'c';
```

Kullanıcı Tanımlı Veri Yapıları

- ✓ Sıralı Tipler (Enumerated)
- ✓ Programcının kendi oluşturduğu hiyerarşide sıralanmış veri tipleridir.
- ✓ Ör:

```
type  gunler = (Pts, Sal, Crs, Prs, Cum, Cts, Pz);
      rutbe = (tegmen, yuzbasi, binbasi, albay, yarbay);
      okullar = (ilk, orta, lise, universite);
```

Kullanıcı Tanımlı Veri Yapıları

- Kayıt (Record) – Struct Veri Tipi
- Temel veri yapılarının birleştirilmesi ile oluşturulmuş veri yapılarıdır.

```
struct kayit {
  char ad[15];
  char soyad[25];
  char adres[150];
  unsigned short int yas;
}
```

Kullanıcı Tanımlı Veri Yapıları

- Kayıt (Record) – Struct Veri Tipi
- Bir **struct** veri yapısının boyutu, içinde yer alan tüm temel veri yapılarının boyutlarının toplamına eşittir.

```
struct kayit {
   char ad[15];                  15
   char soyad[25];               25
   char adres[150];             150
   unsigned short int yas;        2
 }
```

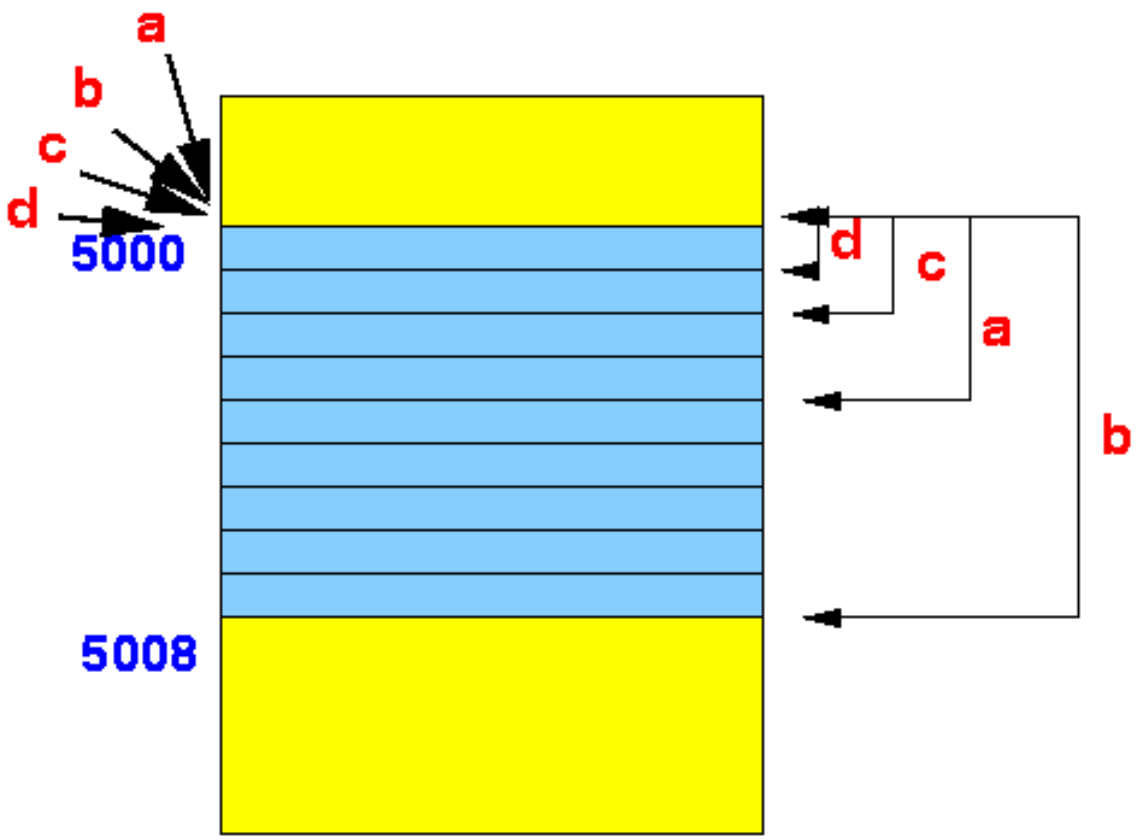
Kullanıcı Tanımlı Veri Yapıları

- Birden çok değişkenin aynı bellek alanını kullanmasına izin veren **union** veri yapısının boyutu ise; içinde yer alan temel veri yapılarından en büyüğünün boyutuna eşittir.

```
union kisiler {
  char ad[15];                  15
  char soyad[25];               25
  unsigned long int tel;         4
}
```

Kullanıcı Tanımlı Veri Yapıları

```
union  myExample
{
    int a;
    double b;
    short c;
    char d;
}
```



Haftaya...

Veri Modelleri