Temel Programlama 1

Öğr. Gör. Furkan DURMUŞ

Samsun Üniversitesi
Teknik Bilimler
Meslek Yüksekokulu

Bilgisayar ve Programlama

- Temel Programlama dersinin iyi anlaşılabilmesi için bu dersin temelini oluşturan kavramların iyi bilinmesi gerekir.
- Bilgisayarın tanımı ve çalışma mantığı
- Programın tanımı ve çeşitleri
- Programlama Dilleri
- Programlamanın Aşamaları
- Algoritma
- Akış şemasıları (çizelge, diyagram) yapısı

Öğr. Gör. Furkan DURMUŞ

Bu Haftanın Ders Kazanımları

Bu haftayı bitirdiğinizde,

{ 1 } Bilgisayar ve çalışma mantığını

{ 2 } Programlama nedir ve programlama dilleri

{ 3 } Programlamanın aşamalarını

{ 4 } Algoritmanın ne olduğunu ve Algoritma mantığını

{ 5 } Temel algoritma kavramlarını

{ 6 } Yazılımın yaşam döngüsünü

öğrenmiş olacaksınız.

Bilgisayar Nedir?

Bilgisayar, kullanıcıdan aldığı verilerle mantıksal ve aritmetiksel işlemleri yapan, yaptığı işlemlerin sonucunu saklayabilen, sakladığı bilgilere istenildiğinde ulaşılabilen elektronik bir makinedir.

Bu işlemleri yaparken veriler girilir ve işlenir. Ayrıca,istenildiğinde yapılan işlemler depolanabilir ve çıkısı alınabilir.

Bilgisayarın Temel Bileşenleri

Giriş

Kişi tarafından veya bilgisayar tarafından sağlanan verilerdir. Bu veriler, sayılar, harfler, sözcükler, ses sinyalleri ve komutlardır. Veriler giriş birimleri tarafından toplanır.

İşlem

Veriler insanların amaçları doğrultusunda, programın yetenekleri ölçüsünde işlem basamaklarından geçer.

Bellek

Verilerin depolandığı yerdir. Giriş yapılan ve işlenen veriler bellekte depolanır.

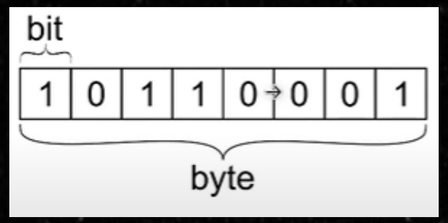
Çıkış

Bilgisayar tarafından işlem basamaklarından geçirilerek üretilen yazı, resim, tablo, müzik, grafik, görüntü, vb.nin ekrandan ya da yazıcı, hoparlör gibi farklı çıkış birimlerinden alınmasıdır.

Bilgisayarı Nasıl Çalışır?

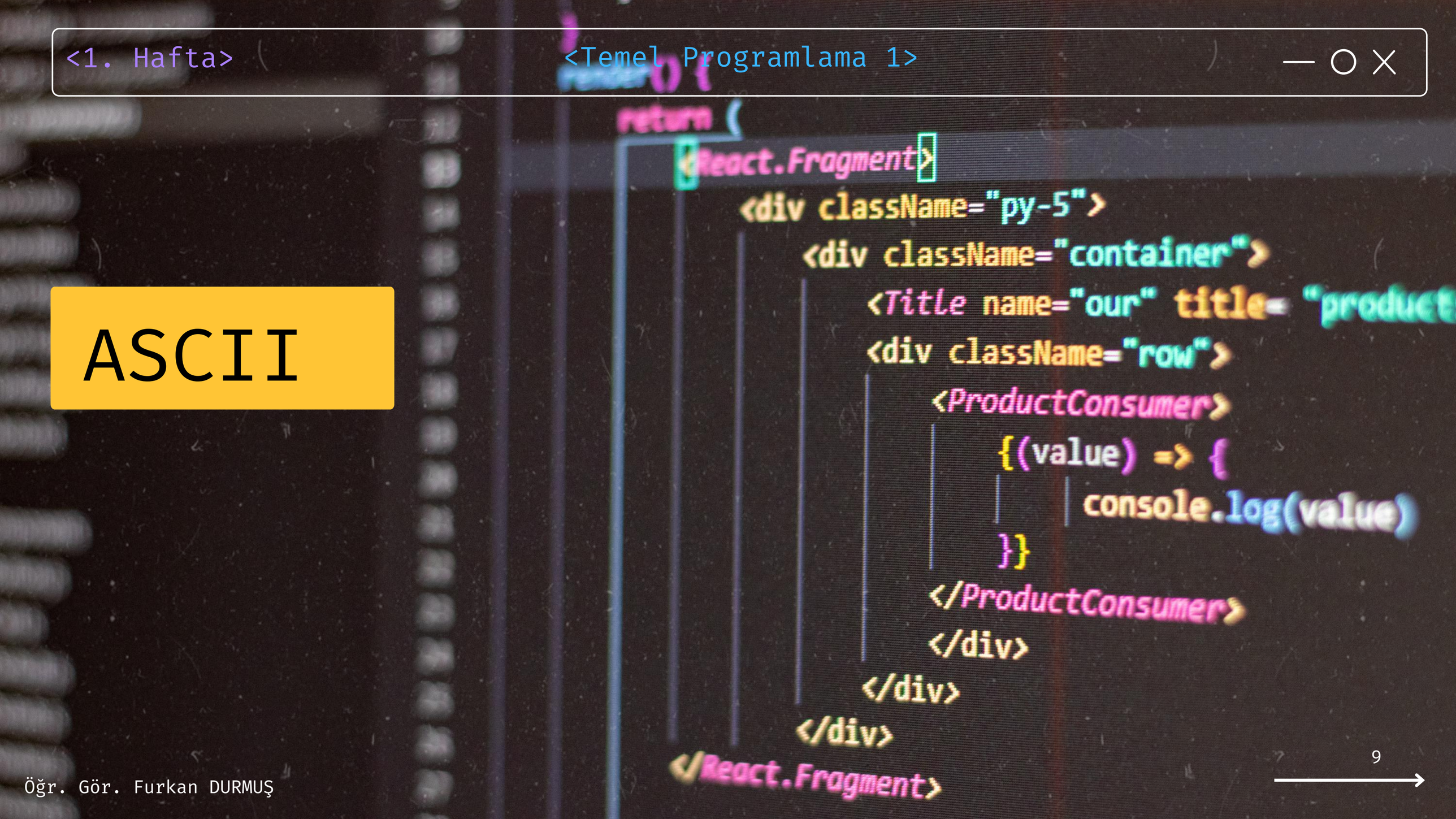
Bilgisayarın nasıl çalıştığını öğrenmek için onun bilgileri nasıl kullandığını anlamak gerekir. Harfler ve rakamlar bilgisayarda kodlar şeklinde ifade edildikten sonra kullanılır. Bilgisayarlarda kodlar elektrik olarak voltajın olup olmaması ile ifade edilir. Voltaj var, lamba yanıyorsa 1; voltaj yok, lamba yanmıyorsa 0 kodlarını alır. İki durumlu olan bu kodlamaya "ikilik sistem" denir. Bilgisayara tuşlardan verilen her bilgi 1 ve 0 kodlarına çevrilir. Her 0 ve 1, bit olarak; sekiz bitlik grup ise byte olarak tarif edilir. Bilgisayar, işlemlerini ikilik sayı sistemi ile yapar. İşlemler çok sade ve basit olmakla beraber çok hızlıdır.

0 1



Bit	En küçük bilgi parçası
Byte	8 bit = 1 Byte

ASCII


return (
<React.Fragment>

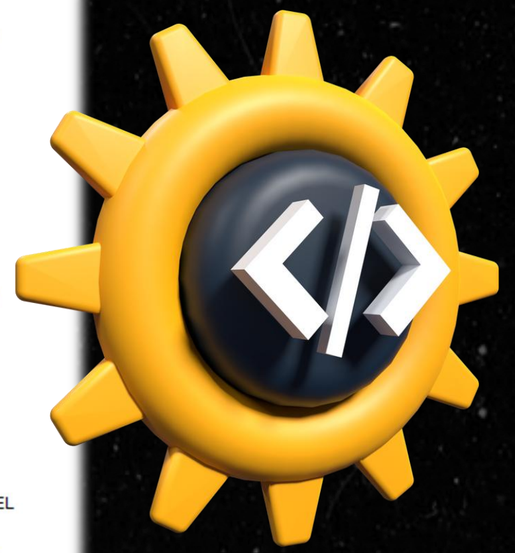
<Title name="our" title= "product

<ProductConsumer>
{(value) => {
console.log(value)
}}
</ProductConsumer>

</React.Fragment>

Decimal - Binary - Octal - Hex – ASCII Conversion Chart

Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII
0	00000000	000	00	NUL	32	00100000	040	20	SP	64	01000000	100	40	@	96	01100000	140	60	`
1	00000001	001	01	SOH	33	00100001	041	21	!	65	01000001	101	41	A	97	01100001	141	61	a
2	00000010	002	02	STX	34	00100010	042	22	"	66	01000010	102	42	B	98	01100010	142	62	b
3	00000011	003	03	ETX	35	00100011	043	23	#	67	01000011	103	43	C	99	01100011	143	63	c
4	00000100	004	04	EOT	36	00100100	044	24	$	68	01000100	104	44	D	100	01100100	144	64	d
5	00000101	005	05	ENQ	37	00100101	045	25	%	69	01000101	105	45	E	101	01100101	145	65	e
6	00000110	006	06	ACK	38	00100110	046	26	&	70	01000110	106	46	F	102	01100110	146	66	f
7	00000111	007	07	BEL	39	00100111	047	27	'	71	01000111	107	47	G	103	01100111	147	67	g
8	00001000	010	08	BS	40	00101000	050	28	(	72	01001000	110	48	H	104	01101000	150	68	h
9	00001001	011	09	HT	41	00101001	051	29	)	73	01001001	111	49	I	105	01101001	151	69	i
10	00001010	012	0A	LF	42	00101010	052	2A	*	74	01001010	112	4A	J	106	01101010	152	6A	j
11	00001011	013	0B	VT	43	00101011	053	2B	+	75	01001011	113	4B	K	107	01101011	153	6B	k
12	00001100	014	0C	FF	44	00101100	054	2C	,	76	01001100	114	4C	L	108	01101100	154	6C	l
13	00001101	015	0D	CR	45	00101101	055	2D	-	77	01001101	115	4D	M	109	01101101	155	6D	m
14	00001110	016	0E	SO	46	00101110	056	2E	.	78	01001110	116	4E	N	110	01101110	156	6E	n
15	00001111	017	0F	SI	47	00101111	057	2F	/	79	01001111	117	4F	O	111	01101111	157	6F	o
16	00010000	020	10	DLE	48	00110000	060	30	0	80	01010000	120	50	P	112	01110000	160	70	p
17	00010001	021	11	DC1	49	00110001	061	31	1	81	01010001	121	51	Q	113	01110001	161	71	q
18	00010010	022	12	DC2	50	00110010	062	32	2	82	01010010	122	52	R	114	01110010	162	72	r
19	00010011	023	13	DC3	51	00110011	063	33	3	83	01010011	123	53	S	115	01110011	163	73	s
20	00010100	024	14	DC4	52	00110100	064	34	4	84	01010100	124	54	T	116	01110100	164	74	t
21	00010101	025	15	NAK	53	00110101	065	35	5	85	01010101	125	55	U	117	01110101	165	75	u
22	00010110	026	16	SYN	54	00110110	066	36	6	86	01010110	126	56	V	118	01110110	166	76	v
23	00010111	027	17	ETB	55	00110111	067	37	7	87	01010111	127	57	W	119	01110111	167	77	w
24	00011000	030	18	CAN	56	00111000	070	38	8	88	01011000	130	58	X	120	01111000	170	78	x
25	00011001	031	19	EM	57	00111001	071	39	9	89	01011001	131	59	Y	121	01111001	171	79	y
26	00011010	032	1A	SUB	58	00111010	072	3A	:	90	01011010	132	5A	Z	122	01111010	172	7A	z
27	00011011	033	1B	ESC	59	00111011	073	3B	;	91	01011011	133	5B	[	123	01111011	173	7B	{
28	00011100	034	1C	FS	60	00111100	074	3C	<	92	01011100	134	5C	\	124	01111100	174	7C	|
29	00011101	035	1D	GS	61	00111101	075	3D	=	93	01011101	135	5D	]	125	01111101	175	7D	}
30	00011110	036	1E	RS	62	00111110	076	3E	>	94	01011110	136	5E	^	126	01111110	176	7E	~
31	00011111	037	1F	US	63	00111111	077	3F	?	95	01011111	137	5F	_	127	01111111	177	7F	DEL



ASCII BINARY ALPHABET

A	1000001	N	1001110
B	1000010	O	1001111
C	1000011	P	1010000
D	1000100	Q	1010001
E	1000101	R	1010010
F	1000110	S	1010011
G	1000111	T	1010100
H	1001000	U	1010101
I	1001001	V	1010110
J	1001010	W	1010111
K	1001011	X	1010111
L	1001100	Y	1011001
M	1001101	Z	1011010

Bilgisayar Nasıl Karar Verir?

- Bilgisayar Aritmatiksel ve Mantıksal işlemleri ALU birimi sayesinde gerçekleştirir.
- Bilgisayar sadece matematiksel verileri işleyip saymakla kalmaz aynı zamanda mantısal karşılaştırma yaparak karar da verir. Bu işlemler için boolean matematiği denilen matematik kurallarını kullanır.
- Çeşitli şartlara göre bilgisayar, EVET, HAYIR, VE ,VEYA, DEĞİL gibi kararlar alabilir.

Öğr. Gör. Furkan DURMUŞ

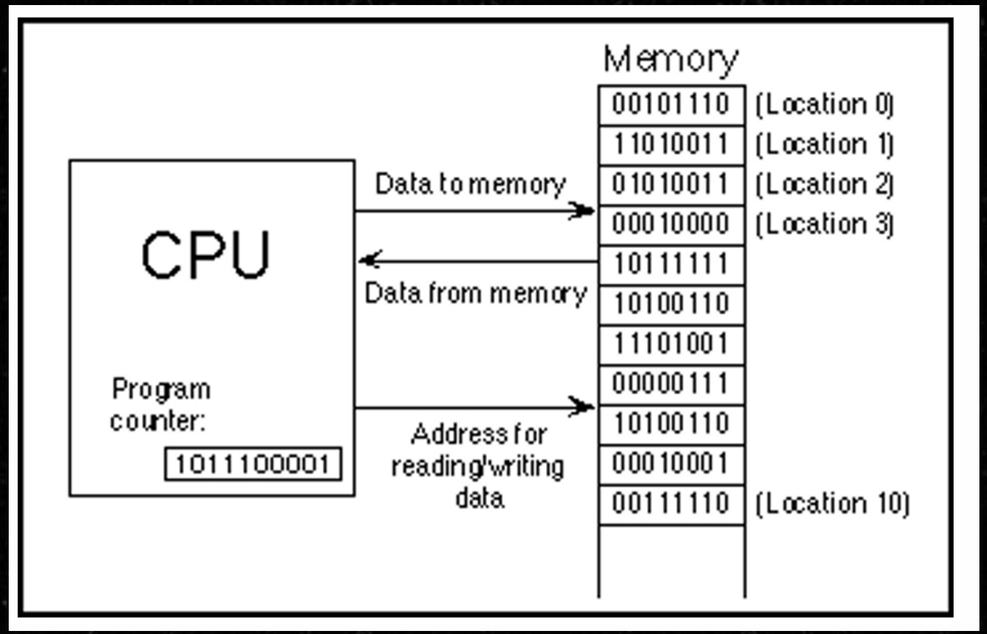
Bilgisayar ve İnsan

Makine Dili (En Düşük Seviye)

Makine dili mikroişlemci ya da mikrodenetleyici gibi komut işleme yeteneğine sahip entegrelerin işleyebilecekleri komutlardan ve buna uygun söz diziliminden oluşan dile verilen addır. Makine dili, işlemcinin verilen komutlar doğrultusunda çalıştırılmasını sağlayan ve işlemci mimarisine göre değişen en alt seviyedeki programlama dilidir.

Makine Dili (En Düşük Seviye)

Bu dil sadece 0 ve 1 ikililerinin anlamlı kombinasyonlarından meydana gelmektedir. Bu sebeple anlaşılması çok güçtür.



Makine Dili (En Düşük Seviye)

Sonuç olarak bilgisayar programı hangi dilde olursa olsun formatı makine diline çevrilmesi gereklidir. Her emrin yerine getirilmesi 1 cycle denilen 4 ana kısımdan oluşur.

Fetch

Bir sonraki emri hafızadan getir.

Decode

Emrin ne demek istediğini çöz.

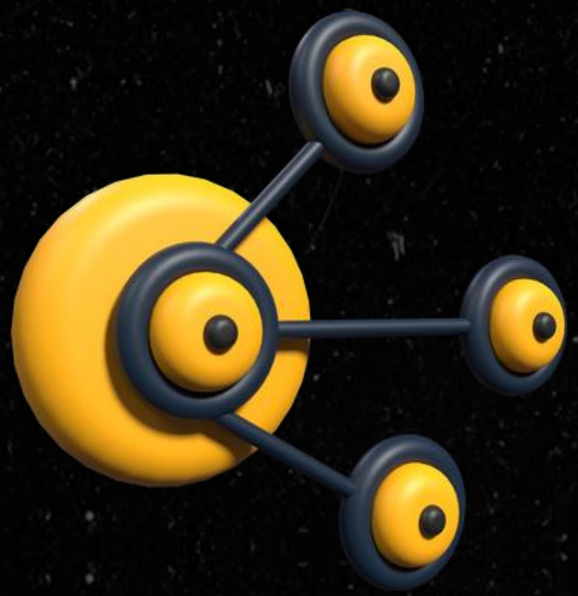
Execute

Emir yerine getirilir, işlem yapılır.

Store

Sonucu sakla.

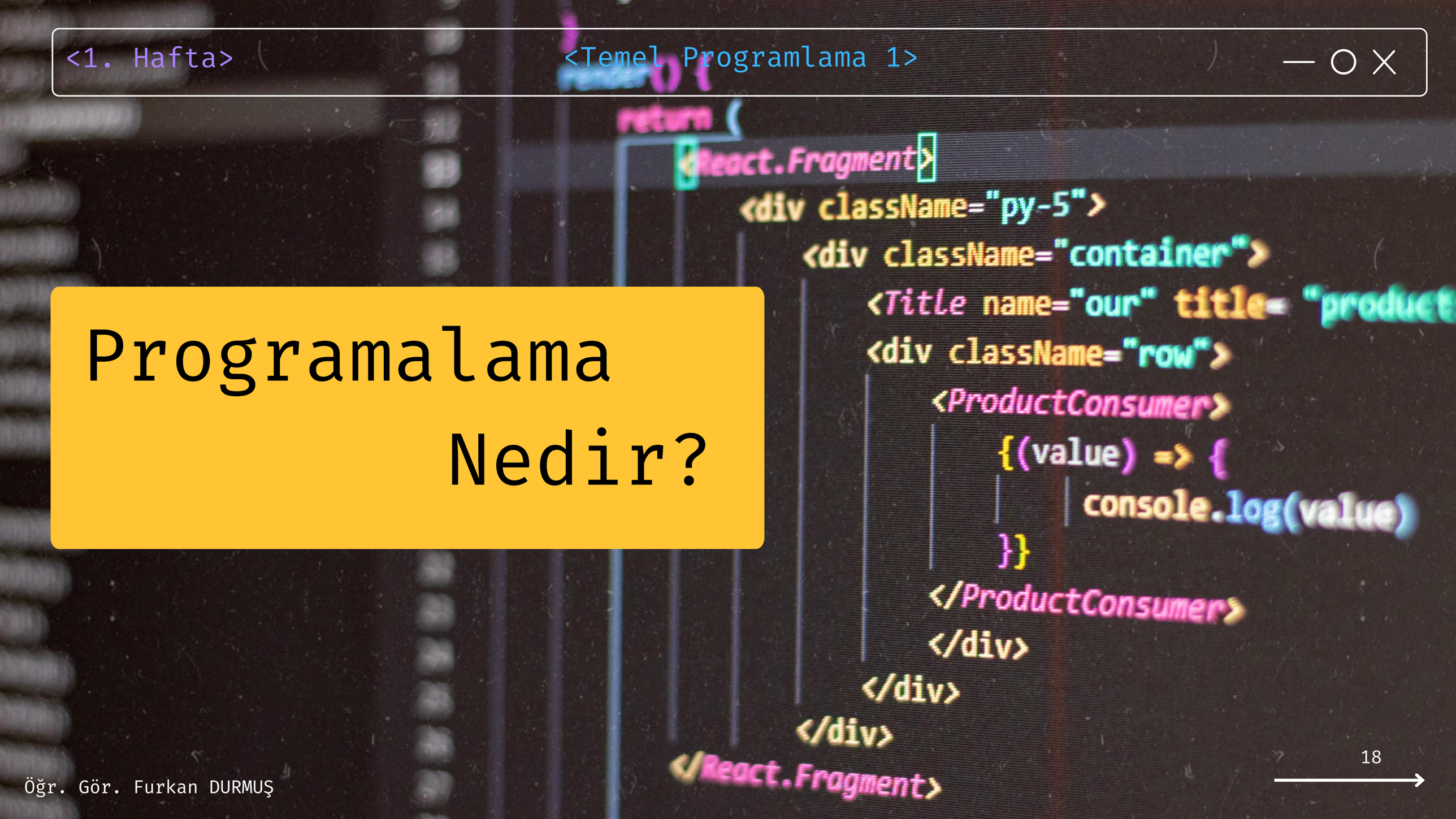
Sayı Sistemleri



Bilgisayar ortamında dört çeşit sayı sistemi kullanılmaktadır. Bunlar;

İkilik sayı sistemi(Binary)
Sekizlik sayı sistemi(Oktal)
Onluk sayı sistemi(Desimal)
Onaltılık sayı sistemi(Heksadesimal)

Programalama Nedir?

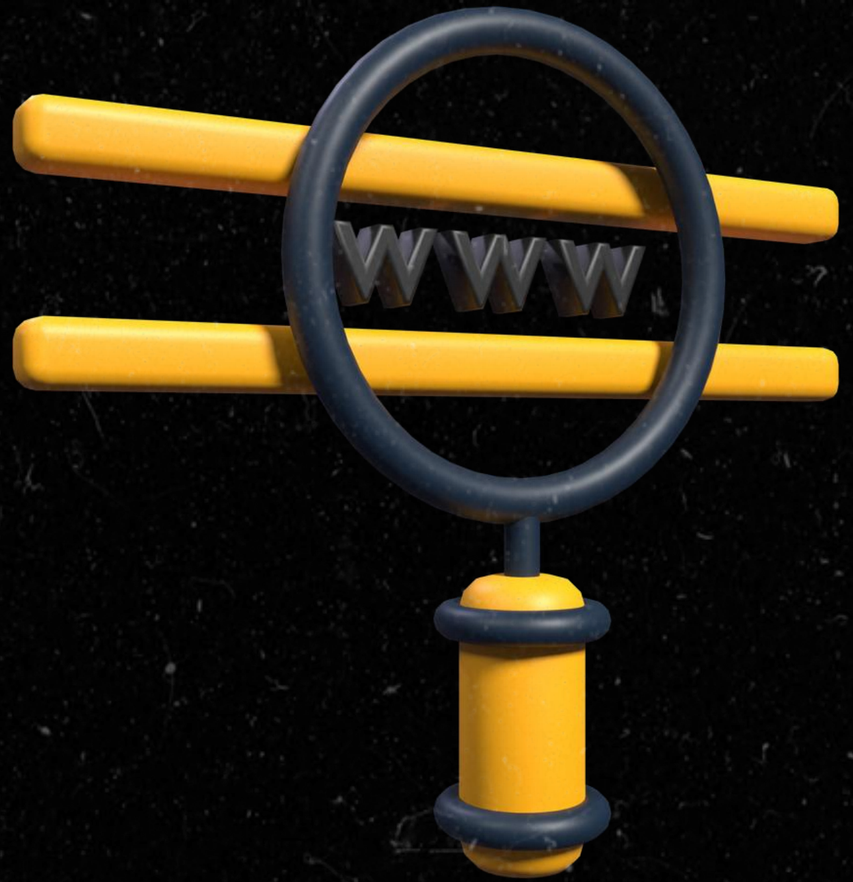


{01} Programlama

Programlama ya da diğer adıyla yazılım , bilgisayarın donanıma nasıl davranacağını anlatan, bilgisayara yön veren komutlar, kelimeler, aritmetik işlemlerdir.

Diğer bir tanım verecek olursak programlama, bilgisayar programlarının edilmesi ve bakımının yapılması sürecine verilen isimdir.

Programlama, bir proglamlama yazılması, test dilinde yapılır. Bu programlama dili Java, C# ve Python gibi yüksek seviyede bir dil olabileceği gibi C, assembly ve bazı durumlarda makine dili de olabilir.



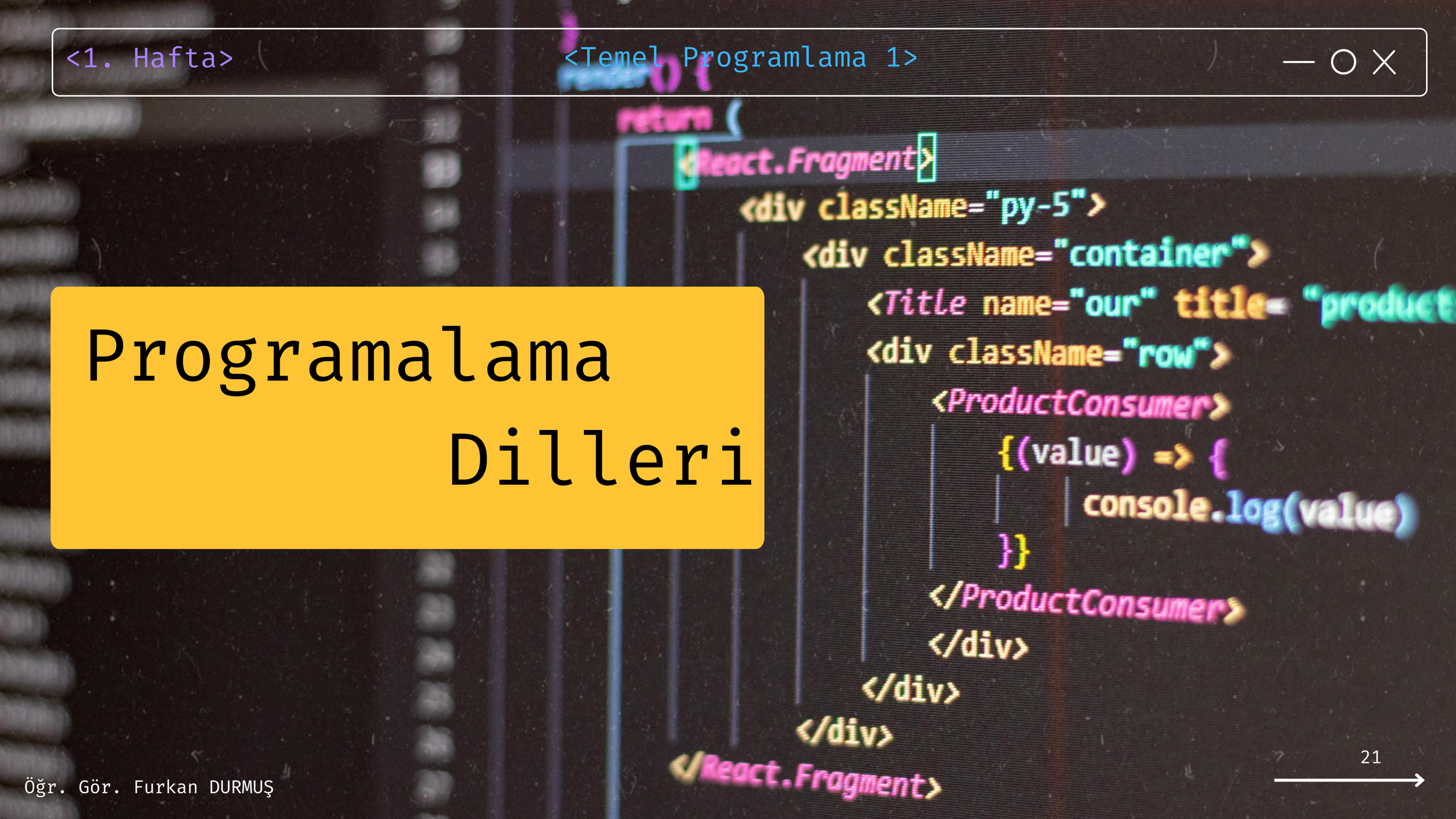
{01} Programlama

Derleyici, yazılan programları okuyup içerisinde mantıksal veya yazınsal hatalar olup olmadığını bulan, bulduğu hataları kullanıcıya göstererek programın düzeltilmesine yardım eden, hata yoksa programı çalıştırıp sonucunu gösteren platformdur.

Programcılar genelde programlamayı gerçek hayata benzetirler. Bir program yazmak veya bir problemi çözmek için öncelikle komutları unutmak ve çözümü gerçek hayatta yapıyormuş gibi düşünmek gerekir onlara göre. Komutlar sadece araçtır.

Programlamaya başlayanların kendi dilleriyle Merhaba Dünya (Genelde : Hello World!) yazmalarıyla başlar.

Programalama Dilleri




Language
Pascal
ALGOL
Assembly
C
C++
shell
Visual
Go
JavaScript
Lisp
VB
Java
PEARL
Oxygene
Script
PHP
Python
Ruby
Programming
Delphi
DASL
Ajax
Object
Prolog
Caml
Common
REXX
EXEC
NET
Microsoft
ABAP
SYMPL
C/AL
Cg
Cesil
Scala
Rust
GDL
JOSS
Bistro
Genie
Inform
HAL/S
SIMSCRIPT
Revolution
SenseTalk
Sawzall
Blue
Chapel
Rapira
CSKA
SIMPLE
Agora
Clojure
Autolt
Microcode
SAM76
Redcode
code
Lynx
Logo
Trac
Not
E
language
Adenine
DATATRIEVE
PL/I
ELAN
SALSA
ATS
Fjölnir
Fuxi
TACPOL
FP
K
Fancy
SuperTalk
P
LC-3
L
XQTcl
dc
Cayenne
ABC
MicroScript
Squirrel
Windows/Dos
Kaleidoscope
IMP
FLJavaFX
R
Forth
D
b
T
Newspeak
GameMonkey
ECMAScript
GJ
JASS
bc
GM
Constraint
G
S
EusLisp
ObjectScript
BLISS
notation
Cyclone
RPL
Agena
CorVision
Action
Little
Simula
Boomerang
COBOL
GraphTalk
PL/SQL
ChucK
A
F
M
RPG
Strand
Turing
ISWIM
S/SL
rex
Robot
SP/k
HaXe
Eiffel
J
Q
Arc
Moby
ORCA/Modula-2
Uniface
Coral
WATFOR

{02} Programlama Dilleri

Programlama dili, yazılımcının bir algoritmayı amacıyla, bir bilgisayara ifade etmek ne yapmasını istediğini anlatmasının tektipleştirilmiş yoludur.

Programlama dilleri, yazılımcının bilgisayara hangi veri üzerinde işlem yapacağını, verinin nasıl depolanıp iletileceğini, hangi koşullarda hangi işlemlerin yapılacağını tam olarak anlatmasını sağlar.

Şu ana kadar 150'den fazla programlama dili geliştirilmiştir. Bunlardan bazıları Pascal , Basic Python , Ada , Fortran , Delphi , C , C #, C ++, Java C Enterprise'dır.

Öğr. Gör. Furkan DURMUŞ

Neden programlama diline ihtiyacımız var?

Neden Programlama Dilleri

Temelde bilgisayarlar 1 ve 0'lardan oluşan ikilik sayı sistemindeki dili anlarlar. Ancak bu dilde bir programın yazılması çok zordur.

Hata yapma olasılığı daha çoktur.

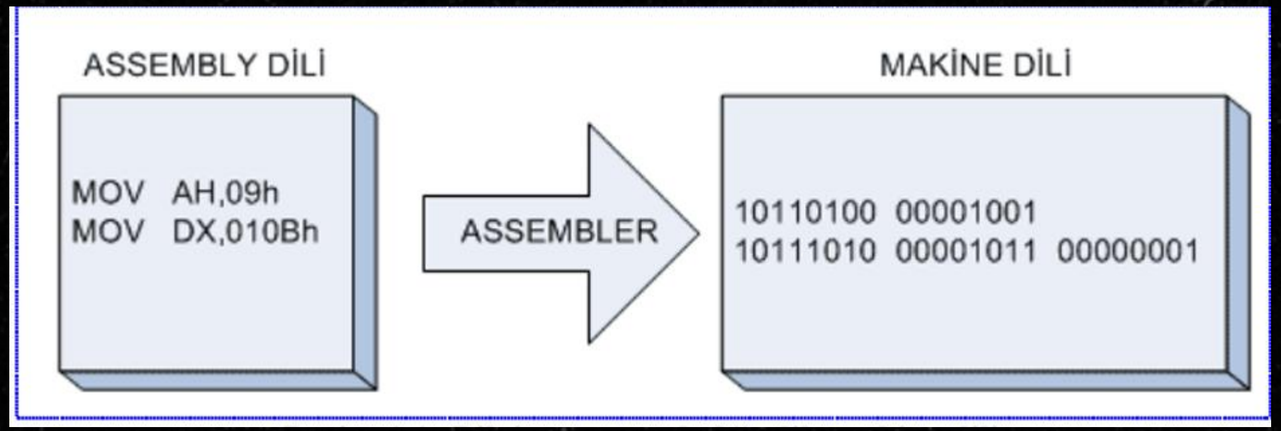
Yazılımı geliştirmesi çok uzun zaman alır.

Bu sebeple, Programcılar 1 ve 0 ile program yazma yerine 1949'lu yıllarda Assembly dili'ni geliştirmişlerdir. Fakat bilgisayarın bu yazılanlarla ilgili hiçbir fikri yoktur. Bu sebeple programcılar "assembly dil" komutlarını "makine dili" ne çeviren programlar yazmışlardır. Bu programlara "makine dili çeviricisi - assembler" denir. Böylece "assembly dili" ile yazılmış bir kod, bilgisayarın anlayabileceği "makine diline dönüştürülür.

{03} Assembly Dili

Assembly dilinde kod örneği

```
Title Yazi Programi
dosseg
.model small
.stack 100h
.data
my_message db 'Selam!',0dh,0ah, '$'
.code
main proc
            mov ax,@data
            mov ds,ax
            mov ah,9
            mov dx,offset my message
            int 21h
            mov ax,4C00h
            int 21h
main endp
end main
```

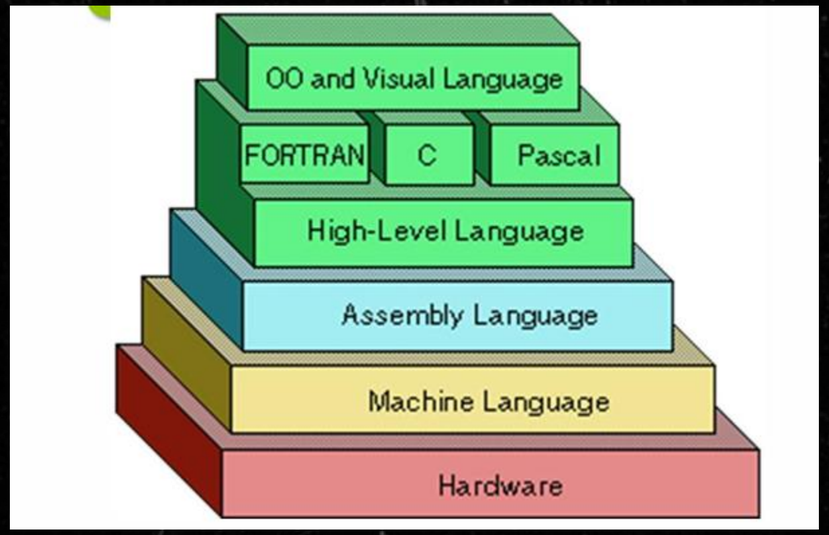


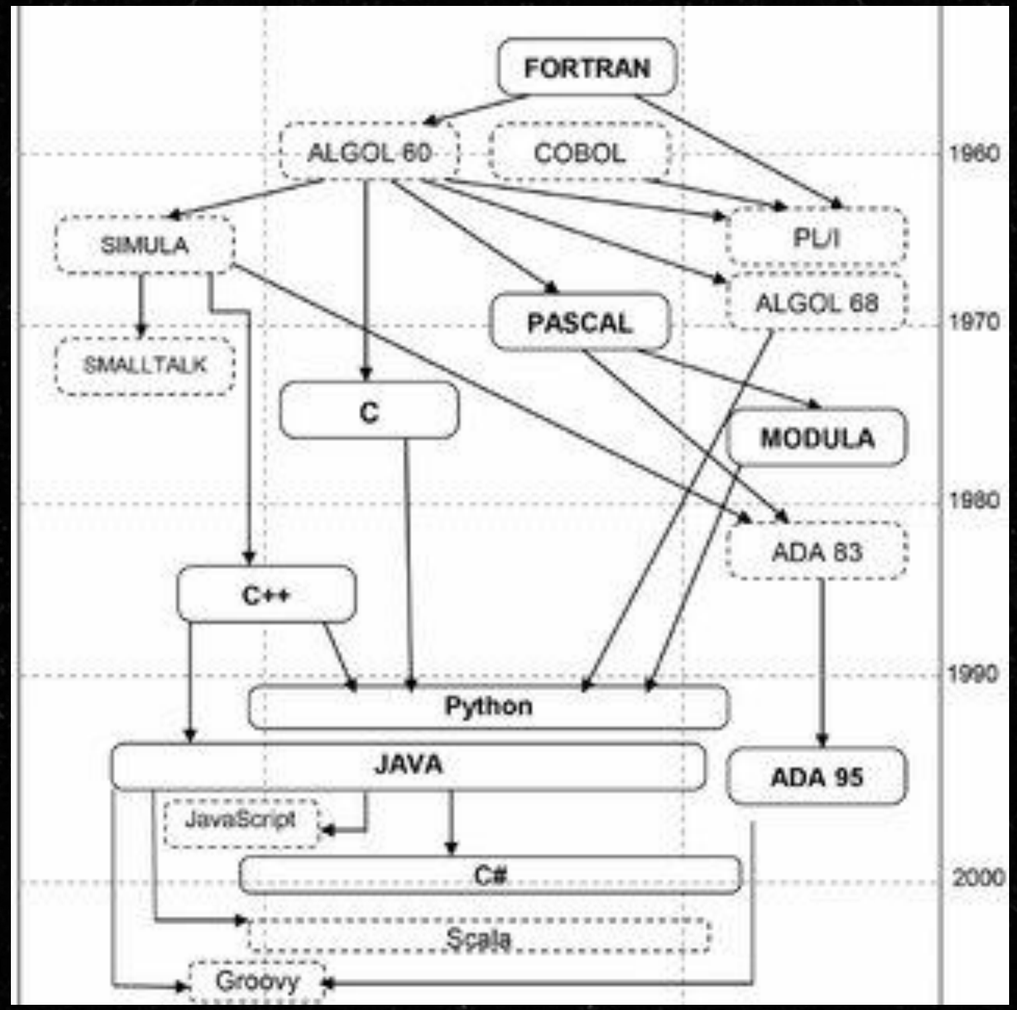
{04} Diğer Programlama Dilleri

Programcılar donanıma erişimi daha kolay olan, okunaklı, yazımı ve düzenlemesi kolay olan bir dile ihtiyaç duymuşlardır. Bunun üzerine C (1972) taşınabilir dili geliştirilmiştir. Cobol (1959) ve Fortran (1957) gibi birçok diller de vardır ama hala günümüzde yaygın olarak kullanılan C dili olmuştur.

İnsanların konuşma diline yakın olan dillere “yüksek seviye dil”, donanıma yakın dillere de ‘düşük seviye dil’ denilmiştir.

Yüksek ve düşük seviyeli diller kendi içinde avantaja ve dezavantaja sahiptir.





Öğr. Gör. Furkan DURMUŞ

Yüksek Seviyeli Dillerin Özellikleri

Yüksek seviyeli dillerin genel özelliklerini şöyle özetleyebiliriz:

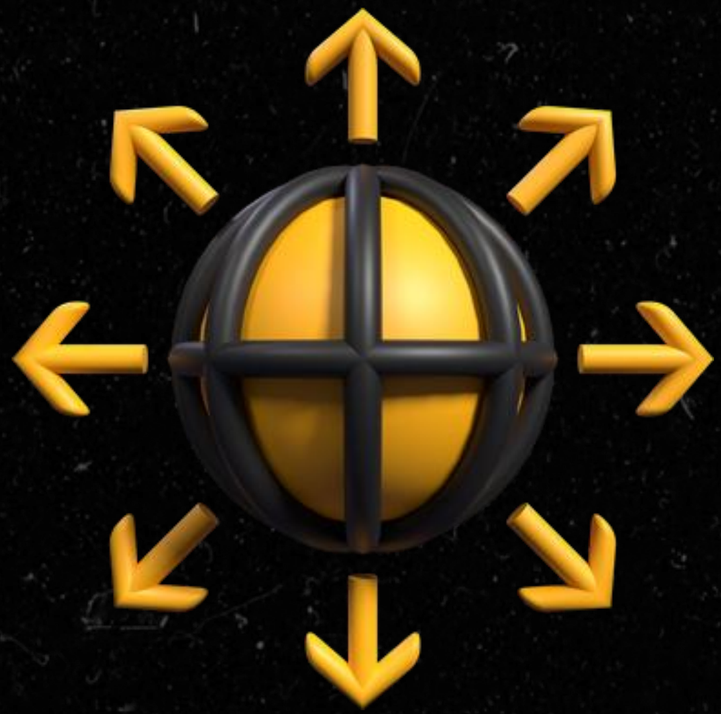
“Makine diline” göre daha şişkin ve yavaş kod meydana getirirler.

Tüm sistem kaynaklarına ulaşılamayabilir,

Bir derleyici gereklidir.

Kısa zamanda program yazmaya başlanır.

Öğrenme ve ustalaşma için geçen zaman fazla değildir.

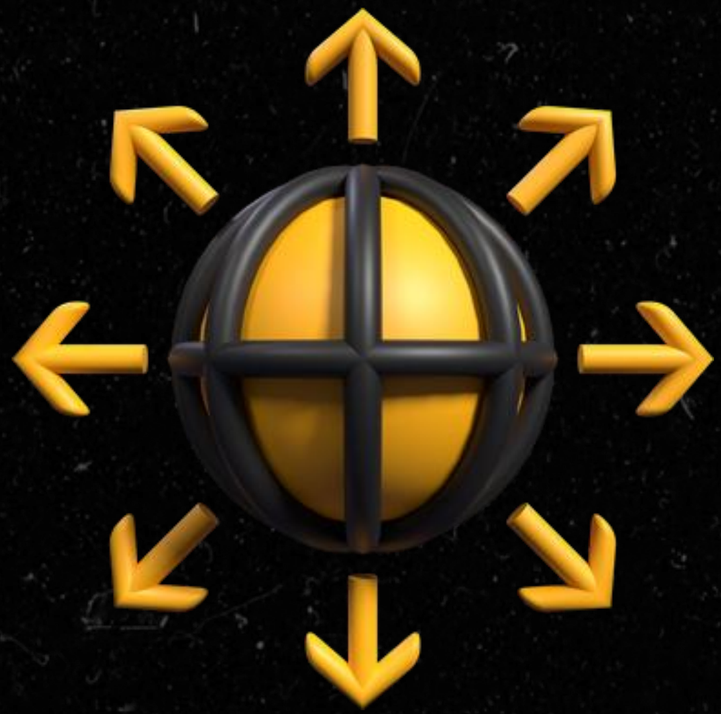


Yüksek Seviyeli Dillerin Özellikleri

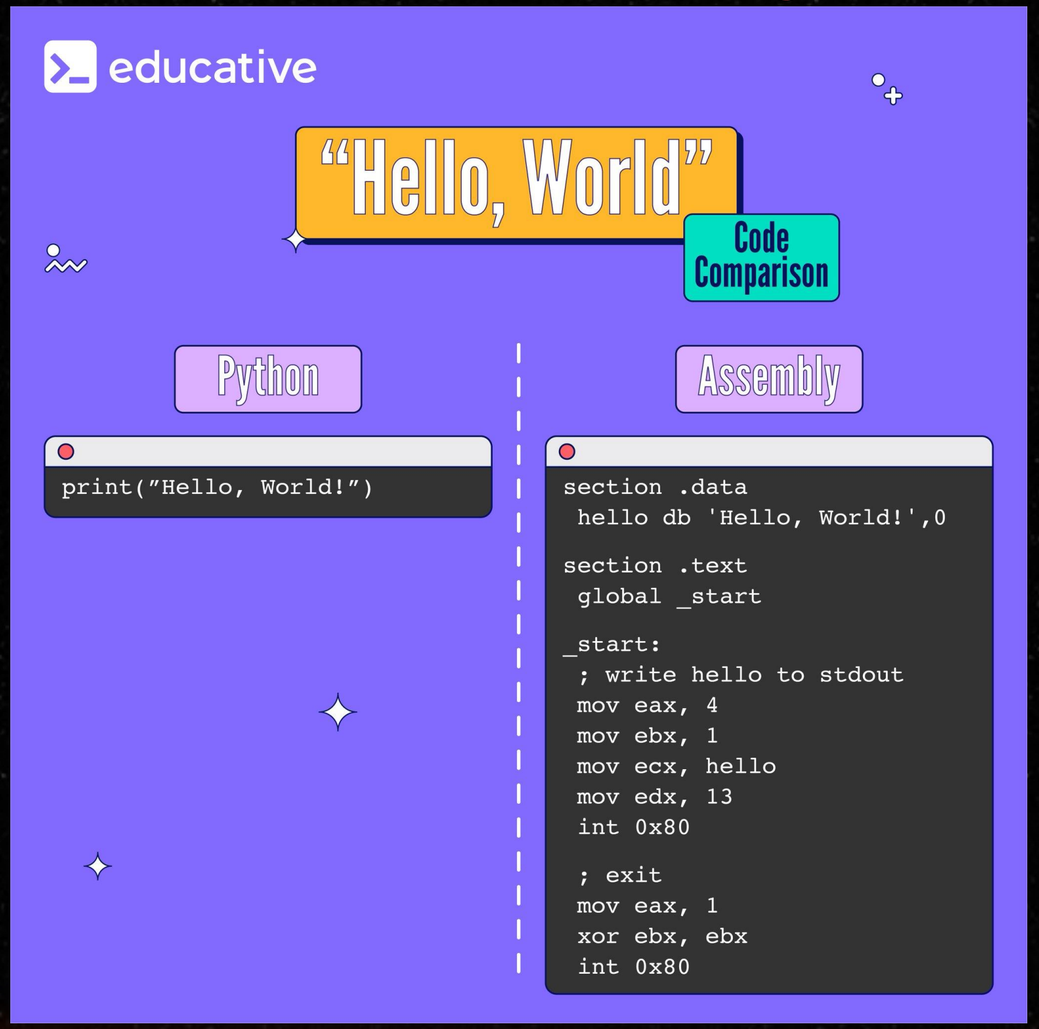
Yanlışlıkla sistem kaynaklarının bozulmaması için kalkanları vardır.

Okuması ve değiştirmesi kolaydır.

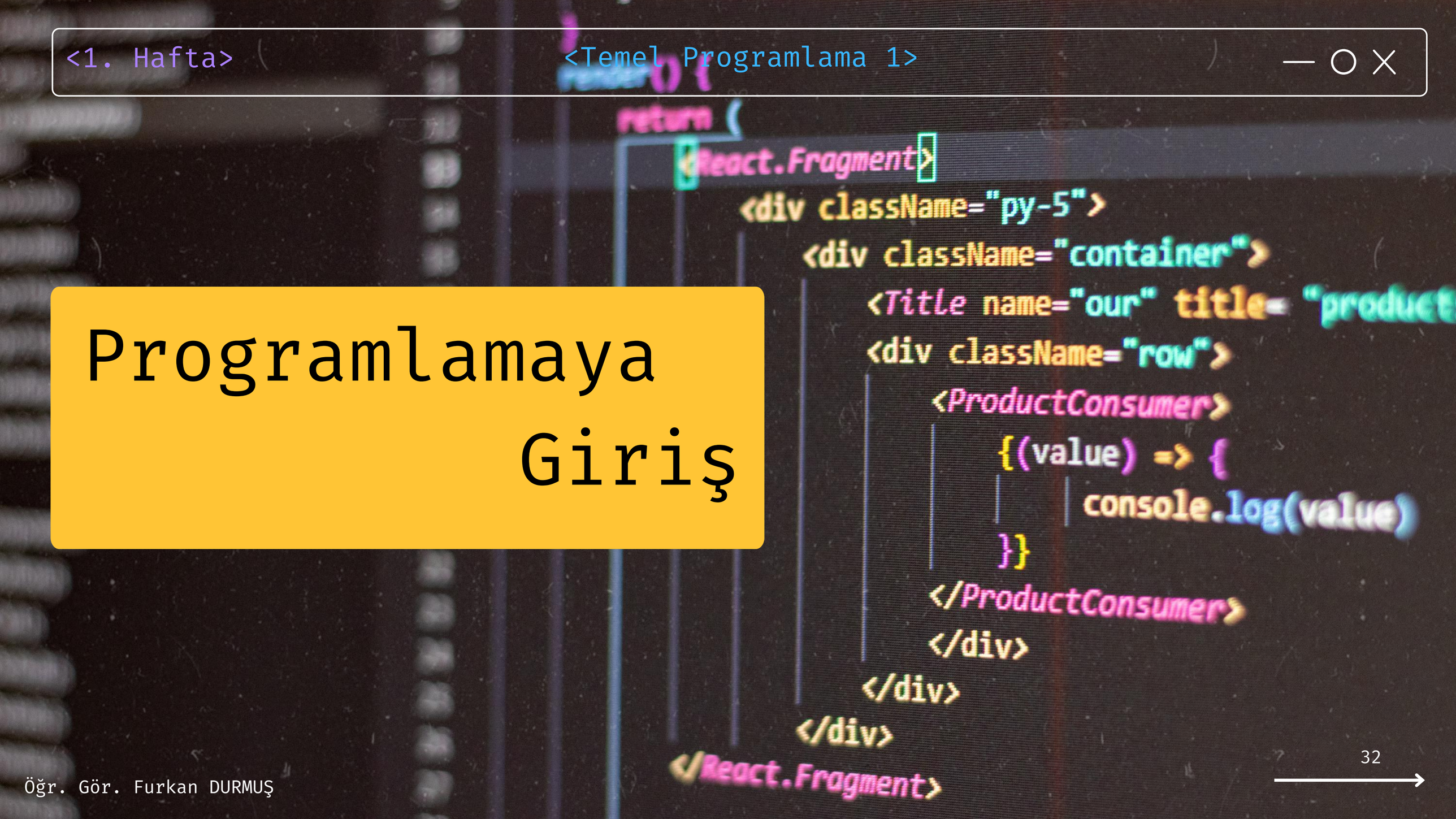
Başka bilgisayar çeşitlerinde de çalışabilirler, yani taşınabilirler.



Yüksek Seviyeli vs Düşük Seviyeli

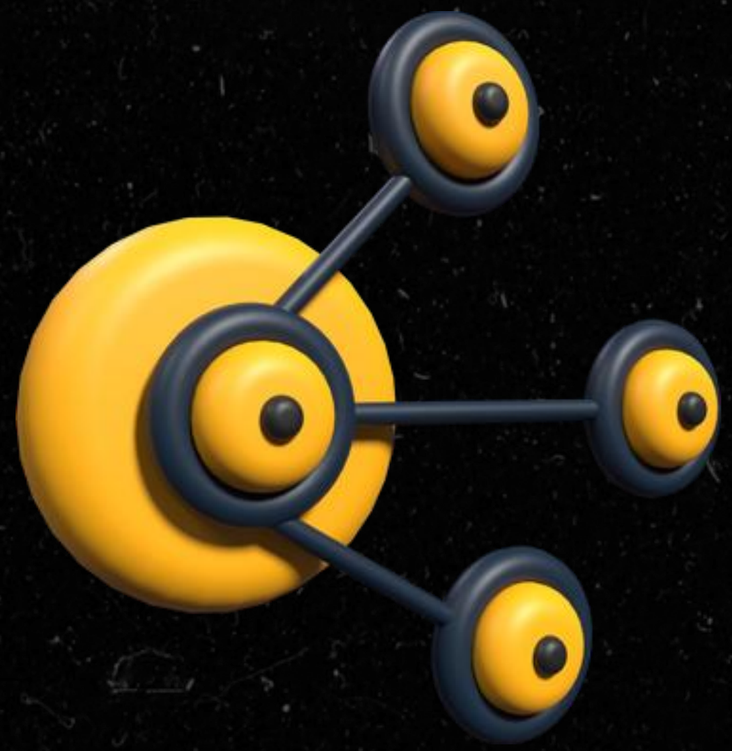


Programlamaya Giriş



{05} Programlamaya Giriş

İyi bir yazılımcı nasıl olmalıdır ?



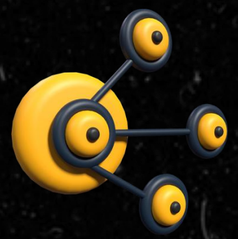
Öğr. Gör. Furkan DURMUŞ

İyi bir yazılımcı nasıl olmalıdır ?

İyi bir programcı sadece kod yazan kişi değildir. Aşağıdaki özellikler iyi bir programcıyı tanımlar:

Problem çözme yeteneği: Sorunu doğru analiz edip etkili çözüm yolları geliştirebilmelidir.

Analitik düşünme: Karmaşık yapıları sadeleştirebilmeli ve mantıklı yapılar kurabilmelidir.



Temiz ve okunabilir kod yazma: Kodun anlaşılır, modüler ve sürdürülebilir olması önemlidir.

Sürekli öğrenme: Teknoloji sürekli gelişiyor. Yeni dilleri, araçları ve yaklaşımları öğrenmeye açık olmalıdır.

Takım çalışması: Yazılım geliştirme genellikle ekip işidir. Etkili iletişim kurabilmelidir.

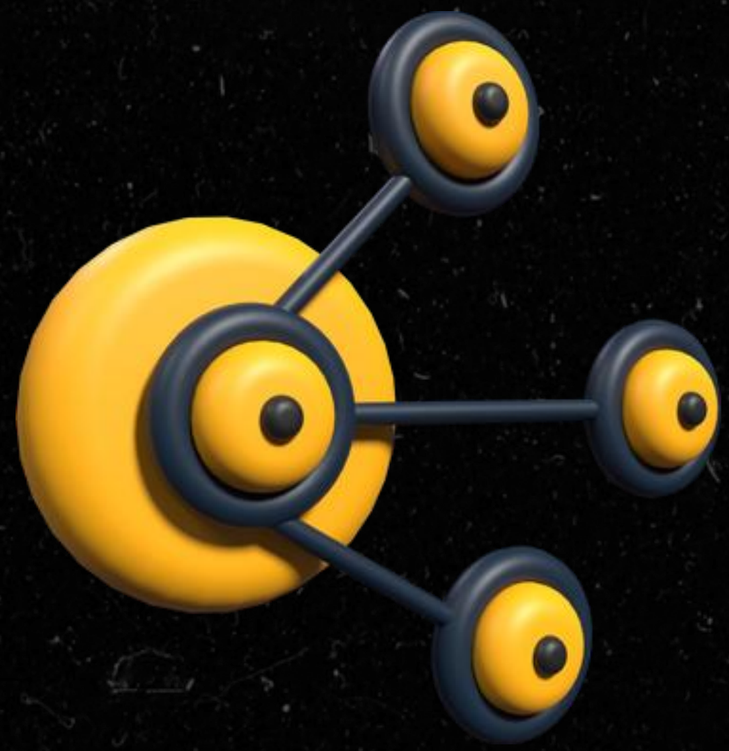
İyi bir yazılımcı nasıl olmalıdır ?

Test ve hata ayıklama becerisi: Yazdığı kodun doğru çalışıp çalışmadığını kontrol edebilmeli ve hataları çözebilmelidir.

Dokümantasyon: Kodun nasıl çalıştığını ve ne yaptığını açıklayan belgeler oluşturmalıdır.

{05} Programlamaya Giriş

Sadece programı yazıyor olmak iyi bir yazılımcı olduğunuz anlamına gelir mi?



Öğr. Gör. Furkan DURMUŞ

Sadece programı yazıyor olmak iyi bir yazılımcı olduğunuz anlamına gelir mi?

Sadece programı yazmak iyi bir yazılımcı olduğunuz anlamına gelmez. Çünkü Yazılım geliştirme süreci sadece "kod yazmak"tan ibaret değildir.

Eğer sadece kod yazarsanız ama:

Sistemin ihtiyaçlarını tam analiz etmezseniz,
Kullanıcı dostu bir arayüz oluşturmazsanız,
Güvenlik açıkları bırakırsanız,
Test yapmazsanız,
Dokümantasyon yazmazsanız,
Sürdürülmesi ve geliştirilmesi zor bir yapı kurarsanız,

O zaman sistem çalışsa bile iyi bir yazılım sayılmasınız.

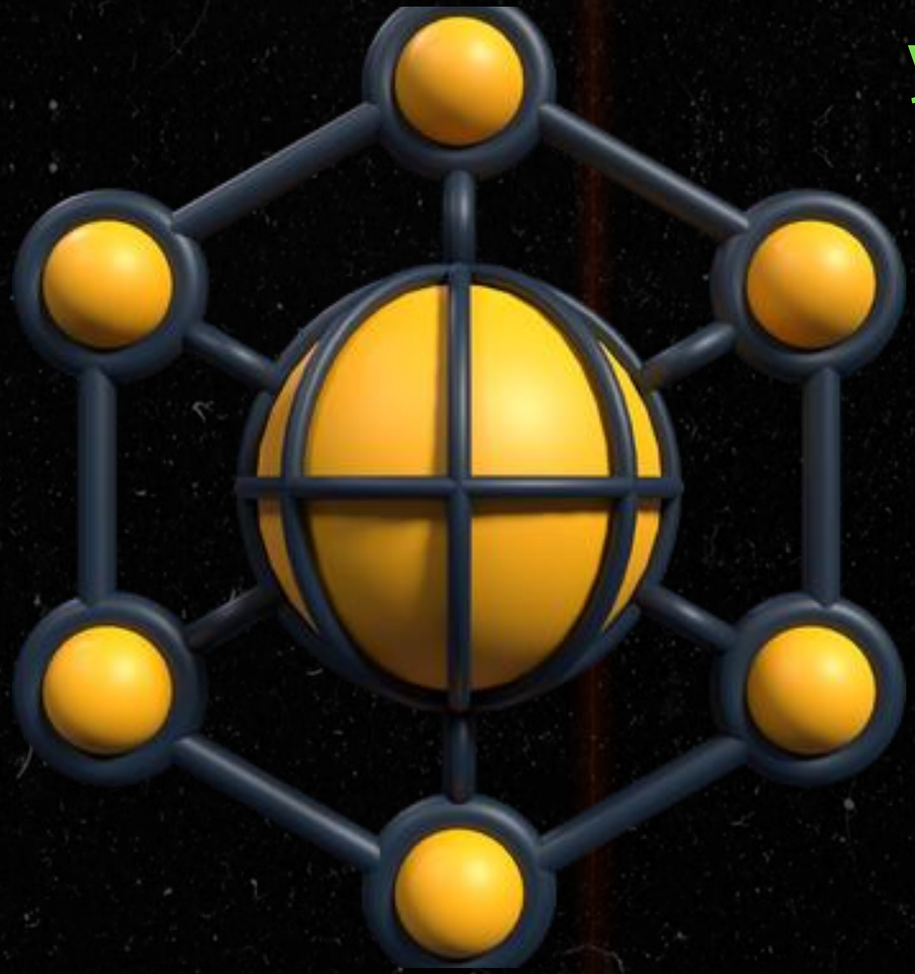
Programlamaya Giriş

Programlamaya başlamadan önce yapılması gerekenler nelerdir?

Programlamaya başlamadan önce yapılması gerekenler nelerdir?

{01} İhtiyaç Analizi:

- Program ne işe yarayacak?
- Kimler kullanacak?
- Hangi sorunları çözecek?



Programlamaya başlamadan önce yapılması gerekenler nelerdir?

{02} Sistem Tasarımı (Planlama):

- Arayüz nasıl olacak?
- Hangi modüller olacak?
- Veritabanı yapısı nasıl olacak?

Programlamaya başlamadan önce yapılması gerekenler nelerdir?

{03} Algoritma ve Akış Diyagramı:

- İşleyiş sırası nasıl olacak?
- Hangi durumlarda hangi işlemler yapılacak?

Programlamaya başlamadan önce yapılması gerekenler nelerdir?

{04} Teknoloji Seçimi:

- Hangi programlama dili, veritabanı, çerçeve (framework) vb. kullanılacak?

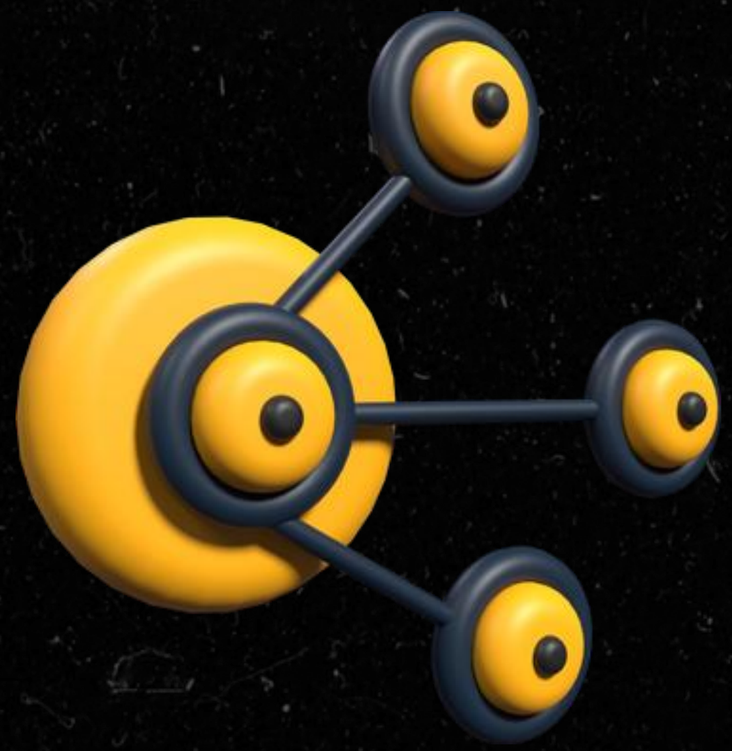
Programlamaya başlamadan önce yapılması gerekenler nelerdir?

{05} Zaman ve kaynak planlaması:

- Ne kadar sürede geliştirilecek?
- Kim, neyi yapacak?

{05} Programlamaya Giriş

İlk yapılması gereken programı yazacağımız dili belirleyip kodlamaya başlamak diyebilir miyiz?



{05} Programlamaya Giriş

Sıra	Yapılacaklar	Açıklama
1	Problemi ve ihtiyaçları analiz et	Ne yazılacak? Neden? Kim için?
2	Sistem tasarımı yap	Hangi özellikler olacak? Veri yapısı nasıl?
3	Akış ve algoritmalar oluştur	Mantık nasıl işleyecek?
4	Kullanılacak teknolojileri belirle	Hangi dil, veritabanı, araçlar?
5	Zaman ve kaynak planlaması yap	Ne kadar sürede kim neyi yapacak?
6	Kodlamaya başla	Artık uygulamayı yazma zamanı!

{06} Yazılımın Yaşam Döngüsü

Planlama

Personel ve donanım gereksinimlerinin çıkarıldığı, fizibilite çalışmasının yapıldığı ve proje planının oluşturulduğu aşamadır.

{06} Yazılımın Yaşam Döngüsü

Analiz

Sistem gereksinimlerinin ve işlevlerinin ayrıntılı olarak çıkarıldığı aşama. Var olan işler incelenir, temel sorunlar ortaya çıkarılır.

{06} Yazılımın Yaşam Döngüsü

Tasarım

Belirlenen gereksinimlere yanıt verecek yazılım sisteminin temel yapısının oluşturulduğu aşamadır.

- Mantıksal; önerilen sistemin yapısı anlatılır (akış şemaları)
- Fiziksel; yazılımı içeren bileşenler ve bunların ayrıntıları (ekran tasarımları)

{06} Yazılımın Yaşam Döngüsü

Uygulama

Kodlamaların yapıldığı adımdır.

{06} Yazılımın Yaşam Döngüsü

Test ve Birleştirme

Yazılım parçalarının bir bütün haline getirilmesi yani database, api gibi farklı modüllerin birleştirilmesi ve sistemin genel testinin yapıldığı aşamadır.

{06} Yazılımın Yaşam Döngüsü

Bakım ve Sürdürülebilirlik

Teslimden sonra hata giderme ve yeni eklentiler yapma aşamasıdır.

Son