

IEEE

METUNCC

# C Programlama

```
11011000010101100100010000111000100111
0100110010110100110110100111101111011110
00011010001000111010001101000011010
0100100110100001010001110
10001001int main()
010101001{
1110011000    printf("Hello World");
001000001111    return 42;
00011010001000111010010110001101000011010
01001001101111010111011110000001010001110
```

# Syntax

`/*Çok satırlı bir açıklama.`

`Yıldızlar arasında kalan bütün alan, yorum olarak değerlendirilir ve derleyici (compiler) tarafından işlenmez.`

`*/`

`#include<stdio.h>`

`int main()`

`{`

`//Tek satırlık bir açıklama.`

`printf("Hello World\n");`

`return 0;`

`}`

Çok satırlı yorum

Standart kütüphane

Programın çalışmaya başladığı yer

Programın başarılı bir şekilde çalıştığını belirtir.

# Girinti

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Ali: \"Naber, nasılsın?\" dedi.\n");
```

```
    return 0;
```

```
}
```

# Değişken nedir? Tanımı nasıl yapılır?

Değişkenler, girdiğimiz değerleri alan veya programın çalışmasıyla bazı değerlerin atandığı, veri tutucularıdır.

Değişken tanımlamaysa, gelecek veya girilecek verilerin ne olduğuna bağlı olarak, değişken tipinin belirlenmesidir. Yani a isimli bir değişkeniniz varsa ve buna tam sayı bir değer atamak istiyorsanız, a değişkenini tam sayı olarak tanıtmamız gerekir.

# Örnek 1

```
#include<stdio.h>
```

```
int main( void )
```

```
{
```

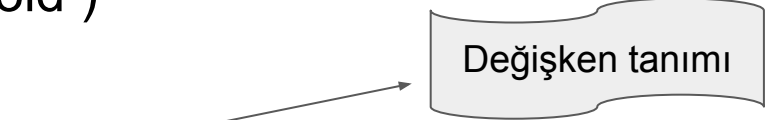
```
    int a;
```

```
    a = 25;
```

```
    printf("a sayısı %d",a);
```

```
    return 0;
```

```
}
```



Değişken tanımı

## Örnek 2

/\* İki tam sayıyı toplayan C kodu\*/

#include <stdio.h>

int main()

{


int ilkSayi = 35, ikinciSayi = 21, toplam;

toplam = ilkSayi + ikinciSayi;

printf("İlk sayi %d, ikinci sayi %d, toplam %d\n", ilkSayi, ikinciSayi, toplam);

return 0;

}



Değişken tanımlı

# Özel Kelimeler

**Table 2.1** Keywords

auto	default	float	register	struct	volatile
break	do	for	return	switch	while
case	double	goto	short	typedef	
char	else	if	signed	union	
const	enum	int	sizeof	unsigned	
continue	extern	long	static	void	

Tabloda geçen kelimeler C dilinde özel anlamları temsil ettiğinden dolayı değişken ismi olarak kullanılamaz.



# Veri Tipleri

1- int : Tam sayılar için

2- char : Karakterler için

3- float: Kesirli sayılar için

**Table 2.3** C's Built-in Data Types

Data Type	Supplied Operations
Integer	<code>+, -, *, /,</code> <code>%, =, ==, !=,</code> <code>&lt;=, &gt;=, sizeof(),</code> and bit operations (see Sec. 14.2)
Floating Point	<code>+, +, -, *, /,</code> <code>=, ==, !=,</code> <code>&lt;=, &gt;=, sizeof()</code>

# Tam sayı(Integer) Veri Tipi

- **int:** tüm tam sayılar (integers)
- Örnek: 0, -10, 253, -26351
- printf ve scanf komutlarında tam sayılar için %d kullanılmalıdır.

# Karakter (char) Veri Tipi

- **char:** özgün karakterleri depolar (ASCII)
- Örnek: 'A', '\$', 'b', '!'
- printf ve scanf komutlarında karakterler için %c kullanılmalıdır.

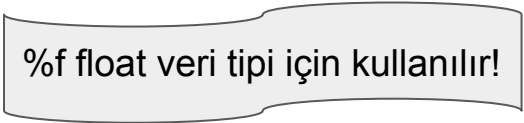
0	<NUL>	32	<SPC>	64	@	96	`	128	Ä	160	†	192	¿	224	‡
1	<SOH>	33	!	65	A	97	a	129	Å	161	°	193	¡	225	•
2	<STX>	34	"	66	B	98	b	130	Ç	162	¢	194	¬	226	,
3	<ETX>	35	#	67	C	99	c	131	É	163	£	195	√	227	"
4	<EOT>	36	\$	68	D	100	d	132	Ñ	164	§	196	ƒ	228	%
5	<ENQ>	37	%	69	E	101	e	133	Ö	165	•	197	≈	229	Â
6	<ACK>	38	&	70	F	102	f	134	Ü	166	¶	198	Δ	230	Ê
7	<BEL>	39	'	71	G	103	g	135	á	167	ß	199	«	231	Á
8	<BS>	40	(	72	H	104	h	136	à	168	®	200	»	232	Ë
9	<TAB>	41	)	73	I	105	i	137	â	169	©	201	...	233	È
10	<LF>	42	*	74	J	106	j	138	ä	170	™	202		234	Í
11	<VT>	43	+	75	K	107	k	139	ã	171	'	203	À	235	Î
12	<FF>	44	,	76	L	108	l	140	å	172	"	204	Ã	236	Ï
13	<CR>	45	-	77	M	109	m	141	ç	173	≠	205	Ö	237	Ì
14	<SO>	46	.	78	N	110	n	142	é	174	Æ	206	Œ	238	Ó
15	<SI>	47	/	79	O	111	o	143	è	175	Ø	207	œ	239	Ô
16	<DLE>	48	0	80	P	112	p	144	ê	176	∞	208	-	240	🍏
17	<DC1>	49	1	81	Q	113	q	145	ë	177	±	209	—	241	Ò
18	<DC2>	50	2	82	R	114	r	146	í	178	≤	210	"	242	Ú
19	<DC3>	51	3	83	S	115	s	147	ì	179	≥	211	"	243	Û
20	<DC4>	52	4	84	T	116	t	148	î	180	¥	212	`	244	Ü
21	<NAK>	53	5	85	U	117	u	149	ï	181	μ	213	'	245	ı
22	<SYN>	54	6	86	V	118	v	150	ñ	182	ð	214	÷	246	^
23	<ETB>	55	7	87	W	119	w	151	ó	183	Σ	215	◇	247	~
24	<CAN>	56	8	88	X	120	x	152	ò	184	Π	216	ÿ	248	—
25	<EM>	57	9	89	Y	121	y	153	ô	185	π	217	Ŷ	249	˘
26	<SUB>	58	:	90	Z	122	z	154	ö	186	∫	218	/	250	˙
27	<ESC>	59	;	91	[	123	{	155	õ	187	ª	219	€	251	°
28	<FS>	60	<	92	\	124		156	ú	188	º	220	<	252	¸
29	<GS>	61	=	93	]	125	}	157	ù	189	Ω	221	>	253	”
30	<RS>	62	>	94	^	126	~	158	û	190	æ	222	fi	254	ˆ
31	<US>	63	?	95	_	127	<DEL>	159	ü	191	ø	223	fl	255	ˇ

# Kesirli Sayılar(Float) Veri Tipi

- Pozitif ya da negatif bütün kesirli sayıları tutabileceğimiz veri tipidir.
- Örnek : 1.5, 2.0, -0.3666
- printf ve scanf komutlarında kesirli sayılar için %f kullanılmalıdır.

# Örnek - 1

```
#include<stdio.h>
int main( void )
{
    float bolunen = 12.0, bolen = 8.0;
    float bolum;
    bolum = bolunen / bolen;
    printf("Sonuc: %f\n",bolum);
    return 0;
}
```



%f float veri tipi için kullanılır!

# Biçimlendirilmiş Çıktılar

**Table 3.6** Effect of Field Width Specifiers

Specifier	Number	Display	Comments
%2d	3	^3	Number fits in field
%2d	43	43	Number fits in field
%2d	143	143	Field width ignored
%2d	2.3	Compiler dependent	Floating-point number in an integer field
%5.2f	2.366	^2.37	Field of 5 with 2 decimal digits
%5.2f	42.3	42.30	Number fits in field
%5.2f	142.364	142.36	Field width ignored but fractional specifier is used
%5.2f	142	Compiler dependent	Integer in a floating-point field

## Örnek - 2

```
#include<stdio.h>
int main()
{
    printf("Tam sayi gosterimi: %d", 56);
    printf("\nIki sayinin toplami: %d", 56 + 10);
    printf("\nIki sayinin carpimi: %d", 56*10);
    printf("\n\nKarakter gosterimi: %c", 'a');
    printf("\nKarakterin ascii kodu: %d", 'a');
    printf("\nKaraktere tam sayi ekleme: %d", 'a' + 1);
    printf("\n\nKesirli sayi gosterimi: %f", 2.2);
    printf("\nIki kesirli sayinin toplami: %f", 2.2 + 2.6);
    printf("\n\nKesirli sayinin bicimlendirilmis gosterimi: %2.1f", 2.2);
    printf("\nIki kesirli sayinin toplamının bicimlendirilmis hali: %4.3f", 2.2 + 2.6);
    return 0;
}
```

### Programın Çıktısı:

Tam sayi gosterimi: 56  
Iki sayinin toplami: 66  
Iki sayinin carpimi: 560

Karakter gosterimi: a  
Karakterin ascii kodu: 97  
Karaktere tam sayi ekleme: 98

Kesirli sayi gosterimi: 2.200000  
Iki kesirli sayinin toplam²: 4.800000

Kesirli sayinin bicimlendirilmis gosterimi: 2.2  
Iki kesirli sayinin toplamının bicimlendirilmis hali: 4.800



**Table 2.5** Escape Sequences

Escape Sequence	Character Represented	Meaning	ASCII Code
<code>\n</code>	Newline	Move to a new line	00001010
<code>\t</code>	Horizontal tab	Move to next horizontal tab setting	00001001
<code>\v</code>	Vertical tab	Move to next vertical tab setting	00001011
<code>\b</code>	Backspace	Move back one space	00001000
<code>\r</code>	Carriage return	Carriage return (moves the cursor to the start of the current line—used for overprinting)	00001101
<code>\f</code>	Form feed	Issue a form feed	00001100
<code>\a</code>	Alert	Issue an alert (usually a bell sound)	00000111
<code>\\</code>	Backslash	Insert a backslash character (places an actual backslash character within a string)	01011100
<code>\?</code>	Question mark	Insert a question mark character	00111111
<code>\'</code>	Single quotation	Insert a single quote character (places an inner single quote within a set of outer single quotes)	00100111
<code>\"</code>	Double quotation mark	Insert a double quote character (places an inner double quote within a set of outer double quotes)	00100010
<code>\nnn</code>	Octal number	The number <i>nnn</i> ( <i>n</i> is a digit) is to be considered an octal number	—
<code>\xhhhh</code>	Hexadecimal number	The number <i>hhhh</i> ( <i>h</i> is a digit) is to be considered a hexadecimal number	—
<code>\0</code>	Null character	Insert the null character, which is defined as having the value 0	00000000

# Örnek - 1(Artırma operatörleri)

```
#include<stdio.h>
int main()
{
    int sayac;
    sayac = 0;
    printf("Sayacin baslangic degeri: %d", sayac);
    sayac = sayac + 1;
    printf("\n sayacin simdiki degeri: %d", sayac);
    sayac++;
    printf("\n sayacin simdiki degeri: %d", sayac);
    sayac +=1;
    printf("\n sayacin simdiki degeri: %d", sayac);
    return 0;
}
```

## Örnek - 2(Azaltma operatörleri)

```
#include<stdio.h>
int main()
{
    int sayac;
    sayac = 3;
    printf("Sayacin baslangic degeri: %d", sayac);
    sayac = sayac - 1;
    printf("\n sayacin simdiki degeri: %d", sayac);
    sayac--;
    printf("\n sayacin simdiki degeri: %d", sayac);
    sayac -= 1;
    printf("\n sayacin simdiki degeri: %d", sayac);
    return 0;
}
```

# Kullanıcıdan alınan değerleri değişkenlere depolamak

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int tamsayi;
```

```
    float kesirliisayi;
```

```
    printf("Lutfen tam sayi giriniz: ");
```

```
    scanf("%d",&tamsayi);
```

```
    printf("Lutfen kesirli sayi giriniz: ");
```

```
    scanf("%f",&kesirliisayi);
```

```
    printf("Girilen tam sayi degeri: %d, kesirli deger: %f\n", tamsayi, kesirliisayi);
```

```
    return 0;
```

```
}
```



Adres sembolü

# Sabit Değerler

```
#include <stdio.h>
```

```
#define PI 3.1416
```

```
int main()
```

```
{
```

```
    float cap, cevre;
```

```
    cap = 2.0;
```

```
    cevre = 2.0 * PI * cap;
```

```
    printf("Cemberin cevresi: %f\n", cevre);
```

```
    return 0;
```

```
}
```



Sabit değer tanımı

# Alıştırmalar

1. Yüksekliği ve tabanı verilen üçgenin alanını bulan ve ekrana yazdıran bir C programı yazınız.
2. Yazdığınız C programını, yüksekliği ve tabanı kullanıcıdan alacak şekilde tekrar düzenleyiniz.
3. Aşağıdaki formülleri kullanarak hacmi ve alanı hesaplayan C programı yazınız. Yazdığınız programı şu değerler ile test ediniz: 1, 6, 12.2, 0,2. ( Float veri tipi )
  - a.  $V = \frac{4}{3} \pi r^3$
  - b.  $A = 4 \pi r^2$
4. Yazdığınız programda PI değerini sabit bir değer olarak tanımlayınız. ( #define )