PERTEMUAN 2

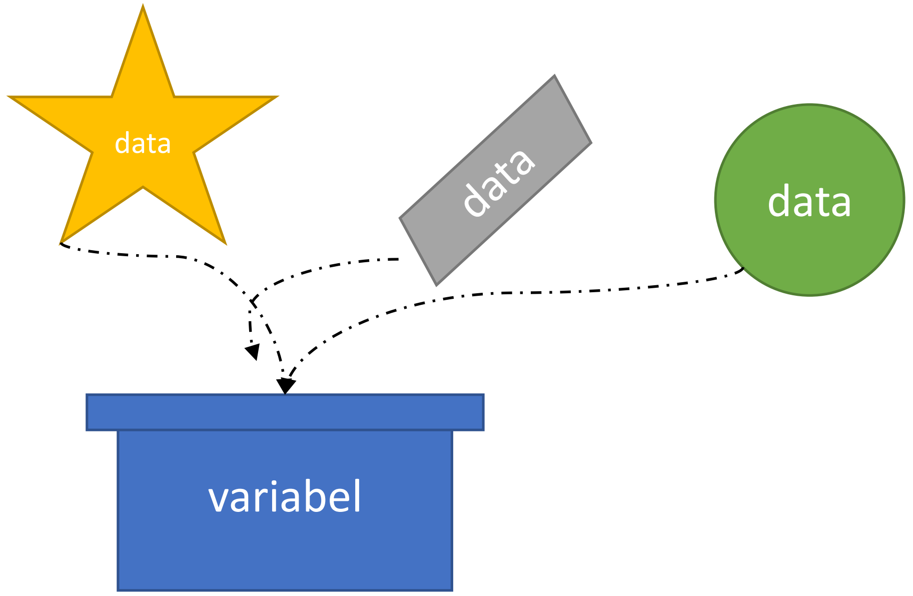
TODAY

- ❑ TIPE DATA
- ❑ VARIABEL
- ❑ OPERATOR

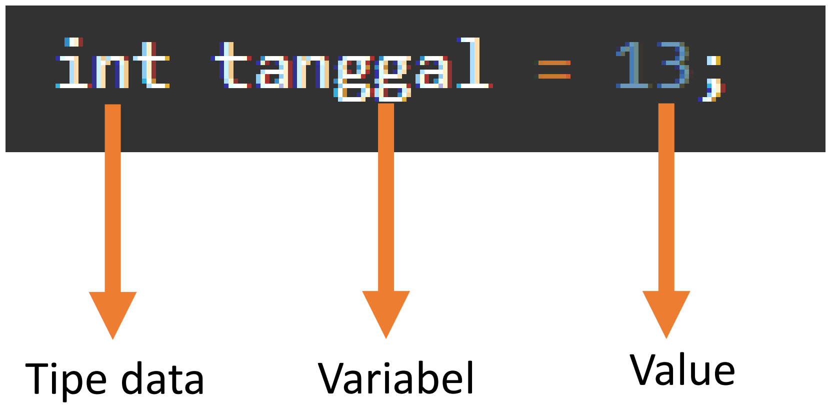
VARIABEL

VARIABEL

Variabel murapakan sebuah tempat atau wadah untuk memuat sebuah nilai.



PENULISAN



```
int tanggal = 13;
```

PENULISAN

- Nama variabel tidak boleh didahului dengan **simbol** dan **angka**.
- Nama variabel tidak boleh menggunakan **kata kunci** yang sudah **ada pada bahasa C**, contoh: if, int, void, dll.
- Nama variabel bersifat **case sensitive**, artianya huruf besar dan kecil dibedakan, contoh: **tanggal** dan **Tanggal** adalah dua variabel yang berbeda.
- Disarankan menggunakan **underscore/SentenceCase** untuk nama variabel yang terdiri dari dua suku kata, contoh: **nama_mahasiswa** atau **namaMahasiswa**.

TIPE DATA

TIPE DATA

- TIPE DATA DASAR
- TIPE DATA TURUNAN
- TIPE DATA VOID
- TIPE DATA BENTUKAN (ENUM)

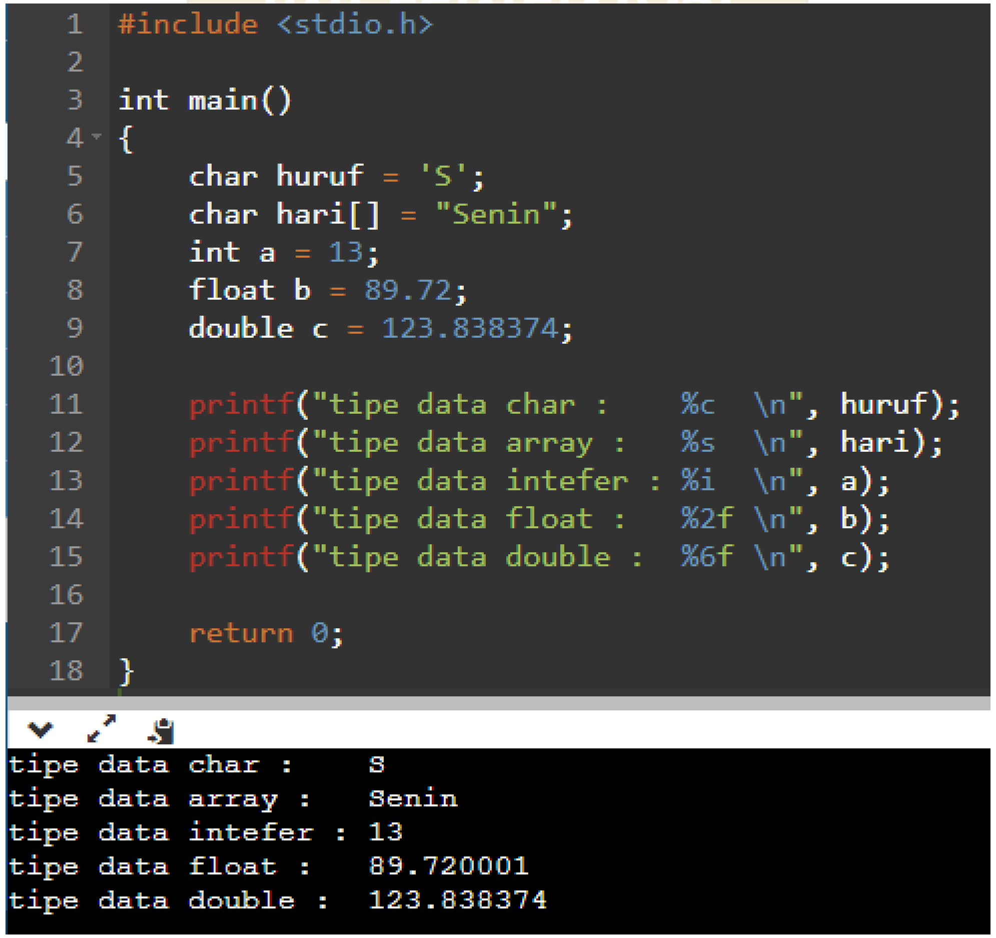
TIPE DATA DASAR

Tipe Data	Ukuran	Panjang	Contoh
char	1 byte	-128 — 127	'A', 'a', '3'
int	2 byte	-2,147,483,648 — 2,147,483,647	32, 1, 4
float	4 byte	1.2E-38 — 3.4E+38	4.3, 2.2, 6.0
double	8 byte	2.3E-308 — 1.7E+308	4.2, 4.22, 3.2

Syntax

Tipe Data	Syntax deklarasi	Syntax output
char	char = ‘value’;	%c
integer	int = value;	%i
float	float = value;	%2f
double	double = value;	%2f

IMPLEMENTASI



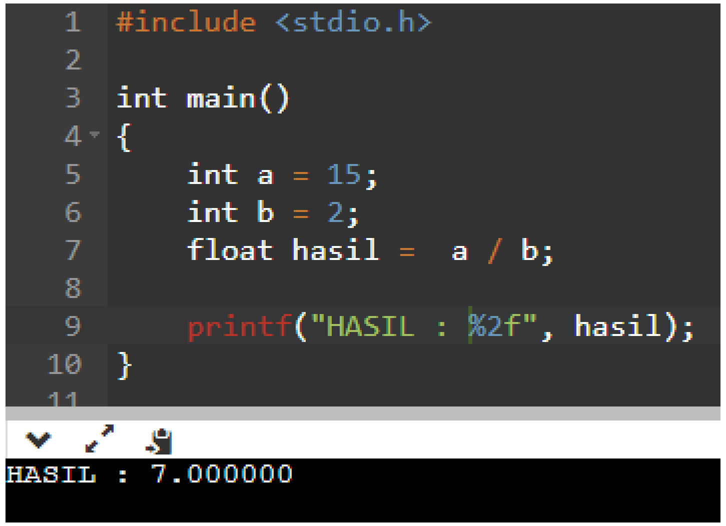
```c
#include <stdio.h>

int main()
{
    char huruf = 'S';
    char hari[] = "Senin";
    int a = 13;
    float b = 89.72;
    double c = 123.838374;

    printf("tipe data char :    %c  \n", huruf);
    printf("tipe data array :   %s  \n", hari);
    printf("tipe data intefer : %i  \n", a);
    printf("tipe data float :   %2f \n", b);
    printf("tipe data double :  %6f \n", c);

    return 0;
}
```

```
tipe data char :    S
tipe data array :   Senin
tipe data intefer : 13
tipe data float :   89.720001
tipe data double :  123.838374
```

KONVERSI TIPE DATA

KONVERSI

- Konversi tipe data adalah merubah jenis tipe data tersebut untuk diolah atau diproses.
- Contoh :



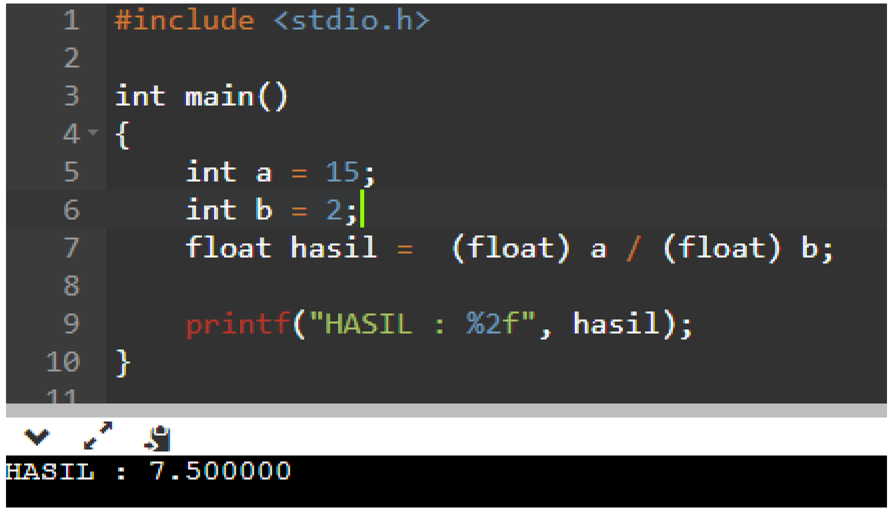
```
#include <stdio.h>

int main()
{
    int a = 15;
    int b = 2;
    float hasil =  a / b;

    printf("HASIL : %2f", hasil);
}
```

```
HASIL : 7.000000
```

- Pada kasus ini dimana hasil dari 15 / 2 harusnya adalah 7,5
- Namun dikarenakan tipe data integer tidak dapat menyimpan bilangan riil, maka valuenya dibulatkan

KONVERSI

- ❑ Konversi dilakukan dari tipe data integer menjdi float
- ❑ Maka akan keluar hasil berupa bilangan riil yang diinginkan



```c
#include <stdio.h>

int main()
{
    int a = 15;
    int b = 2;
    float hasil =  (float) a / (float) b;

    printf("HASIL : %2f", hasil);
}
```

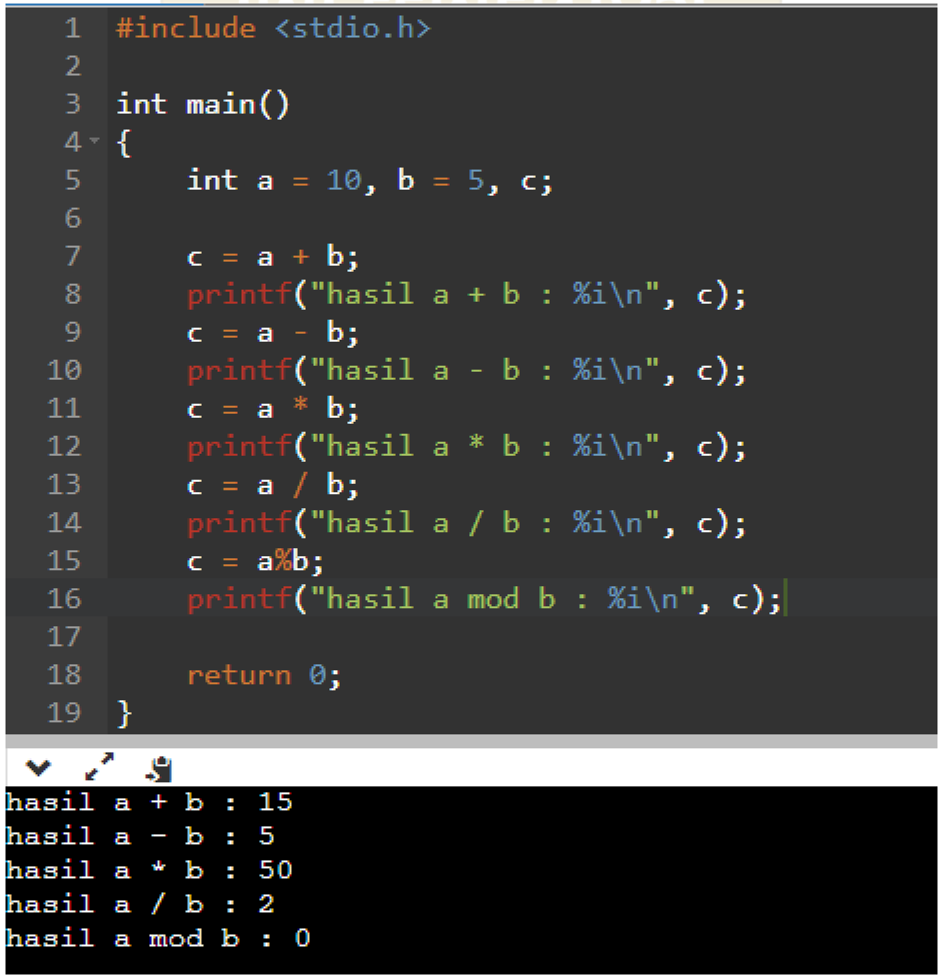
```
HASIL : 7.500000
```

OPERATOR

OPERATOR

Operator	Meaning of Operator
+	addition or unary plus
-	subtraction or unary minus
*	multiplication
/	division
%	remainder after division (modulo division)

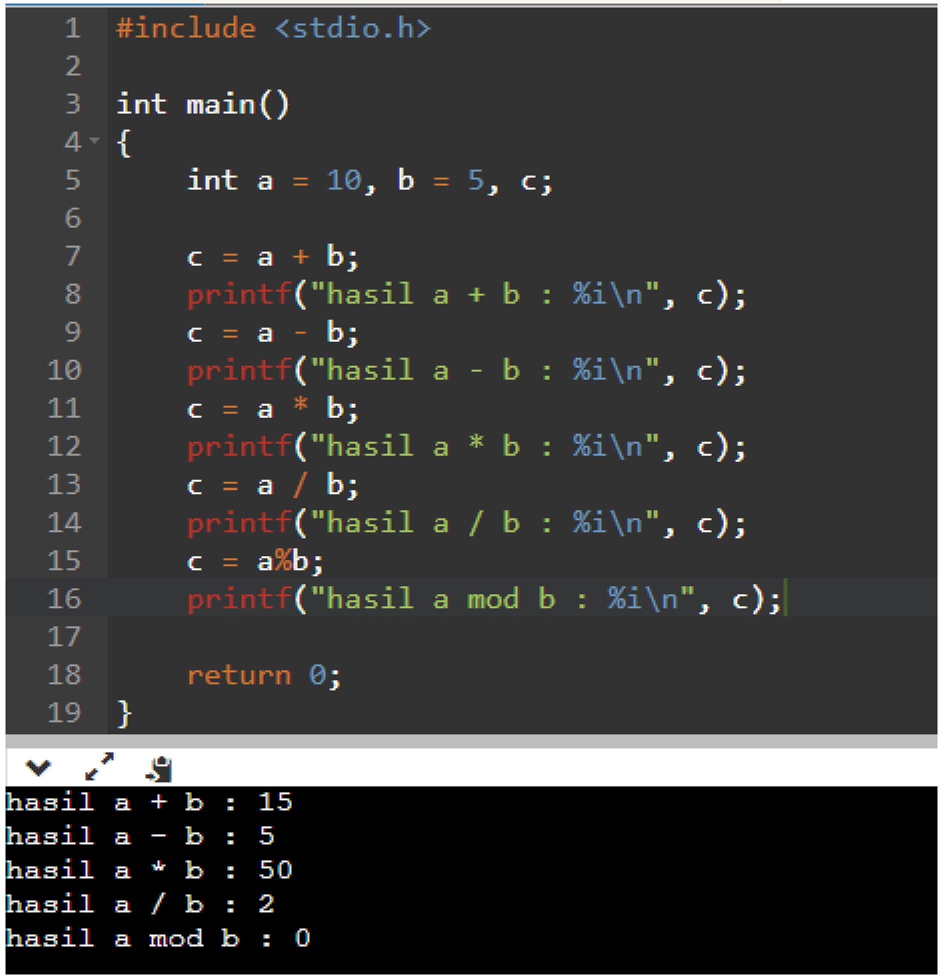
IMPLEMENTASI



```c
#include <stdio.h>

int main()
{
    int a = 10, b = 5, c;

    c = a + b;
    printf("hasil a + b : %i\n", c);
    c = a - b;
    printf("hasil a - b : %i\n", c);
    c = a * b;
    printf("hasil a * b : %i\n", c);
    c = a / b;
    printf("hasil a / b : %i\n", c);
    c = a%b;
    printf("hasil a mod b : %i\n", c);

    return 0;
}
```

```
hasil a + b : 15
hasil a - b : 5
hasil a * b : 50
hasil a / b : 2
hasil a mod b : 0
```

INCREMENT DAN DECREMENT

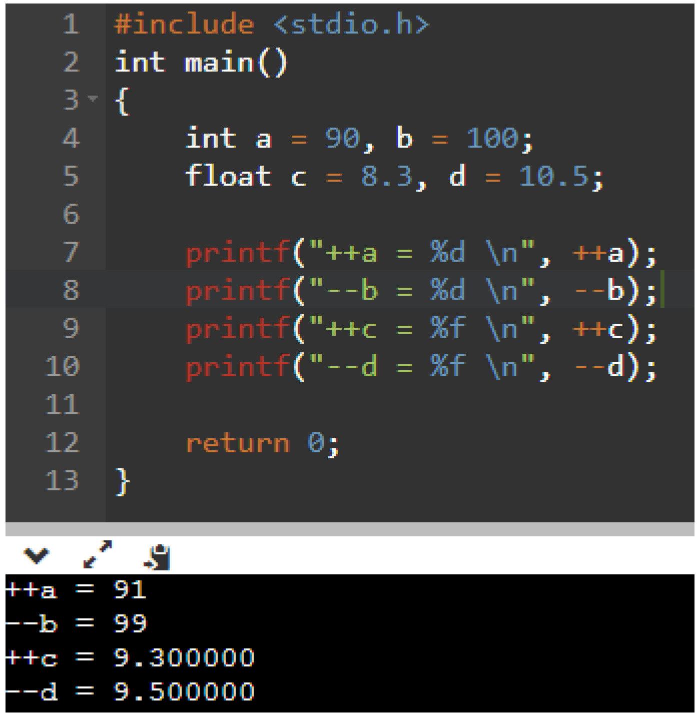


```c
#include <stdio.h>

int main()
{
    int a = 10, b = 5, c;

    c = a + b;
    printf("hasil a + b : %i\n", c);
    c = a - b;
    printf("hasil a - b : %i\n", c);
    c = a * b;
    printf("hasil a * b : %i\n", c);
    c = a / b;
    printf("hasil a / b : %i\n", c);
    c = a%b;
    printf("hasil a mod b : %i\n", c);

    return 0;
}
```

```
hasil a + b : 15
hasil a - b : 5
hasil a * b : 50
hasil a / b : 2
hasil a mod b : 0
```

INCREMENT DAN DECREMENT

Syntax	Meaning of Operator
++	Increment
--	Decrement



```
#include <stdio.h>
int main()
{
    int a = 90, b = 100;
    float c = 8.3, d = 10.5;

    printf("++a = %d \n", ++a);
    printf("--b = %d \n", --b);
    printf("++c = %f \n", ++c);
    printf("--d = %f \n", --d);

    return 0;
}
```

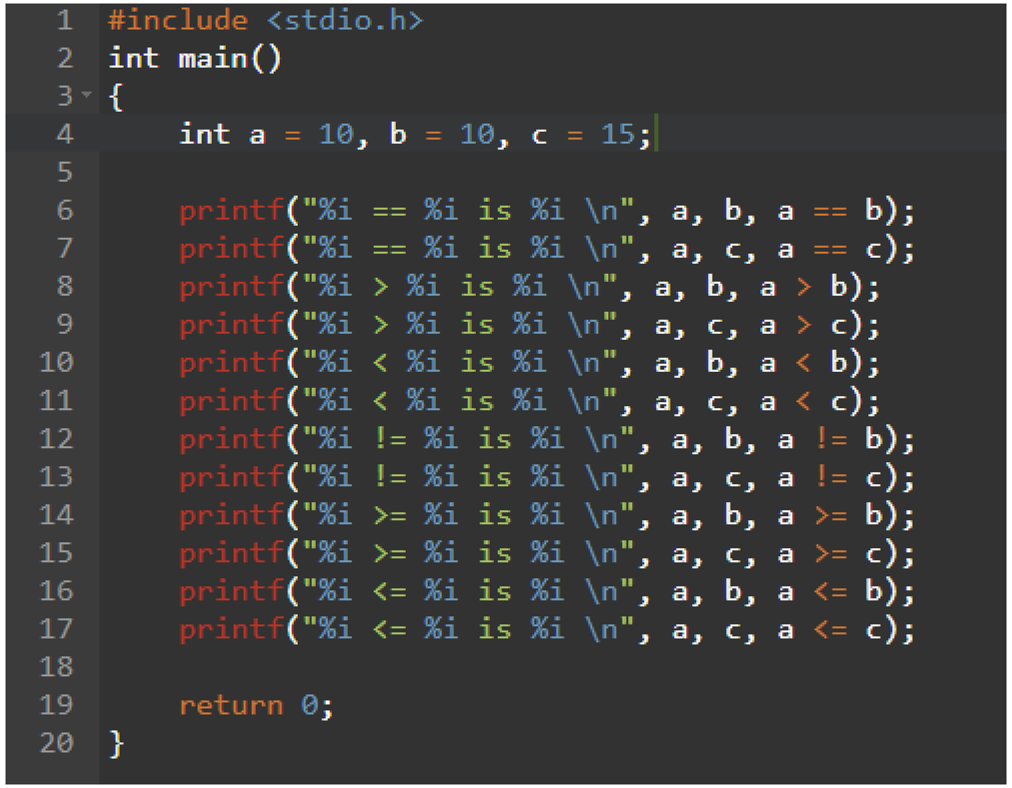
```
++a = 91
--b = 99
++c = 9.300000
--d = 9.500000
```

RELATIONAL OPERATOR

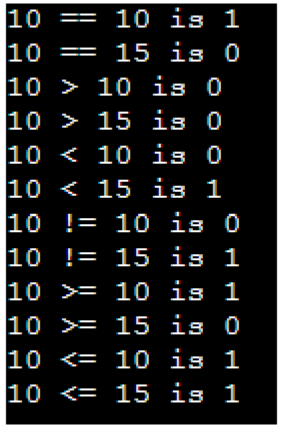
Operator	Meaning of Operator	Example
==	Equal to	5 == 3 is evaluated to 0
>	Greater than	5 > 3 is evaluated to 1
<	Less than	5 < 3 is evaluated to 0
!=	Not equal to	5 != 3 is evaluated to 1
>=	Greater than or equal to	5 >= 3 is evaluated to 1
<=	Less than or equal to	5 <= 3 is evaluated to 0

INFORMATIKA
ITTelkom Surabaya

EXAMPLE



```c
#include <stdio.h>
int main()
{
    int a = 10, b = 10, c = 15;

    printf("%i == %i is %i \n", a, b, a == b);
    printf("%i == %i is %i \n", a, c, a == c);
    printf("%i > %i is %i \n", a, b, a > b);
    printf("%i > %i is %i \n", a, c, a > c);
    printf("%i < %i is %i \n", a, b, a < b);
    printf("%i < %i is %i \n", a, c, a < c);
    printf("%i != %i is %i \n", a, b, a != b);
    printf("%i != %i is %i \n", a, c, a != c);
    printf("%i >= %i is %i \n", a, b, a >= b);
    printf("%i >= %i is %i \n", a, c, a >= c);
    printf("%i <= %i is %i \n", a, b, a <= b);
    printf("%i <= %i is %i \n", a, c, a <= c);

    return 0;
}
```

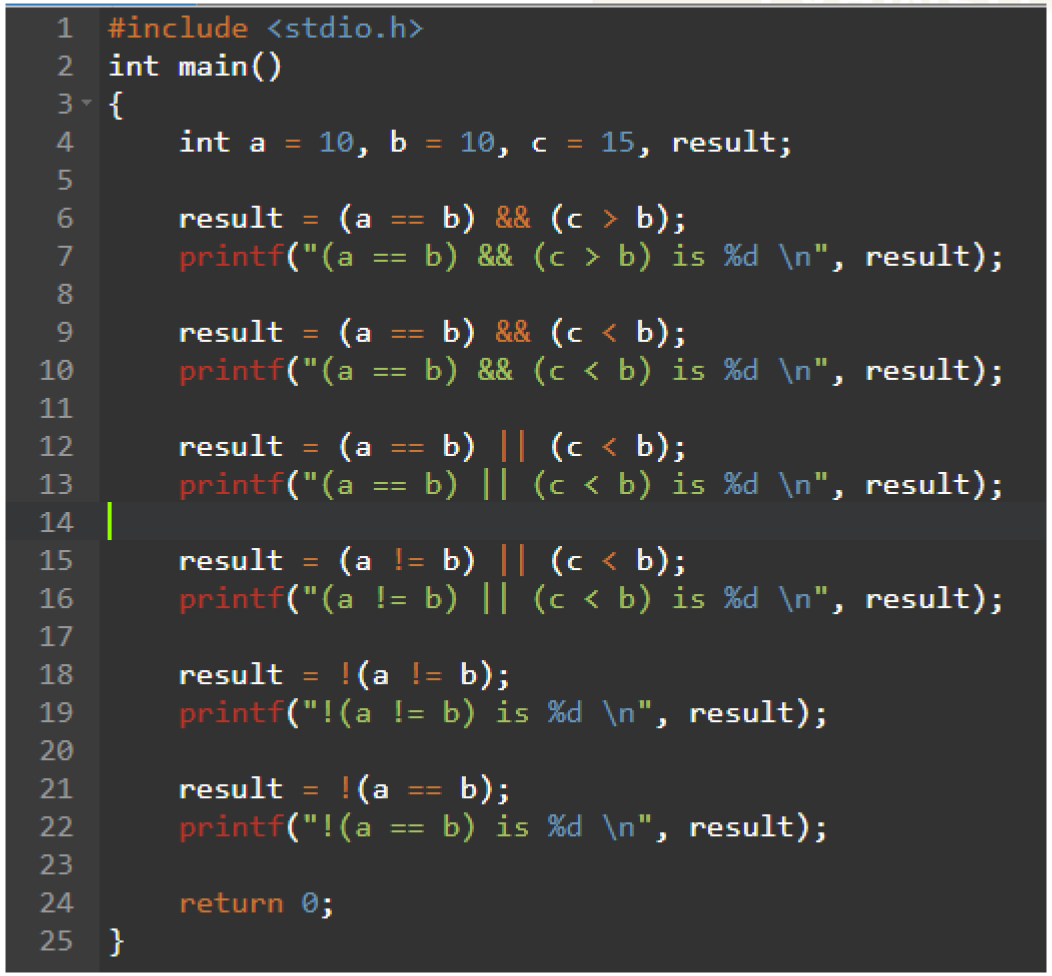


```
10 == 10 is 1
10 == 15 is 0
10 > 10 is 0
10 > 15 is 0
10 < 10 is 0
10 < 15 is 1
10 != 10 is 0
10 != 15 is 1
10 >= 10 is 1
10 >= 15 is 0
10 <= 10 is 1
10 <= 15 is 1
```

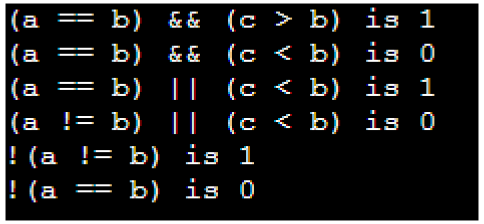
LOGICAL OPERATOR

Operator	Meaning	Example
&&	Logical AND. True only if all operands are true	If c = 5 and d = 2 then, expression ((c==5) && (d>5)) equals to 0.
||	Logical OR. True only if either one operand is true	If c = 5 and d = 2 then, expression ((c==5) || (d>5)) equals to 1.
!	Logical NOT. True only if the operand is 0	If c = 5 then, expression !(c==5) equals to 0.

EXAMPLE



```c
#include <stdio.h>
int main()
{
    int a = 10, b = 10, c = 15, result;

    result = (a == b) && (c > b);
    printf("(a == b) && (c > b) is %d \n", result);

    result = (a == b) && (c < b);
    printf("(a == b) && (c < b) is %d \n", result);

    result = (a == b) || (c < b);
    printf("(a == b) || (c < b) is %d \n", result);

    result = (a != b) || (c < b);
    printf("(a != b) || (c < b) is %d \n", result);

    result = !(a != b);
    printf("!(a != b) is %d \n", result);

    result = !(a == b);
    printf("!(a == b) is %d \n", result);

    return 0;
}
```



```
(a == b) && (c > b) is 1
(a == b) && (c < b) is 0
(a == b) || (c < b) is 1
(a != b) || (c < b) is 0
!(a != b) is 1
!(a == b) is 0
```

HAPPY CODING