Back to the reefs of the Sea of Cs

Operating with Expressions

Ceacon 1 - Last Episode

CS-101 2021

Lec-10

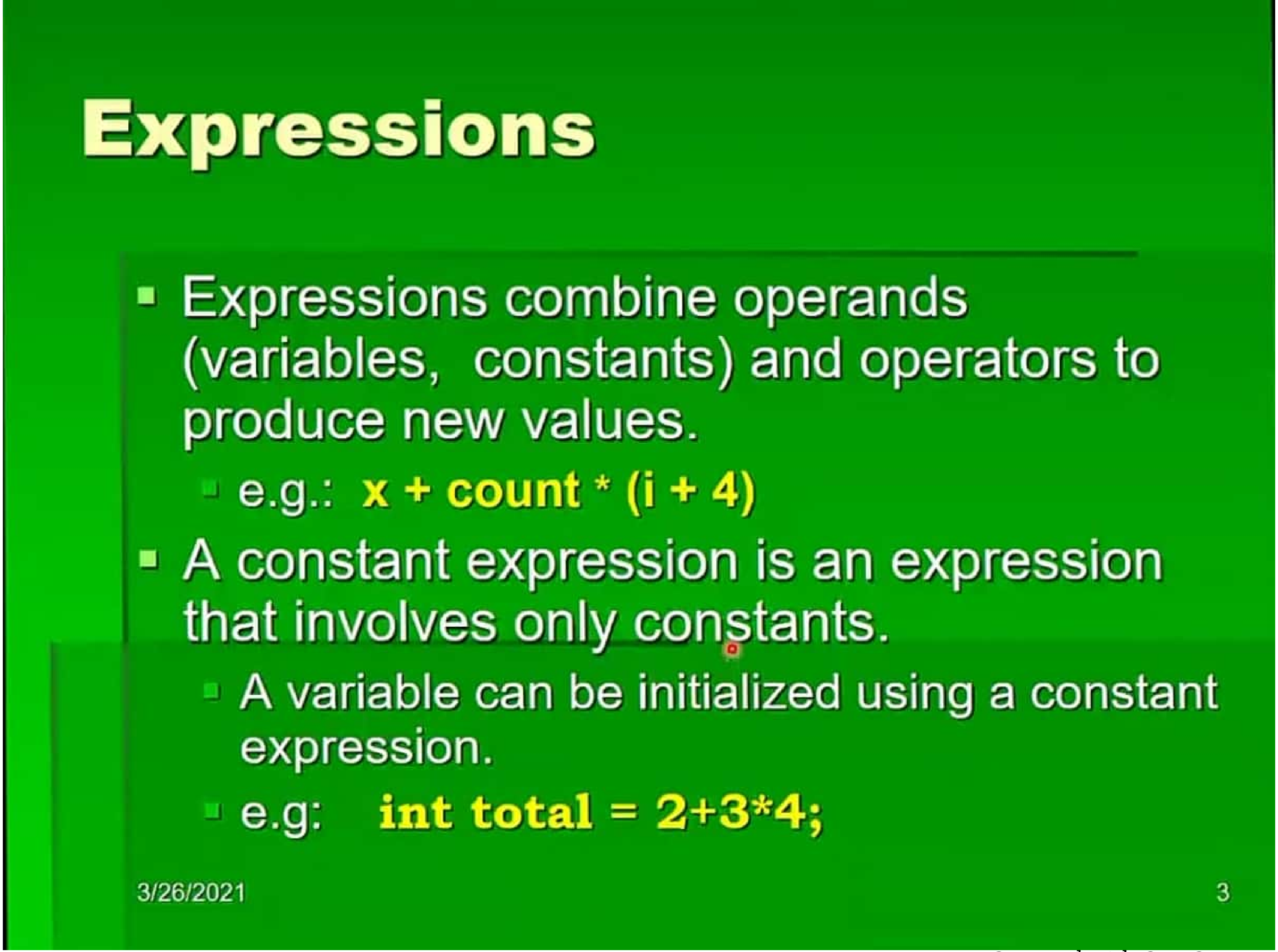
We are now back in the reefs of the Sea of Cs infested by a variety of predators

DANGER lurks everywhere!

Pay close attention to the Instructor.

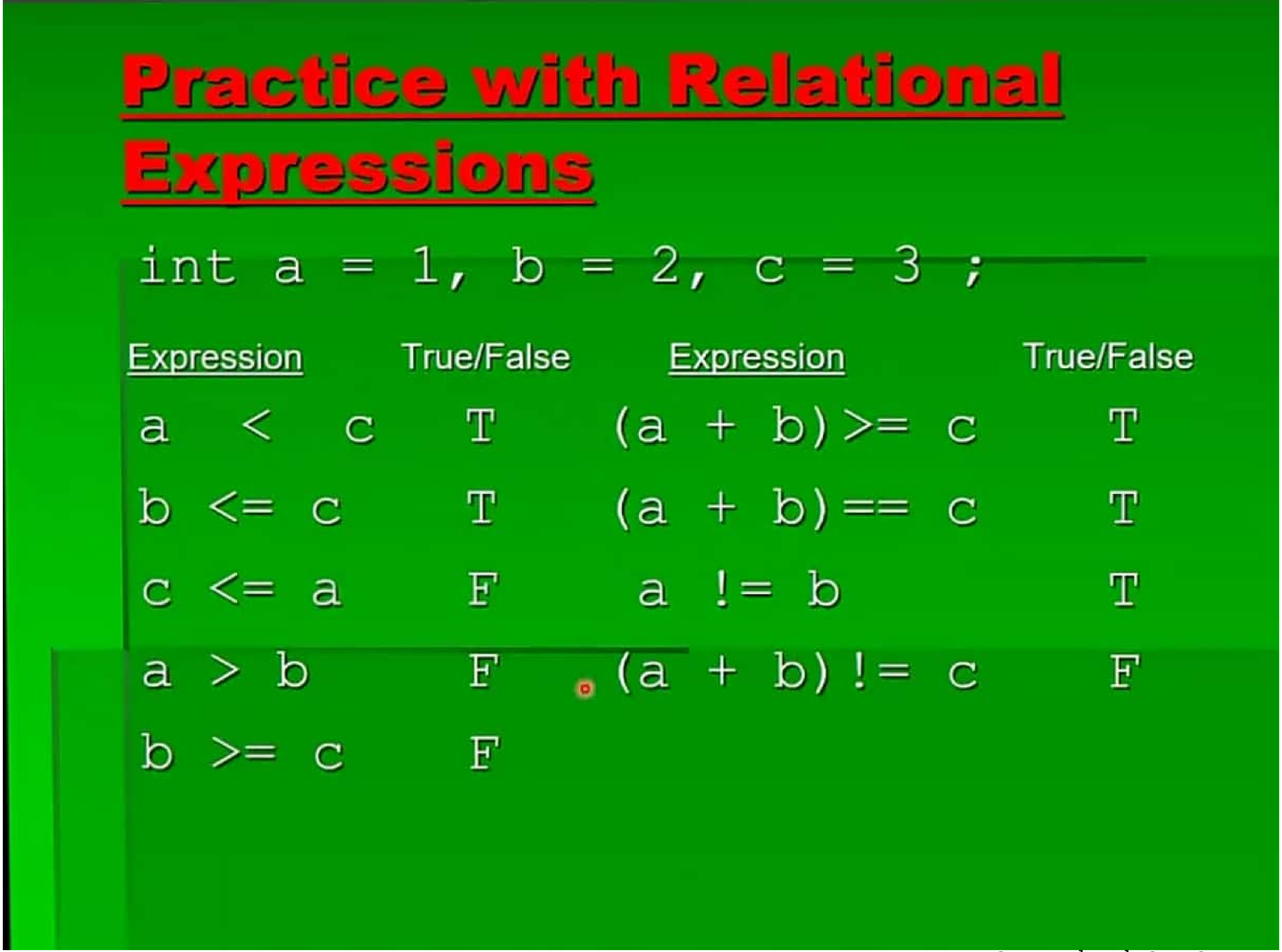
Don't forget to save him, if you detect danger. Your life in the reefs depends on him.

3/26/2021



Expressions

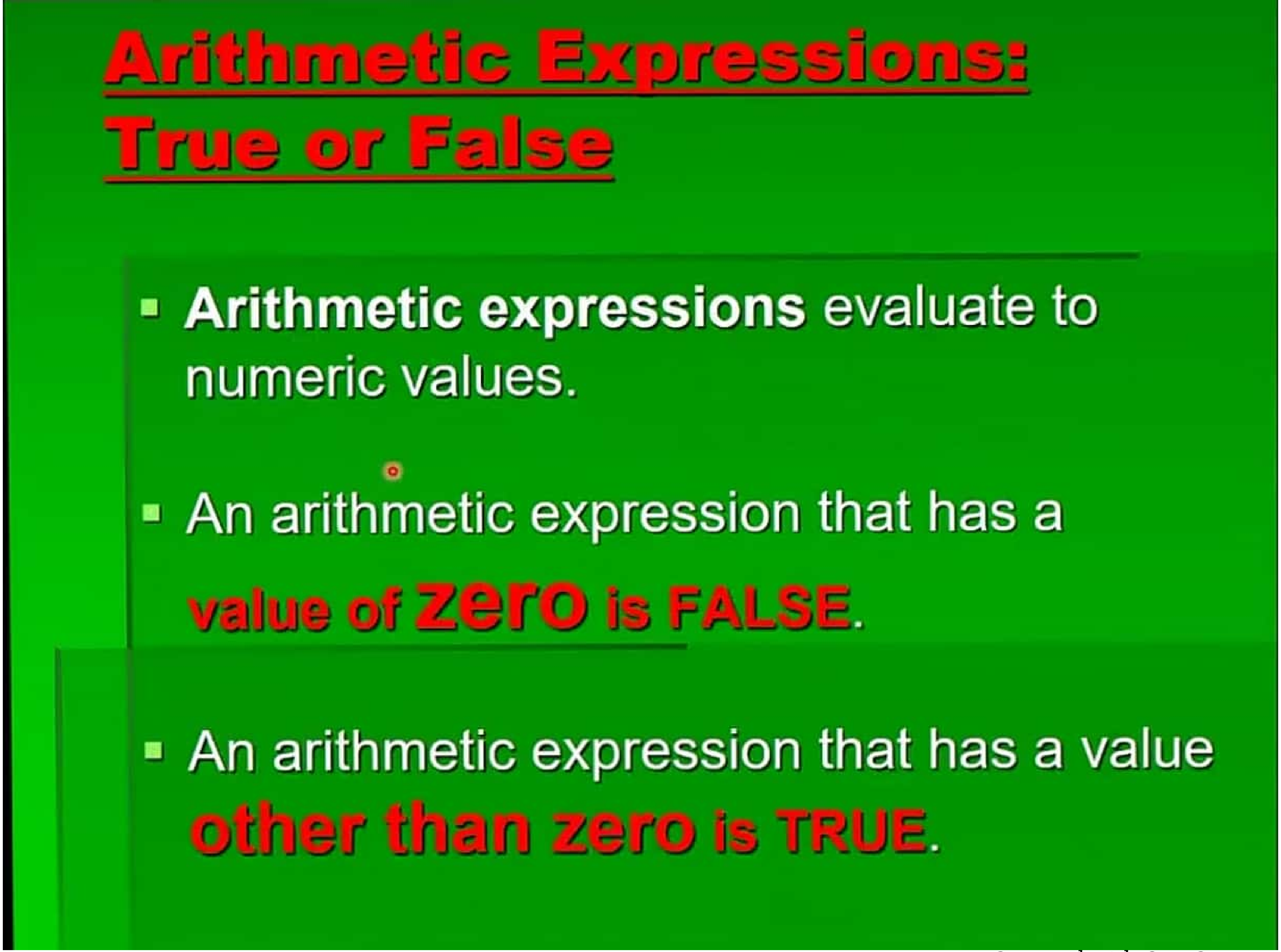
- Expressions combine operands (variables, constants) and operators to produce new values.
  - e.g.: **x + count * (i + 4)**
- A constant expression is an expression that involves only constants.
  - A variable can be initialized using a constant expression.
  - e.g: **int total = 2+3*4;**



Practice with Relational Expressions

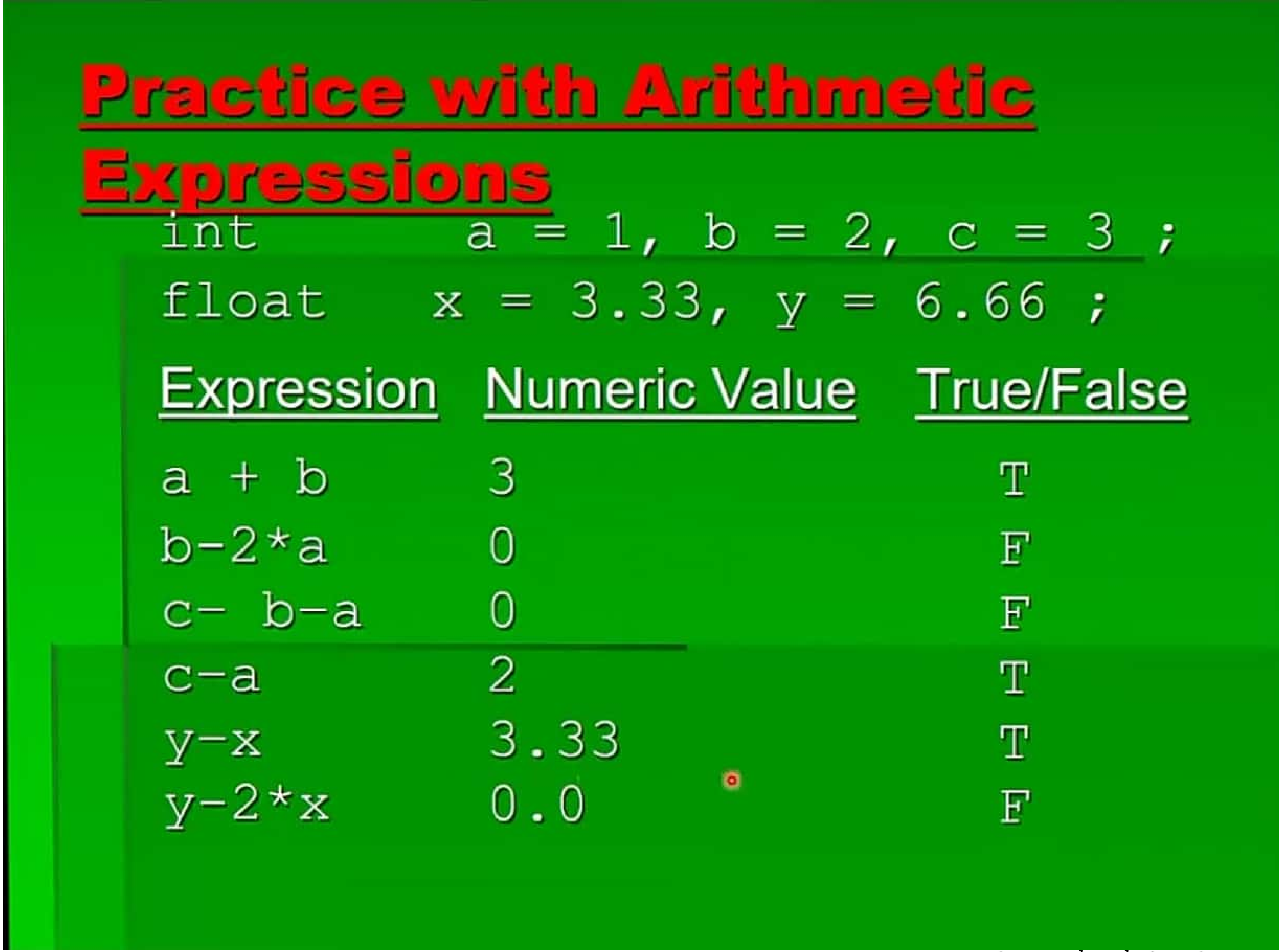
```
int a = 1, b = 2, c = 3 ;
```

Expression	True/False	Expression	True/False
`a < c`	T	`(a + b) >= c`	T
`b <= c`	T	`(a + b) == c`	T
`c <= a`	F	`a != b`	T
`a > b`	F	`(a + b) != c`	F
`b >= c`	F		



Arithmetic Expressions: True or False

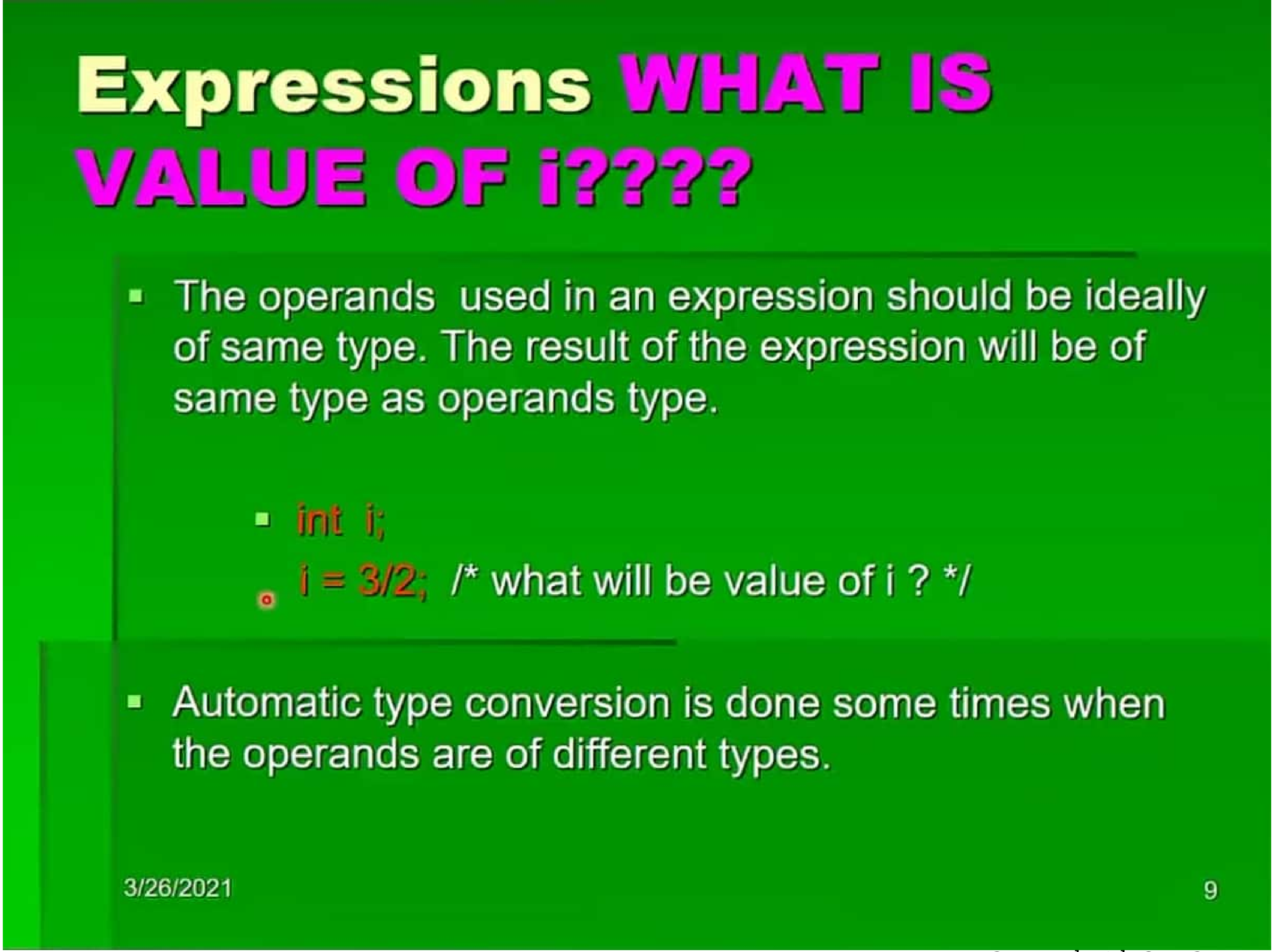
- **Arithmetic expressions** evaluate to numeric values.
- An arithmetic expression that has a **value of zero is FALSE**.
- An arithmetic expression that has a value **other than zero is TRUE**.



Practice with Arithmetic Expressions

```
int       a = 1, b = 2, c = 3 ;
float   x = 3.33, y = 6.66 ;
```

Expression	Numeric Value	True/False
a + b	3	T
b-2*a	0	F
c- b-a	0	F
c-a	2	T
y-x	3.33	T
y-2*x	0.0	F

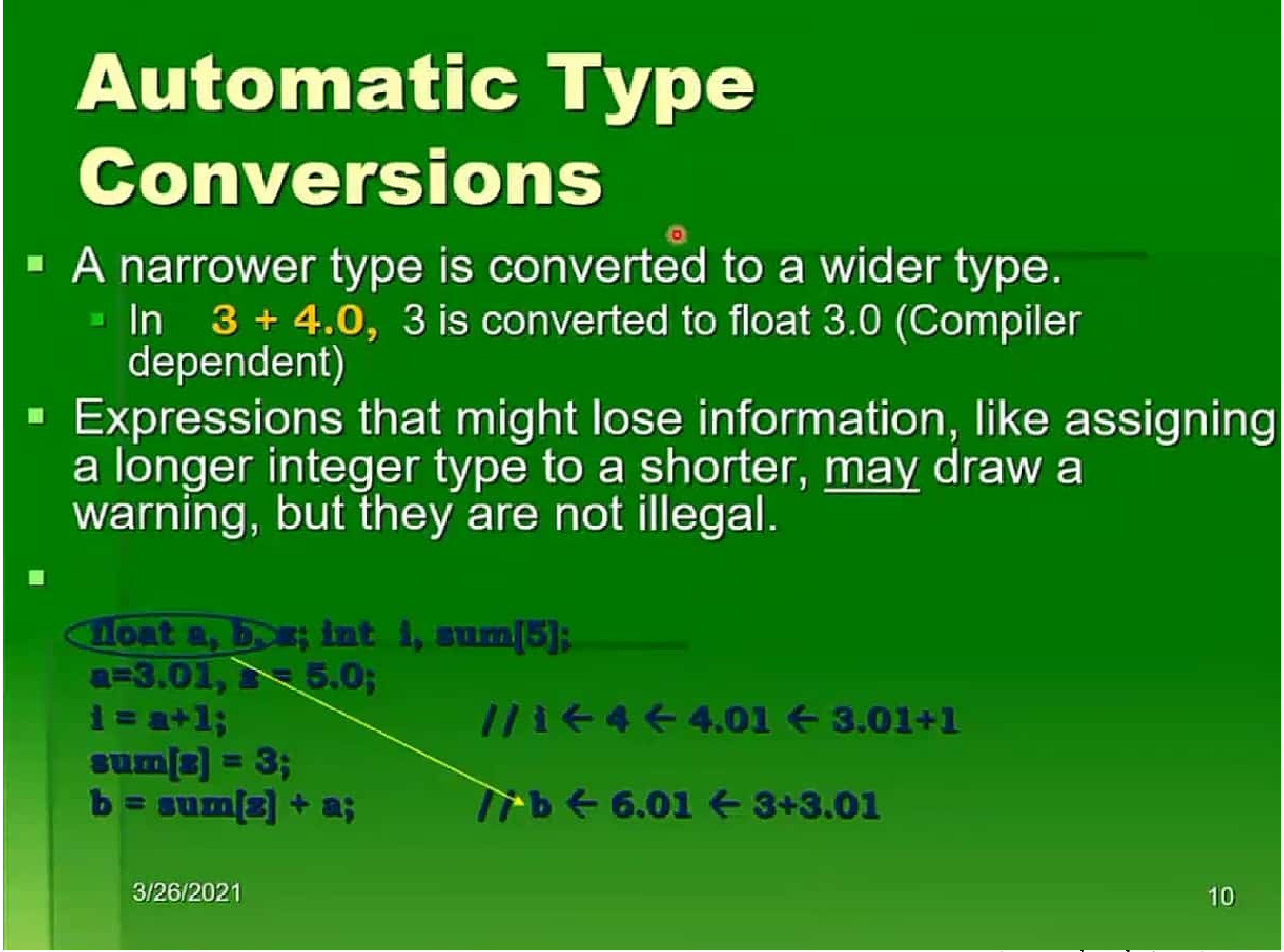


Expressions WHAT IS VALUE OF i????

- The operands used in an expression should be ideally of same type. The result of the expression will be of same type as operands type.

  - int i;
    i = 3/2; /* what will be value of i ? */

- Automatic type conversion is done some times when the operands are of different types.



Automatic Type Conversions

- A narrower type is converted to a wider type.
  - In **3 + 4.0,** 3 is converted to float 3.0 (Compiler dependent)
- Expressions that might lose information, like assigning a longer integer type to a shorter, may draw a warning, but they are not illegal.
-

```
float a, b, z; int  i, sum[5];
a=3.01, z = 5.0;
i = a+1;                    // i ← 4 ← 4.01 ← 3.01+1
sum[z] = 3;
b = sum[z] + a;             // b ← 6.01 ← 3+3.01
```

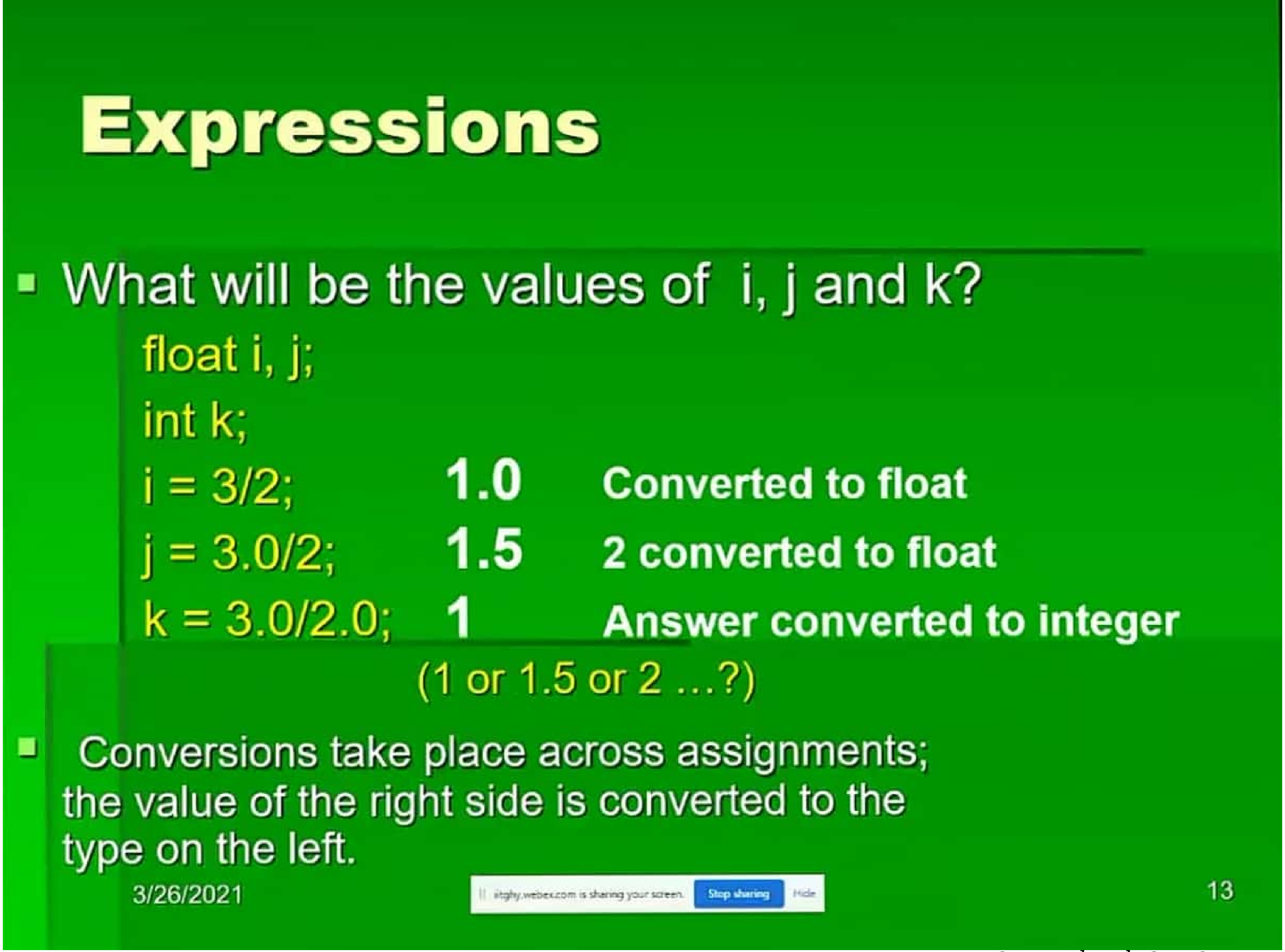
3/26/2021

The Stone fish codones this first error and explains It using an e.g.

In spite of repeated warnings, a careless student (Roll No. 200123024.2, from IITG, name withheld) stepped on me the other day while sCuba diving in the Sea of Cs and entered the *Heaven of Pain*!

```
float a, b,z; int  i, sum[5];
a=3.01, z = 5.1;
i = a+1;                  // i ← 4 ← 4.01 ← 3.01+1
sum[z] = 3;
b = sum[z] + a;           // b ← 6.01 ← 3+3.01
```

The index of an array should be an integer.



Expressions

- What will be the values of i, j and k?

```
float i, j;
int k;
i = 3/2;          1.0     Converted to float
j = 3.0/2;        1.5     2 converted to float
k = 3.0/2.0;   1          Answer converted to integer
                (1 or 1.5 or 2 …?)
```

- Conversions take place across assignments; the value of the right side is converted to the type on the left.

Explicit type conversions (Type casting)

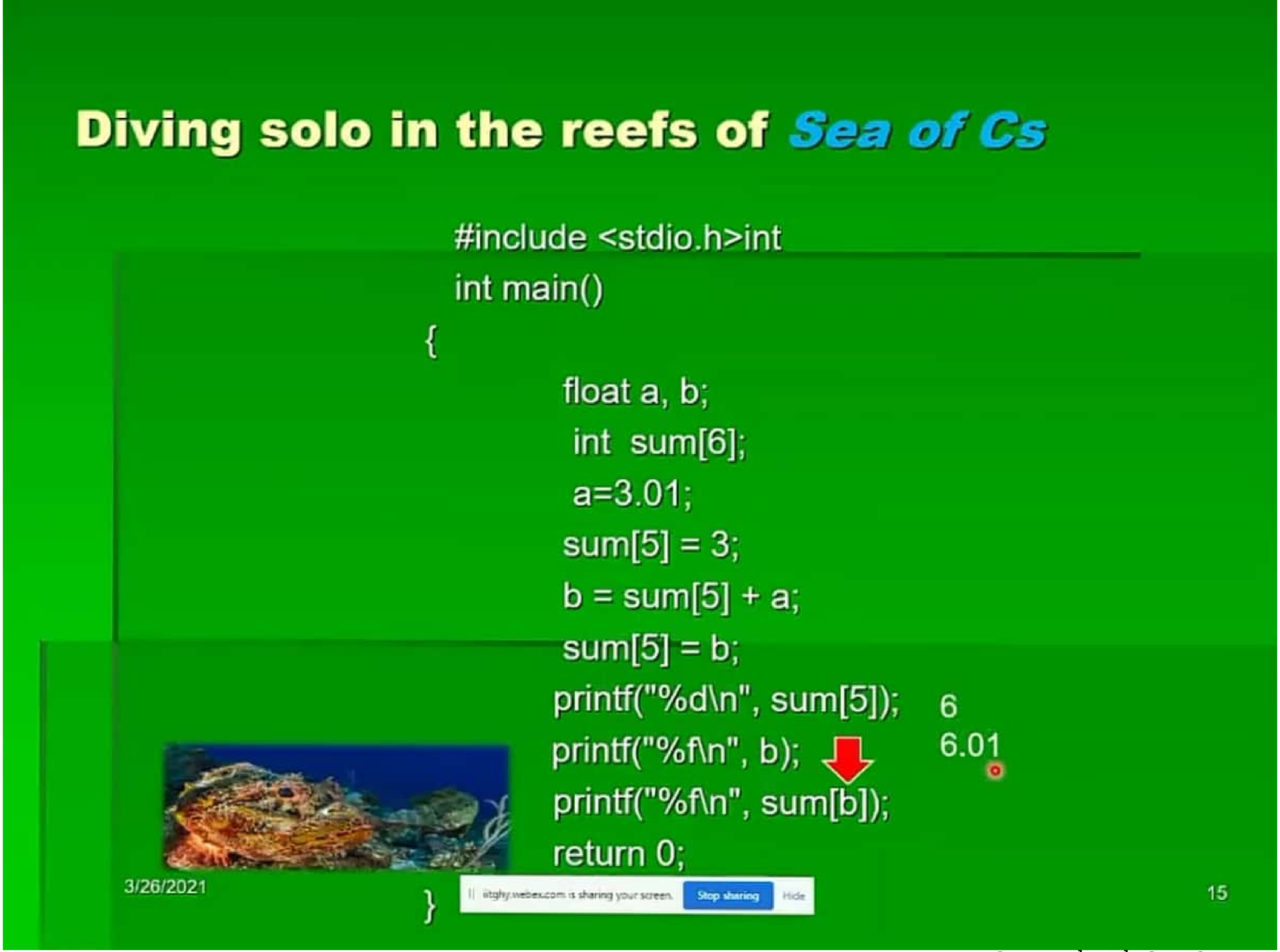
- You can force the type to be converted.
  - (float) 3; /* has value 3.0 */
- Syntax : (type-name) expression
  - float f;
  - f = (float)3/2;
    - /* 3 → 3.0 because of explicit type conversion
    - 3.0/2 → 3.0/2.0 because of automatic conversion
    - So, *f* gets value 1.5 */

- double d = 3.5;

  int i;

  i = (int) d; /* value of *d* itself is not changed */

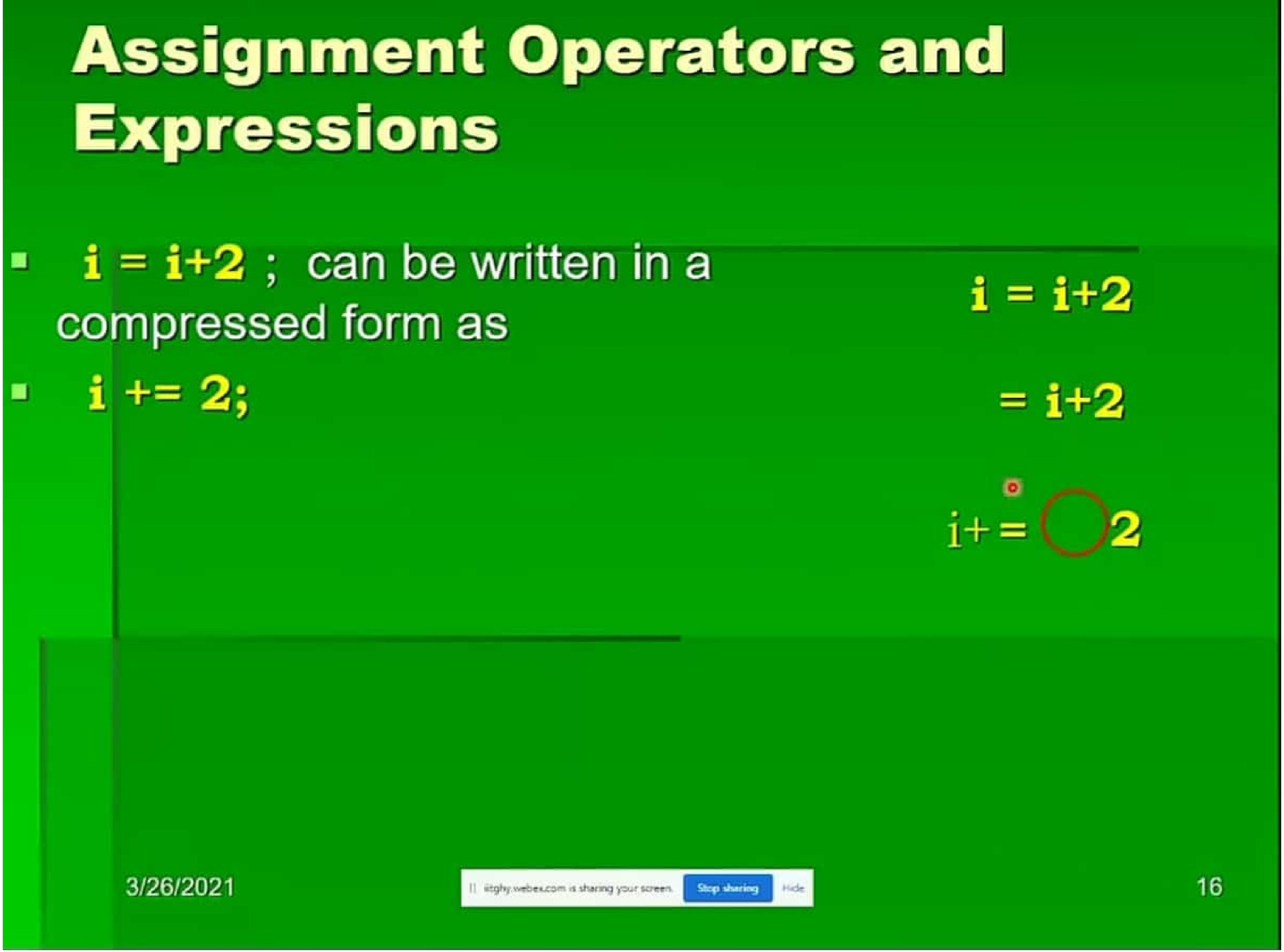
  /* the value of (int) *d* is 3 */



Diving solo in the reefs of *Sea of Cs*

```
#include <stdio.h>int
int main()
{
      float a, b;
       int  sum[6];
       a=3.01;
      sum[5] = 3;
      b = sum[5] + a;
      sum[5] = b;
     printf("%d\n", sum[5]);    6
     printf("%f\n", b);         6.01
     printf("%f\n", sum[b]);
     return 0;
}
```

Scanned with CamScanner



Assignment Operators and Expressions

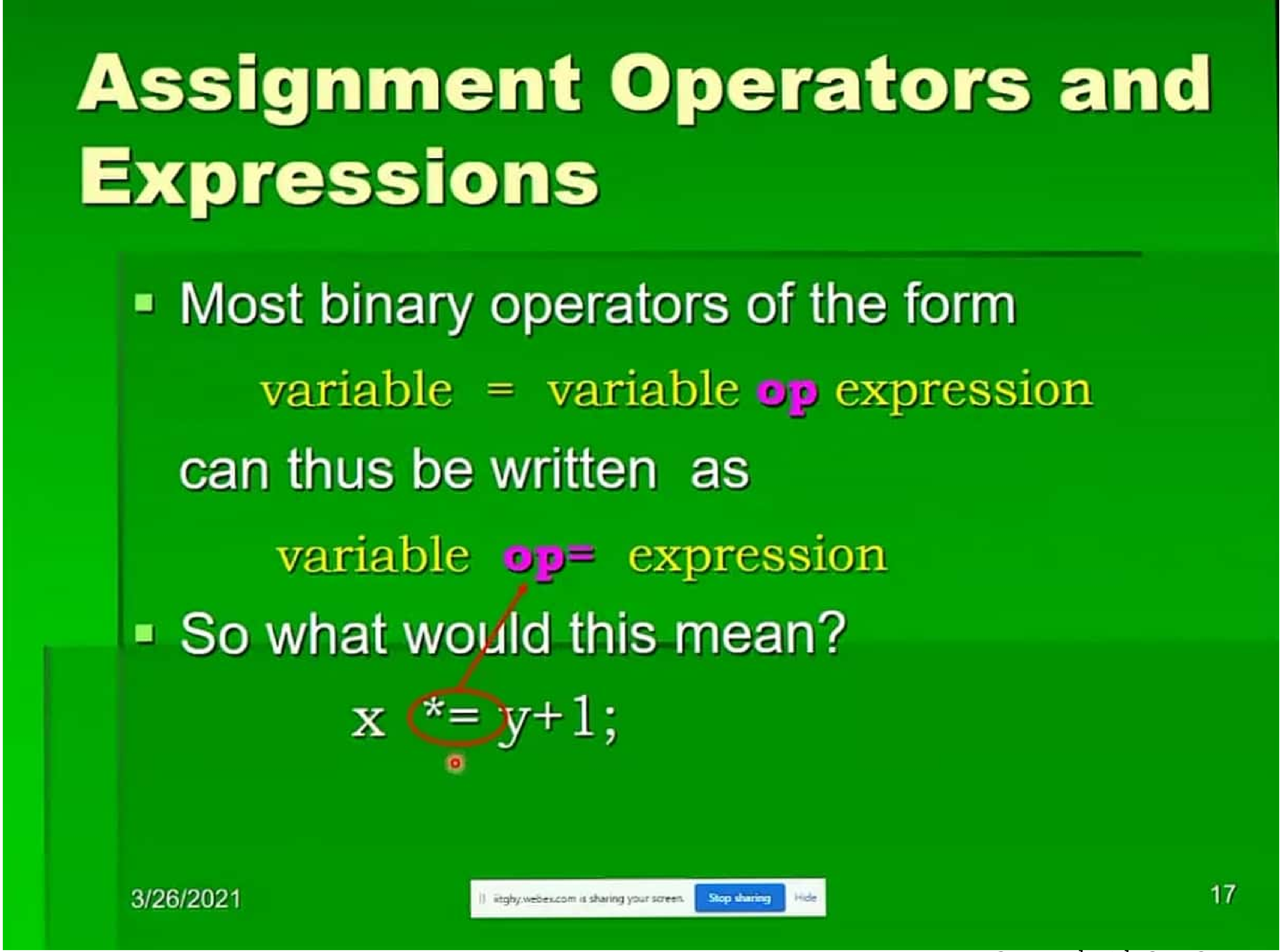
- `i = i+2 ;` can be written in a compressed form as
- `i += 2;`

i = i+2

= i+2

i+= 2

3/26/2021



Assignment Operators and Expressions

- Most binary operators of the form

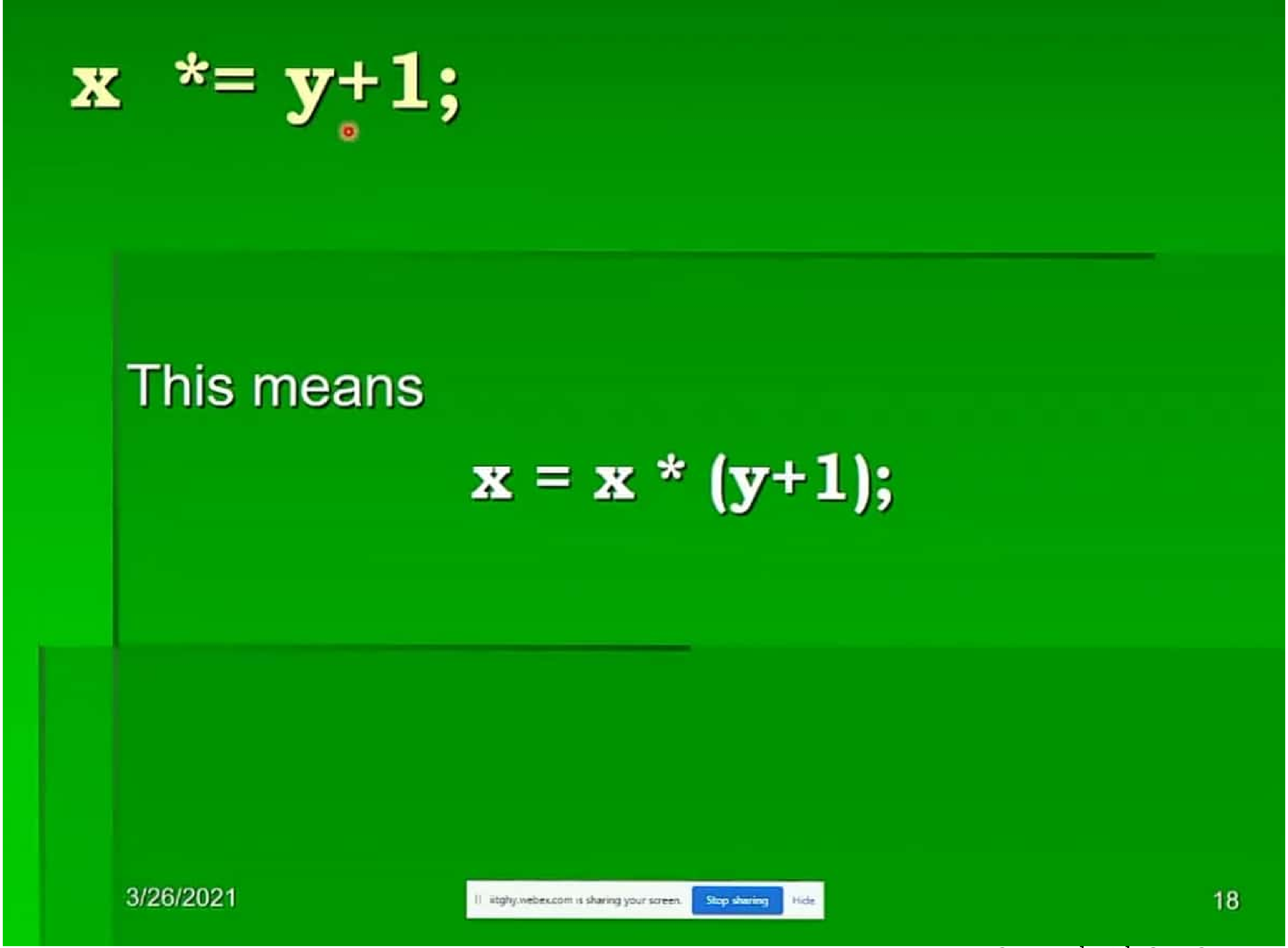
  variable = variable **op** expression

  can thus be written as

  variable **op=** expression

- So what would this mean?

```
x *= y+1;
```

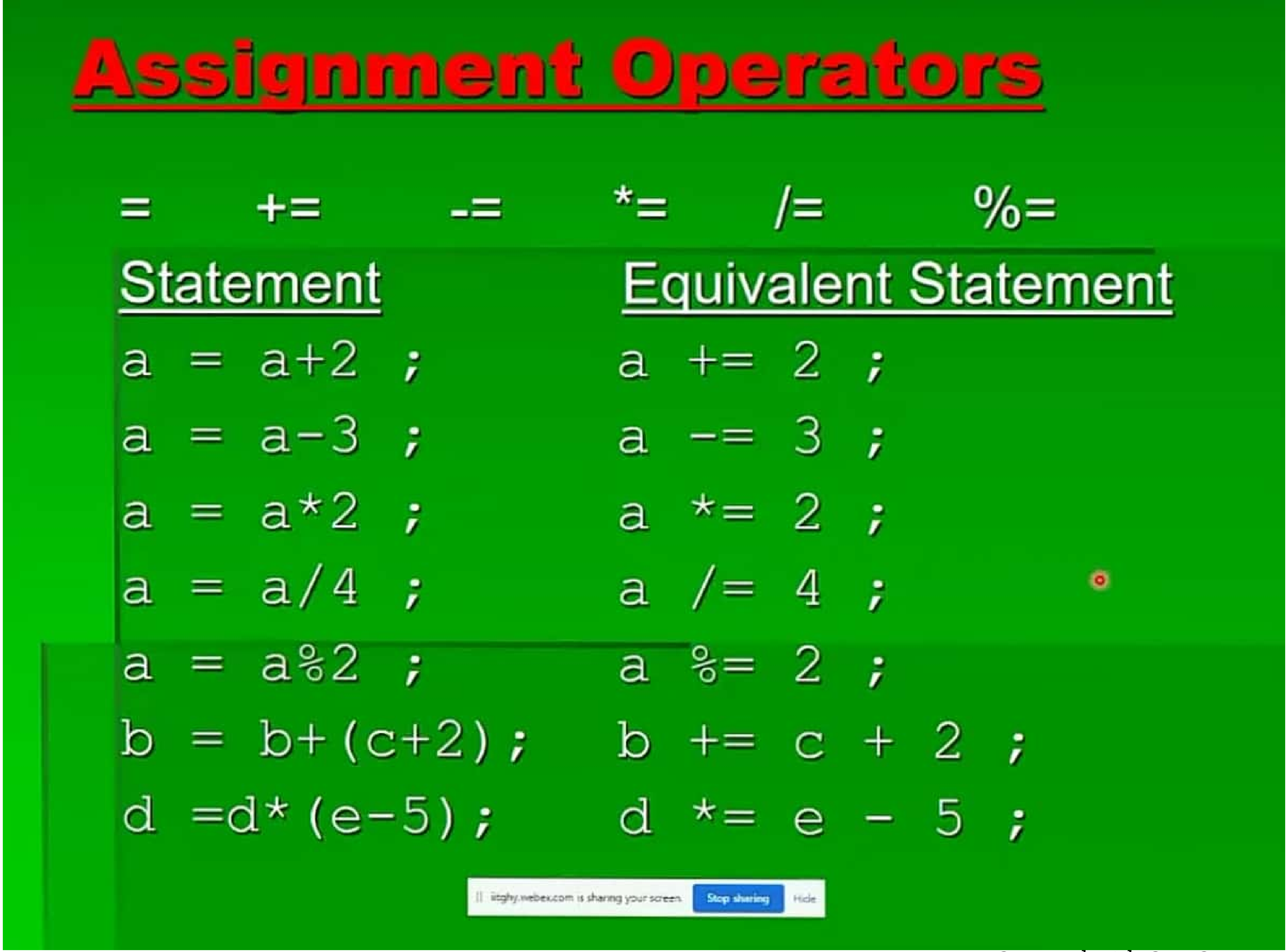


```
x  *= y+1;
```

This means

```
x = x * (y+1);
```

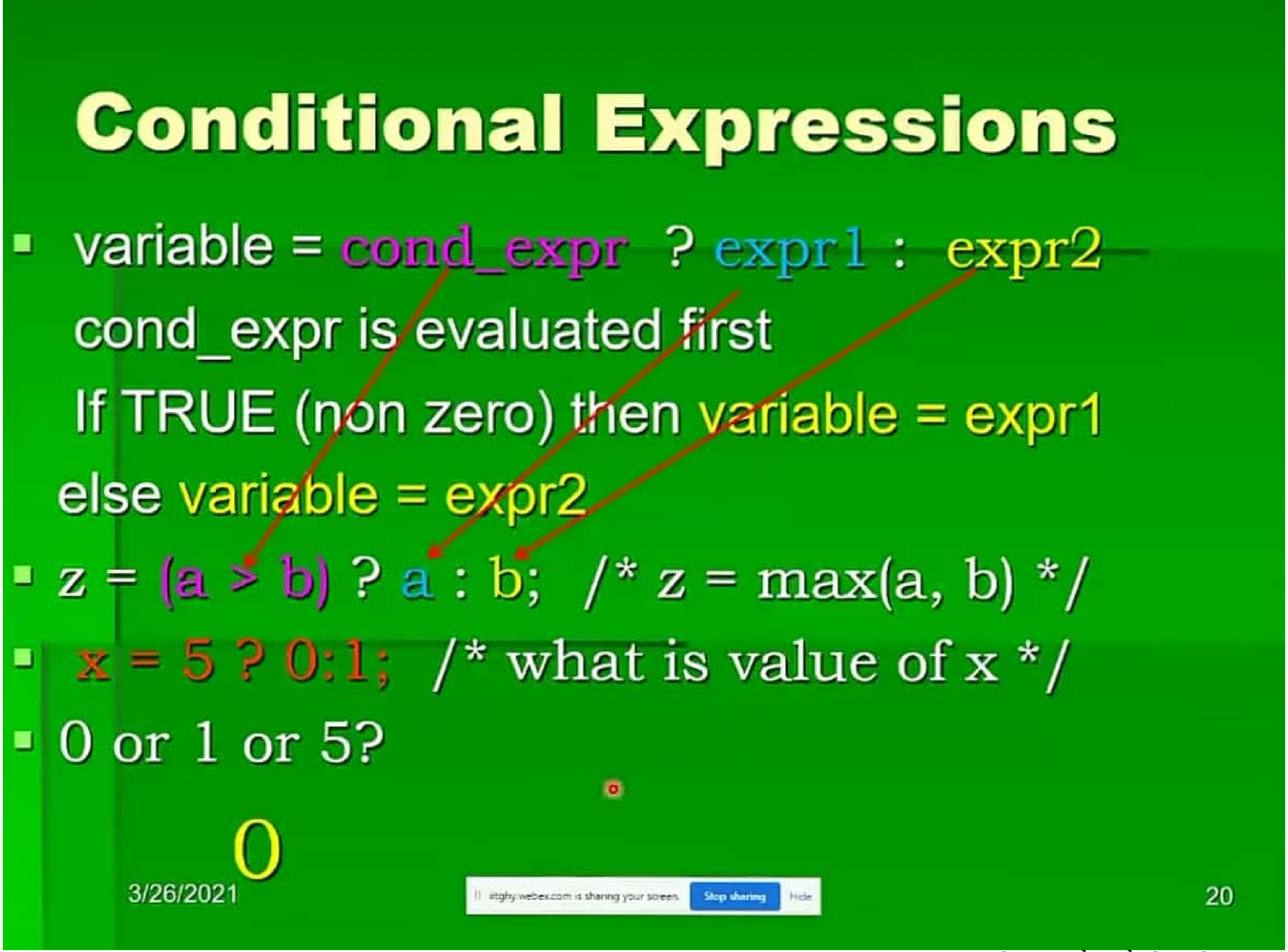
itghy.webex.com is sharing your screen.



Assignment Operators

=   +=   -=   *=   /=   %=

Statement	Equivalent Statement
`a = a+2 ;`	`a += 2 ;`
`a = a-3 ;`	`a -= 3 ;`
`a = a*2 ;`	`a *= 2 ;`
`a = a/4 ;`	`a /= 4 ;`
`a = a%2 ;`	`a %= 2 ;`
`b = b+(c+2);`	`b += c + 2 ;`
`d =d*(e-5);`	`d *= e - 5 ;`



Conditional Expressions

- variable = cond_expr ? expr1 : expr2

cond_expr is evaluated first

If TRUE (non zero) then variable = expr1

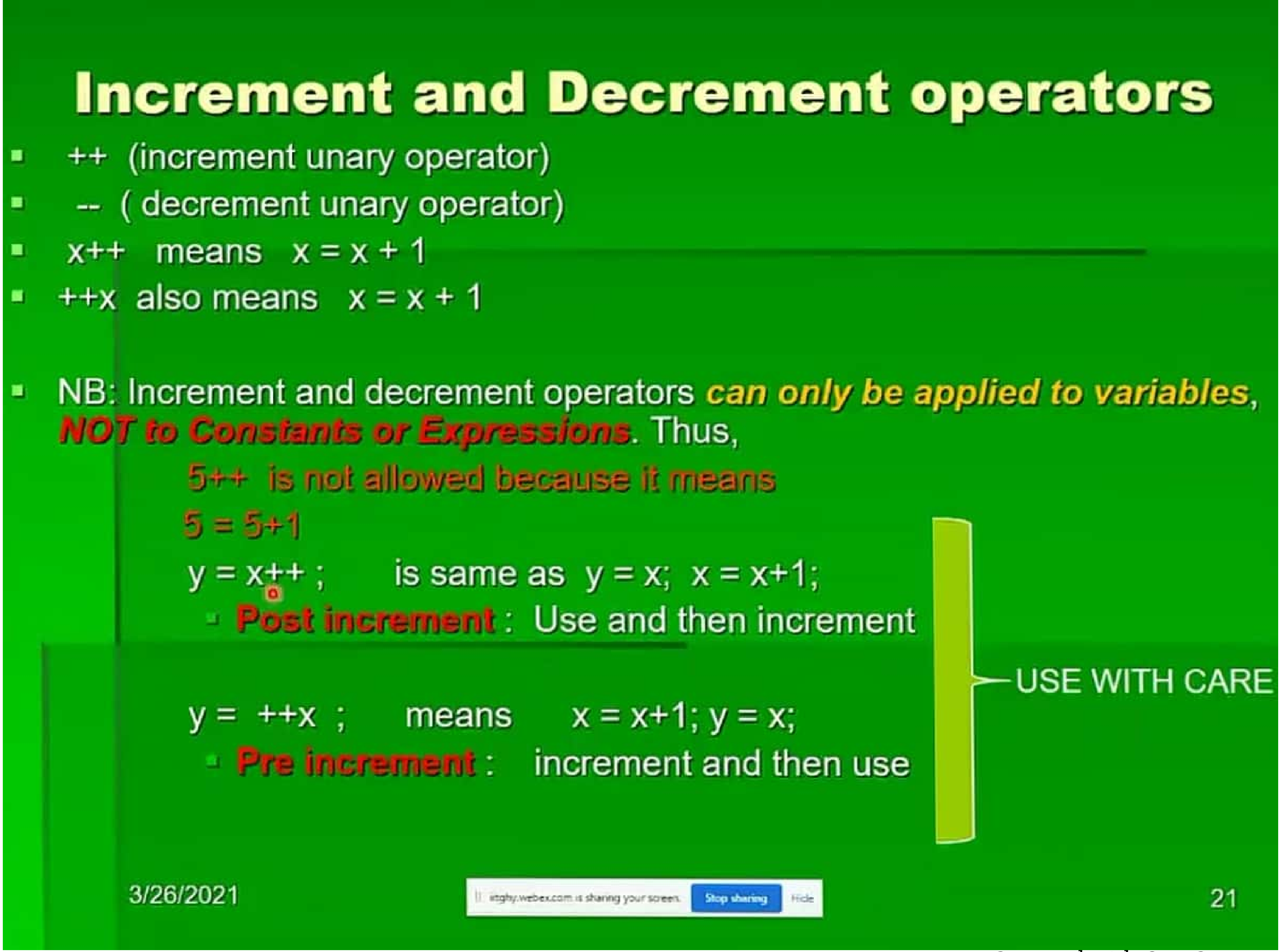
else variable = expr2

- z = (a > b) ? a : b; /* z = max(a, b) */
- x = 5 ? 0:1; /* what is value of x */
- 0 or 1 or 5?

0

3/26/2021

itghy.webex.com is sharing your screen.



Increment and Decrement operators

- ++ (increment unary operator)
- -- (decrement unary operator)
- x++ means x = x + 1
- ++x also means x = x + 1
- NB: Increment and decrement operators ***can only be applied to variables***, ***NOT to Constants or Expressions***. Thus,

  5++ is not allowed because it means

  5 = 5+1

  y = x++ ; is same as y = x; x = x+1;

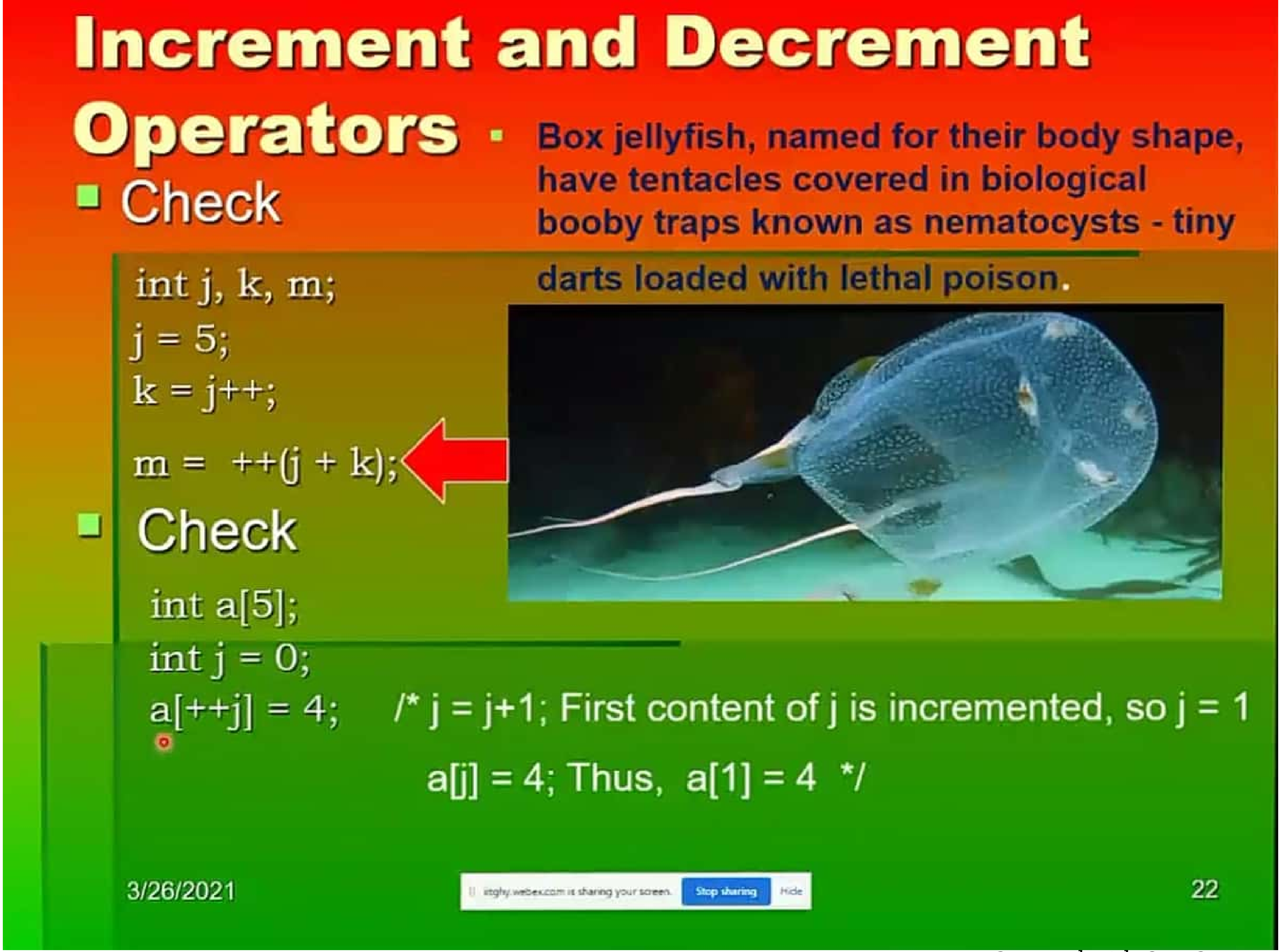
  - **Post increment** : Use and then increment

  y = ++x ; means x = x+1; y = x;

  - **Pre increment** : increment and then use

USE WITH CARE

3/26/2021



Increment and Decrement Operators

- Box jellyfish, named for their body shape, have tentacles covered in biological booby traps known as nematocysts - tiny darts loaded with lethal poison.

- Check

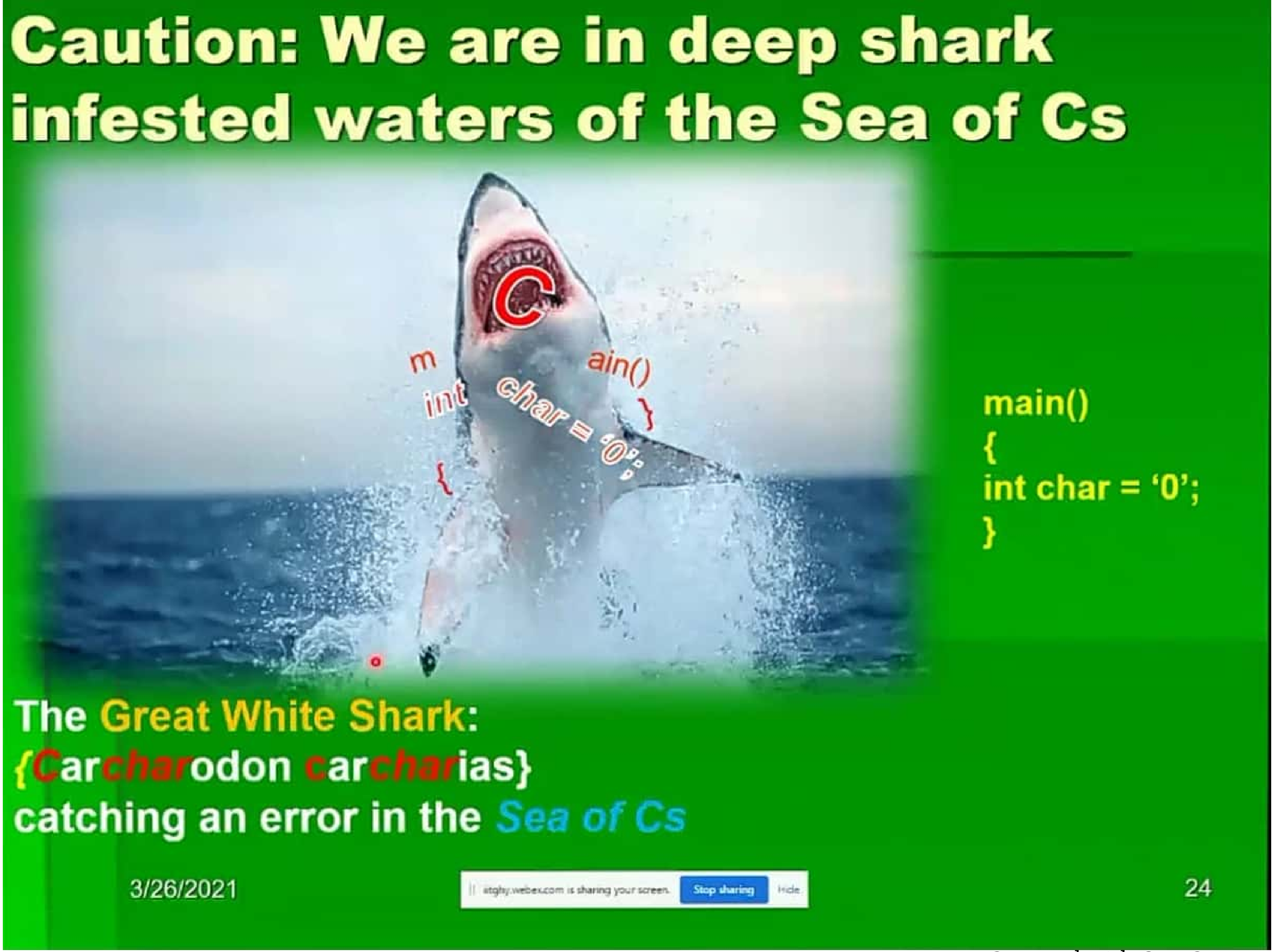
```
int j, k, m;
j = 5;
k = j++;

m =  ++(j + k);
```

- Check

```
int a[5];
int j = 0;
a[++j] = 4;     /* j = j+1; First content of j is incremented, so j = 1
                   a[j] = 4; Thus,  a[1] = 4  */
```

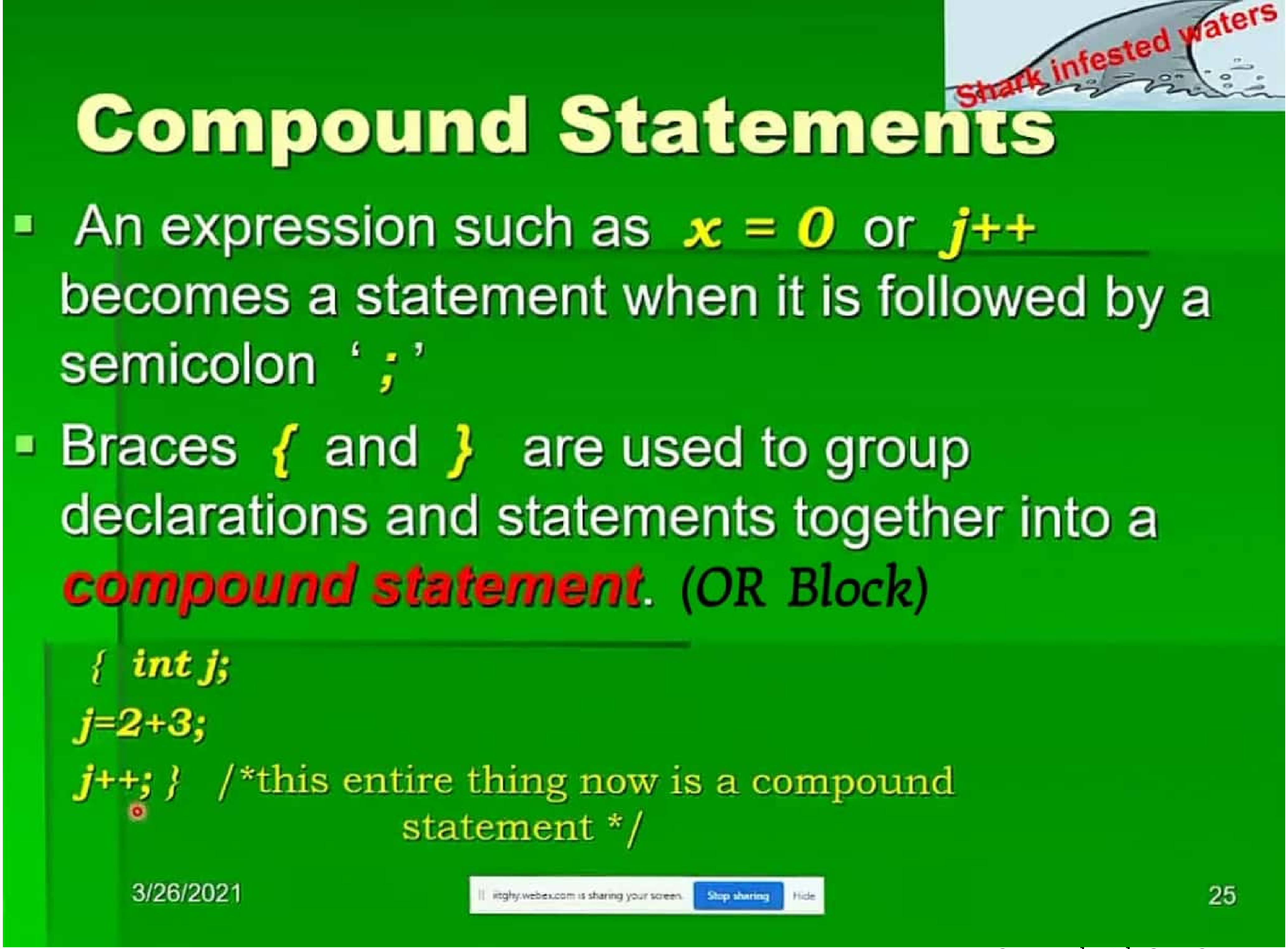
3/26/2021



Caution: We are in deep shark infested waters of the Sea of Cs

```
main()
{
int char = '0';
}
```

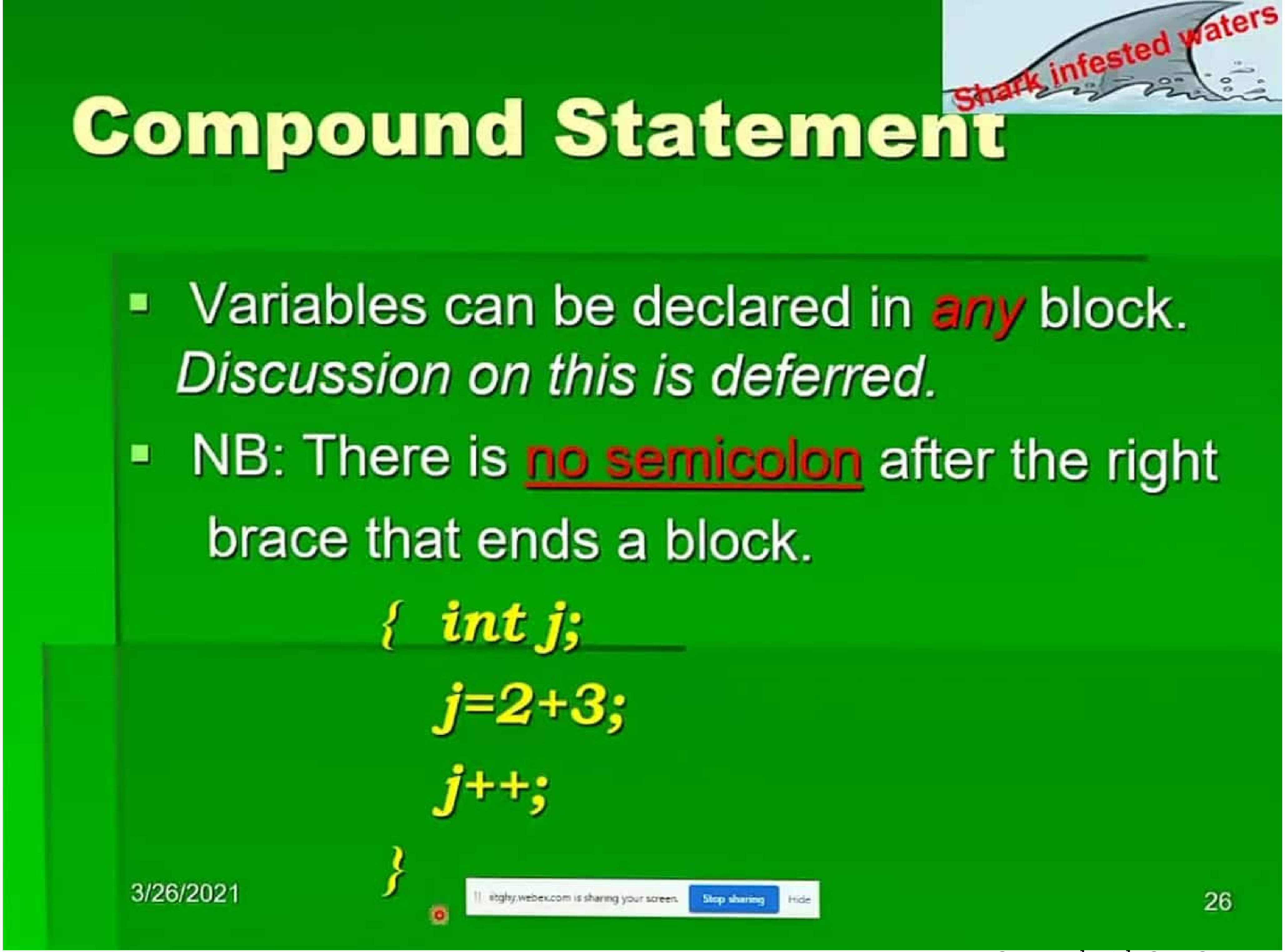
The Great White Shark:
{Carcharodon carcharias}
catching an error in the Sea of Cs



Compound Statements

- An expression such as *x = 0* or *j++* becomes a statement when it is followed by a semicolon ' ; '
- Braces { and } are used to group declarations and statements together into a **compound statement**. (OR Block)

```
{ int j;
j=2+3;
j++; }   /*this entire thing now is a compound
                 statement */
```



Compound Statement

- Variables can be declared in *any* block. *Discussion on this is deferred.*
- NB: There is no semicolon after the right brace that ends a block.

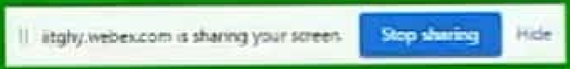
```
{  int j;
   j=2+3;
   j++;
}
```

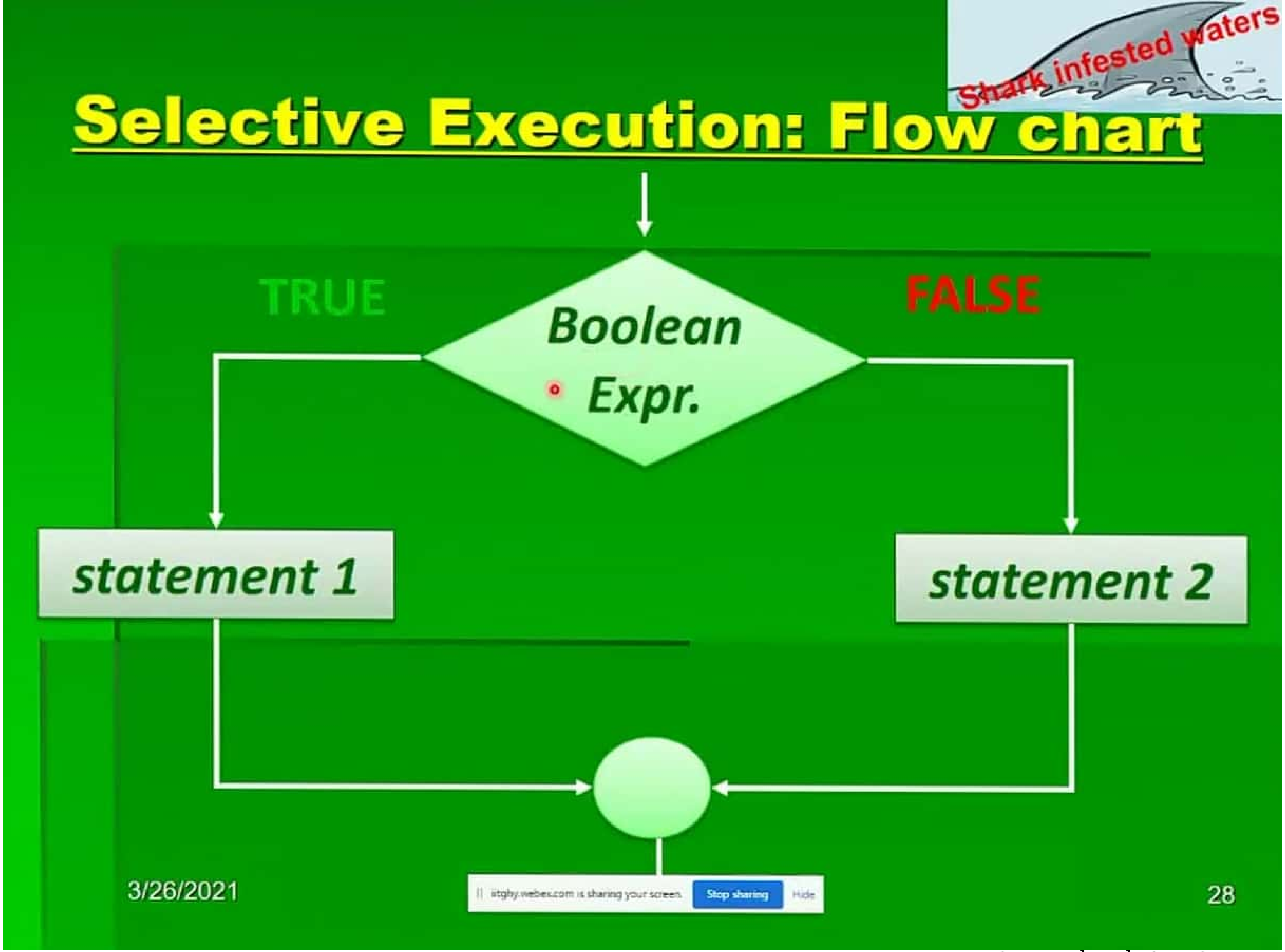
Selection: The *if* statement

```
if( condition )
{
 statement(s) /* body of the if
                      statement */
}
```

The braces are not required if the body contains only a single statement.

However, C Coding Standards recommend their inclusion.





Selective Execution: Flow chart

TRUE

Boolean Expr.

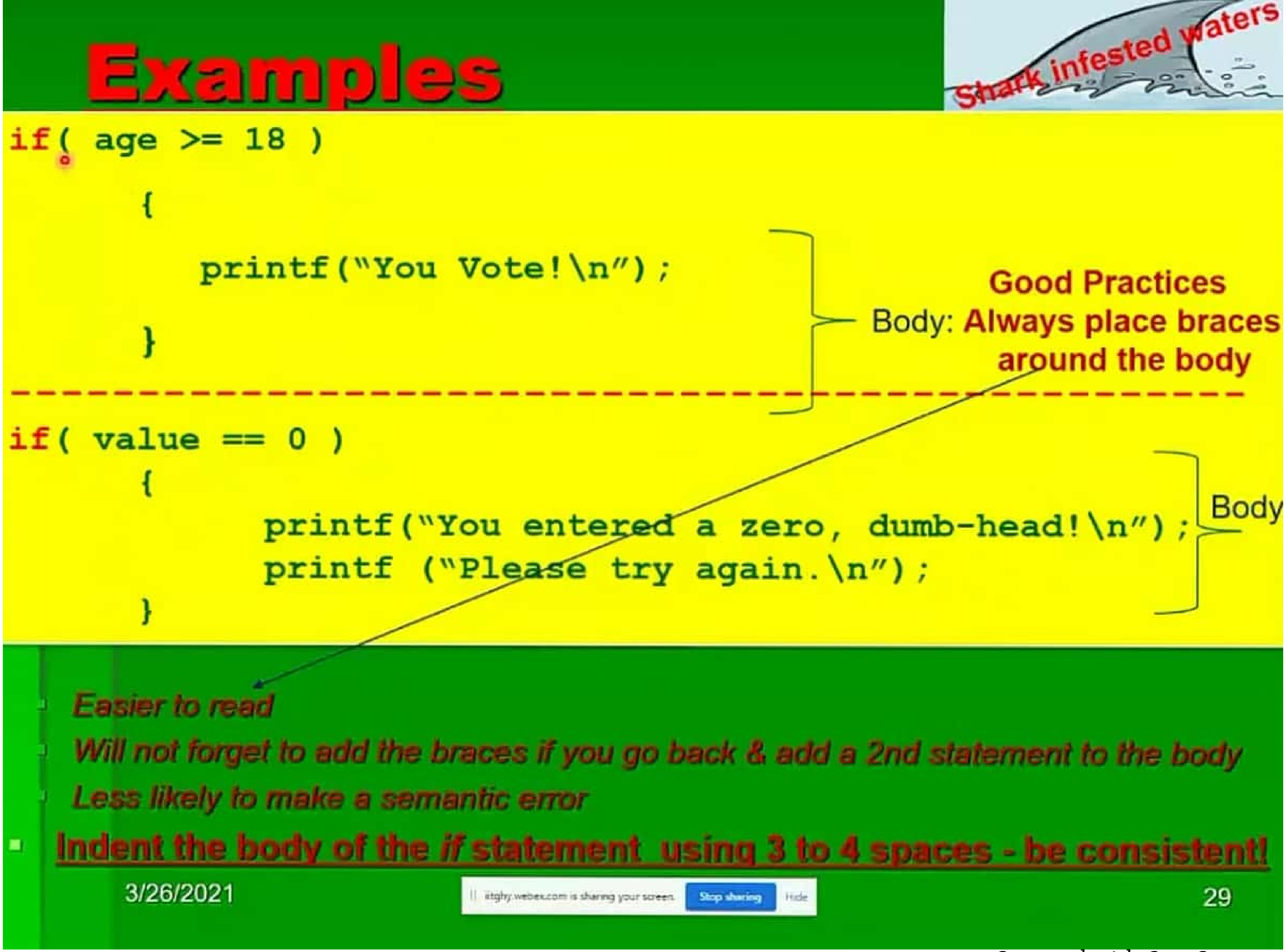
FALSE

statement 1

statement 2

|| iitghy.webex.com is sharing your screen. Stop sharing Hide

3/26/2021



Examples

```
if( age >= 18 )

        {

            printf("You Vote!\n");

        }
```

Good Practices

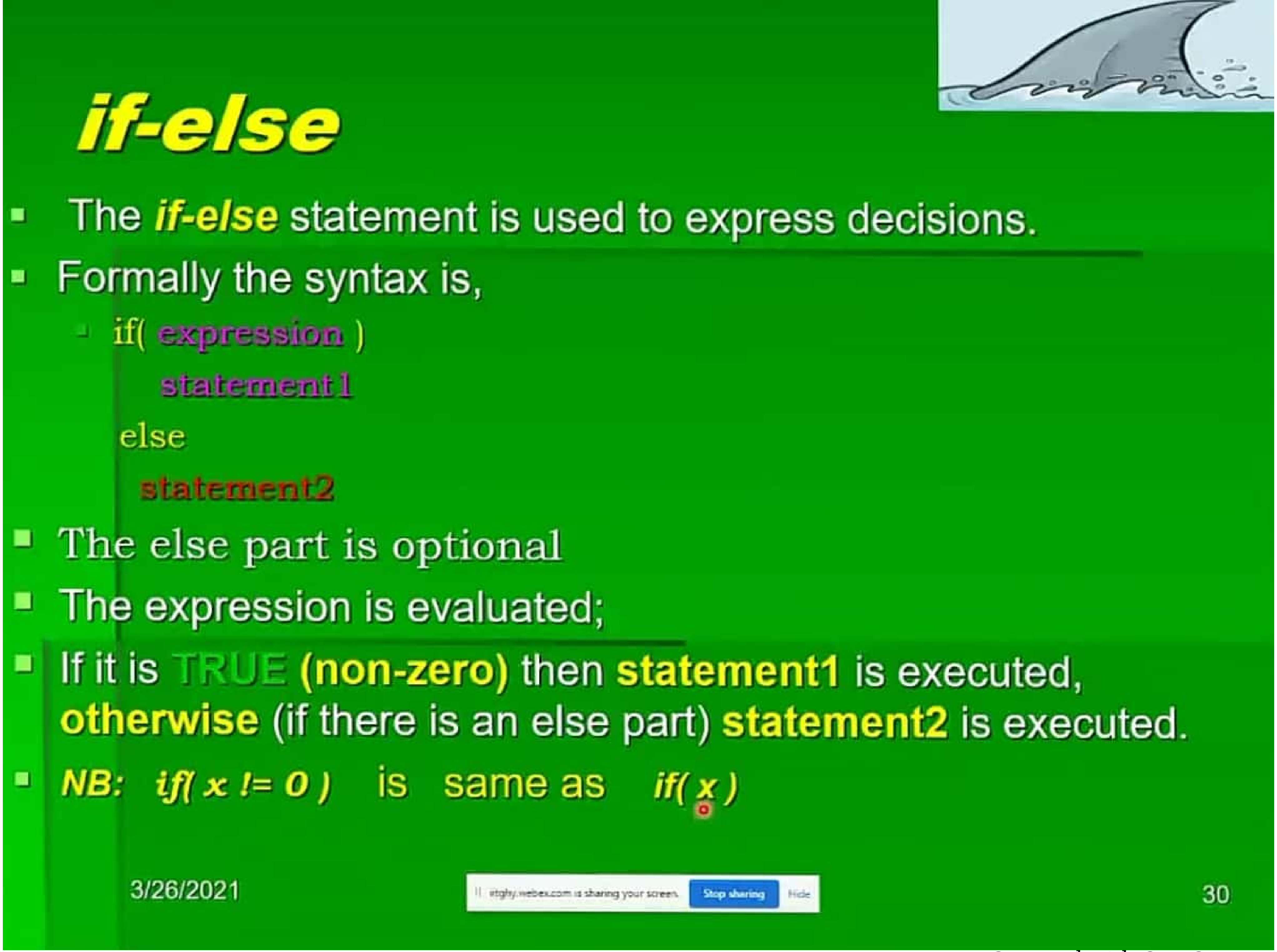
Body: Always place braces around the body

```
if( value == 0 )
        {
                printf("You entered a zero, dumb-head!\n");
                printf ("Please try again.\n");
        }
```

Body

- *Easier to read*
- *Will not forget to add the braces if you go back & add a 2nd statement to the body*
- *Less likely to make a semantic error*
- **Indent the body of the *if* statement using 3 to 4 spaces - be consistent!**

3/26/2021



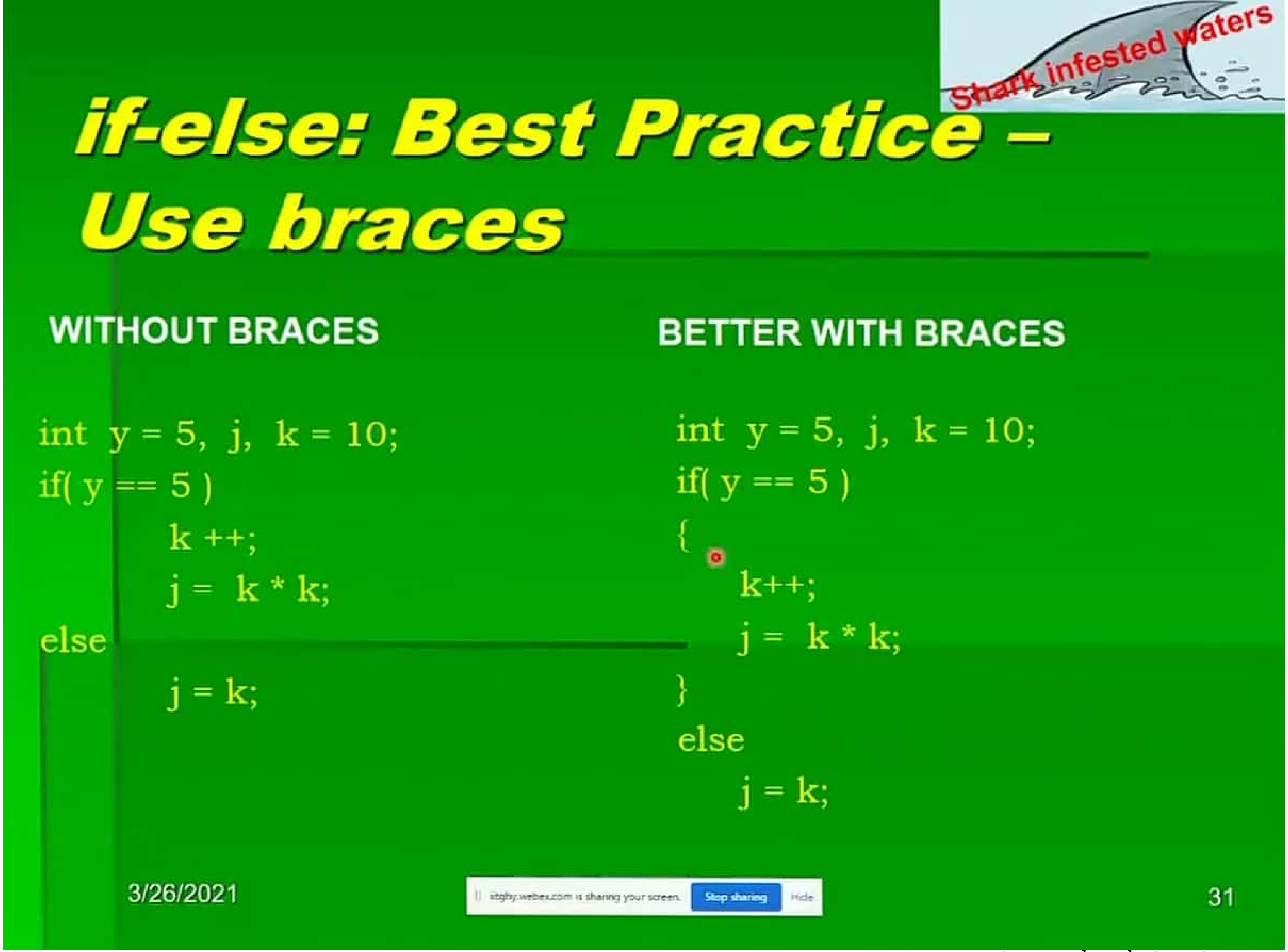
if-else

- The ***if-else*** statement is used to express decisions.
- Formally the syntax is,

```
if( expression )
    statement1
else
    statement2
```

- The else part is optional
- The expression is evaluated;
- If it is TRUE **(non-zero)** then **statement1** is executed, **otherwise** (if there is an else part) **statement2** is executed.
- ***NB:*** *if(x != 0)* is same as *if(x)*

itghy.webex.com is sharing your screen.



if-else: Best Practice – Use braces

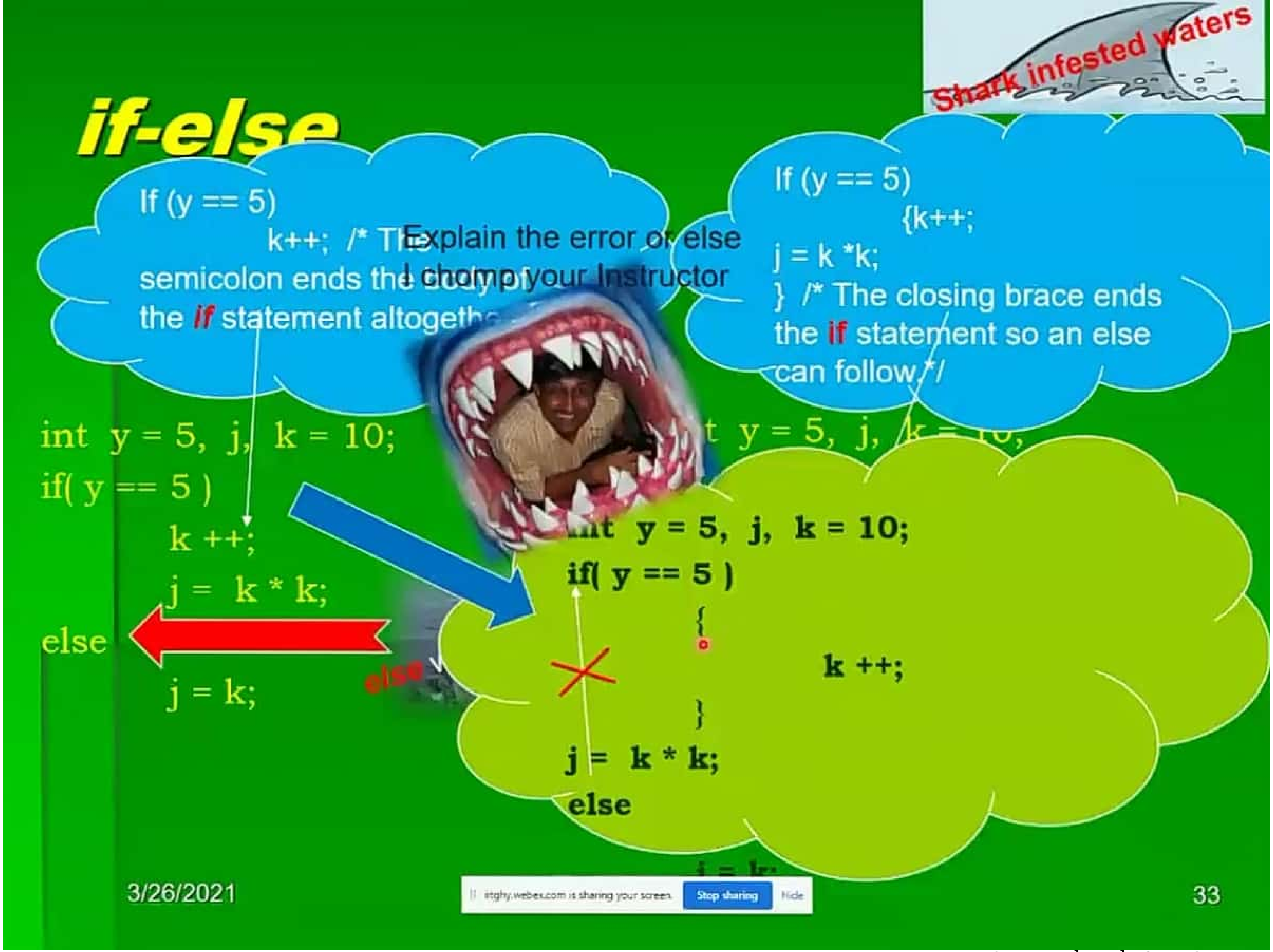
WITHOUT BRACES

```
int  y = 5,  j,  k = 10;
if( y == 5 )
          k ++;
          j =  k * k;
else
          j = k;
```

BETTER WITH BRACES

```
int  y = 5,  j,  k = 10;
if( y == 5 )
{
     k++;
     j =  k * k;
}
else
     j = k;
```

3/26/2021



if-else

```
If (y == 5)
        k++;  /* Th
semicolon ends the body of
the if statement altogeth
```

Explain the error or else
I chomp your Instructor

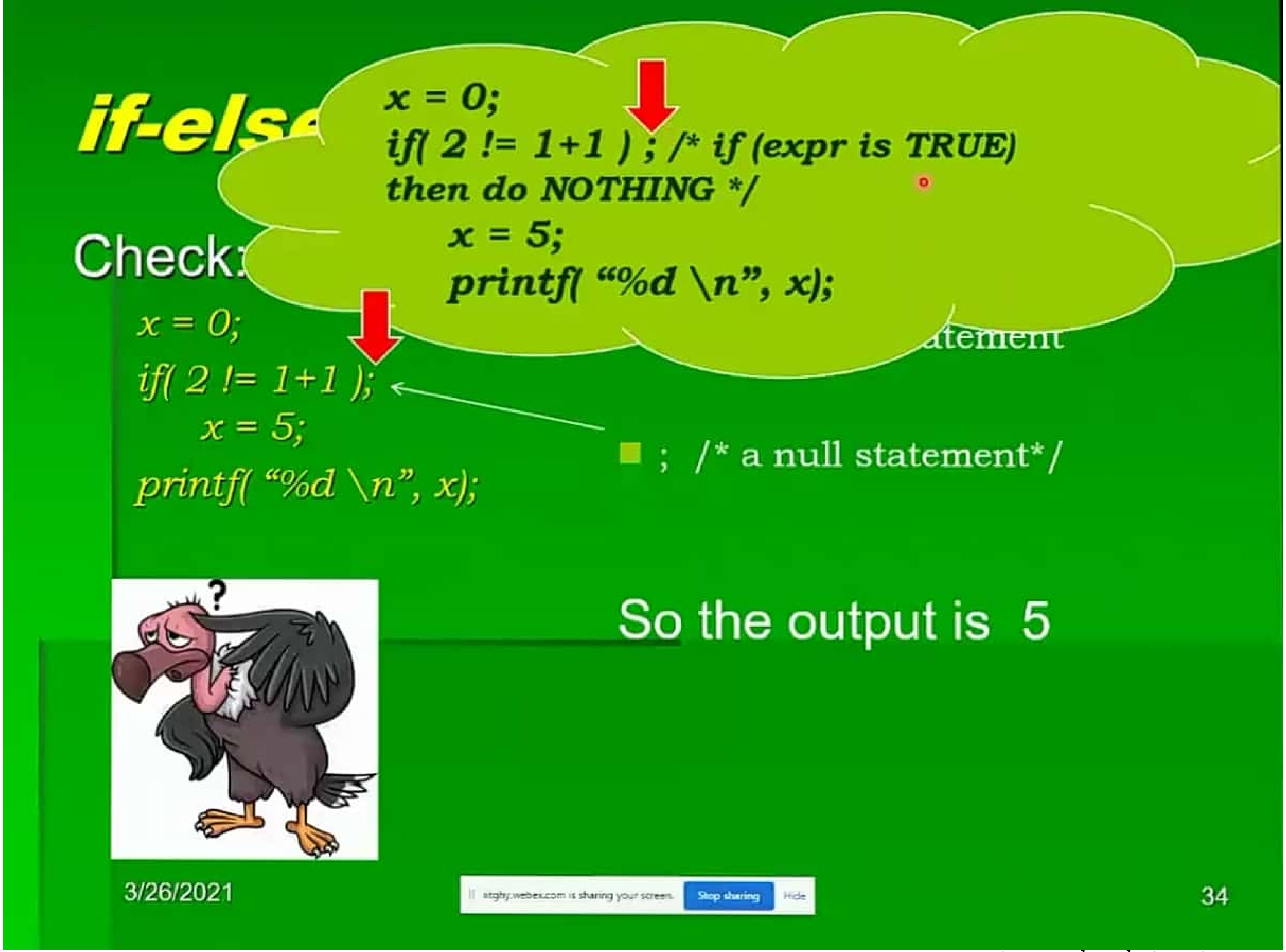
```
If (y == 5)
        {k++;
j = k *k;
}  /* The closing brace ends
the if statement so an else
can follow.*/
```

```
int  y = 5,  j,  k = 10;
if( y == 5 )
      k ++;
      j =  k * k;
else
      j = k;
```

```
t  y = 5,  j,  k = 10;
```

```
int  y = 5,  j,  k = 10;
if( y == 5 )
           {
                      k ++;
           }
j =  k * k;
else
           j = k;
```

itghy.webex.com is sharing your screen. Stop sharing Hide



if-els

```
x = 0;
if( 2 != 1+1 ) ; /* if (expr is TRUE)
then do NOTHING */
      x = 5;
      printf( "%d \n", x);
```

Check:

```
x = 0;
if( 2 != 1+1 );
      x = 5;
printf( "%d \n", x);
```

atement

- ; /* a null statement*/

So the output is 5

3/26/2021

If-else Pitfalls

Check:

```
int j = 200;
if( j = 5)
   printf("A");
else
   printf("B");
```

- What is the output?

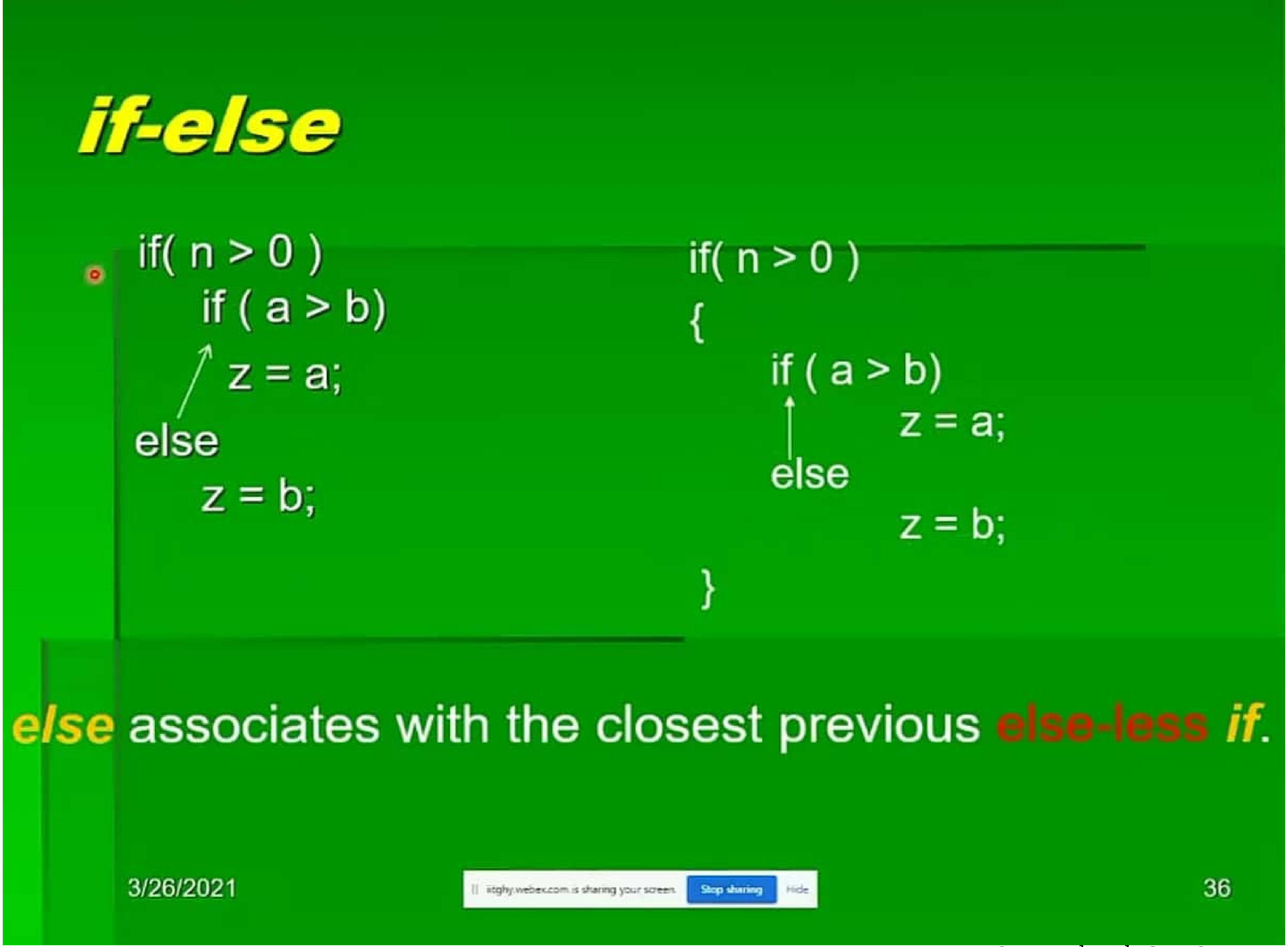
A

Because *j* has been assigned a value (=5) which is non-zero (**TRUE**).

NB: We are not checking whether *j* is equal to 5! If that is so we should have used *if(j==5)*

This is a common pitfall. Beware!

itghy.webex.com is sharing your screen. Stop sharing Hide

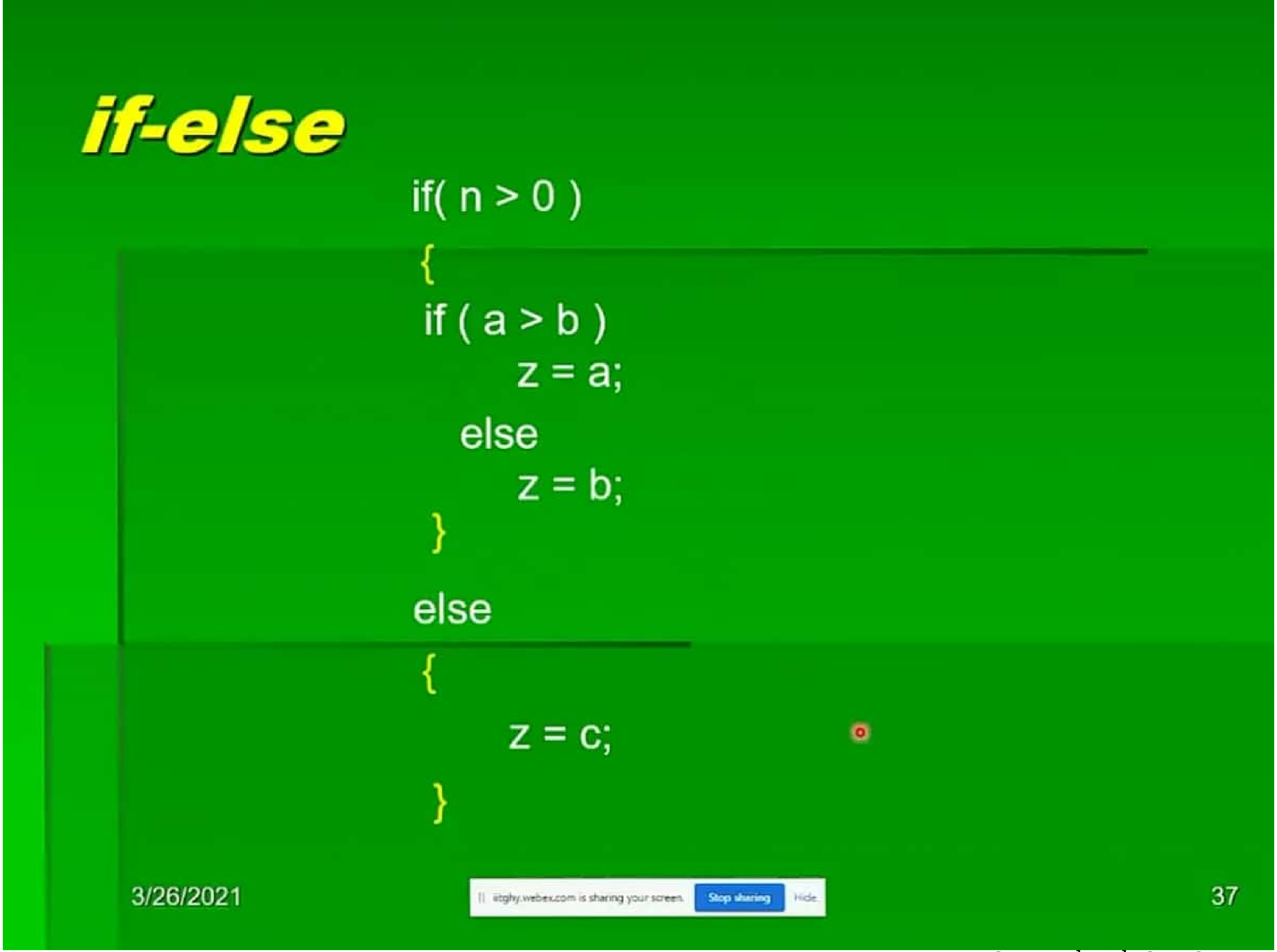


if-else

```
if( n > 0 )
    if ( a > b)
        z = a;
else
    z = b;
```

```
if( n > 0 )
{
    if ( a > b)
            z = a;
    else
            z = b;
}
```

else associates with the closest previous *else-less if*.



if-else

```
if( n > 0 )
 {
 if ( a > b )
        z = a;
    else
        z = b;
  }
else
 {
        z = c;
  }
```

iitghy.webex.com is sharing your screen.