The stage is set for a new lecture to commence

We can use the if statement
to take decisions

```
int input = 5;
int res   = 0;
if( input>3 ){ res = 10; }
assert res == 10;

--------------------

int input = 2;
int res   = 0;
if( input>3 ){ res = 10; }
assert res == 0;
```

In the first example,
input is 5,
the if condition holds
Thus we execute the body
and we get 10

In the second example,
input is 2,
the if condition does not holds
Thus we skip the body
and we get 0

Last time we discussed
local variable update

```
int input = 2;
int res   = 0;
int value = 0;
if( (value=input*10)>40 ){ res = 10; }
assert res == 0;
assert value == 20;
```

We can also use
it inside expressions.

It is common inside
conditions.

Value equal input times ten
is an expression producing
the assigned value.


```
int input = 2;
int res   = 0;
int value = 0;
if( (value=input*10)>40 ){ res = 10; }
assert res == 0;
assert value == 20;
```

Step by step

Input is 2

2 times 10 is 20

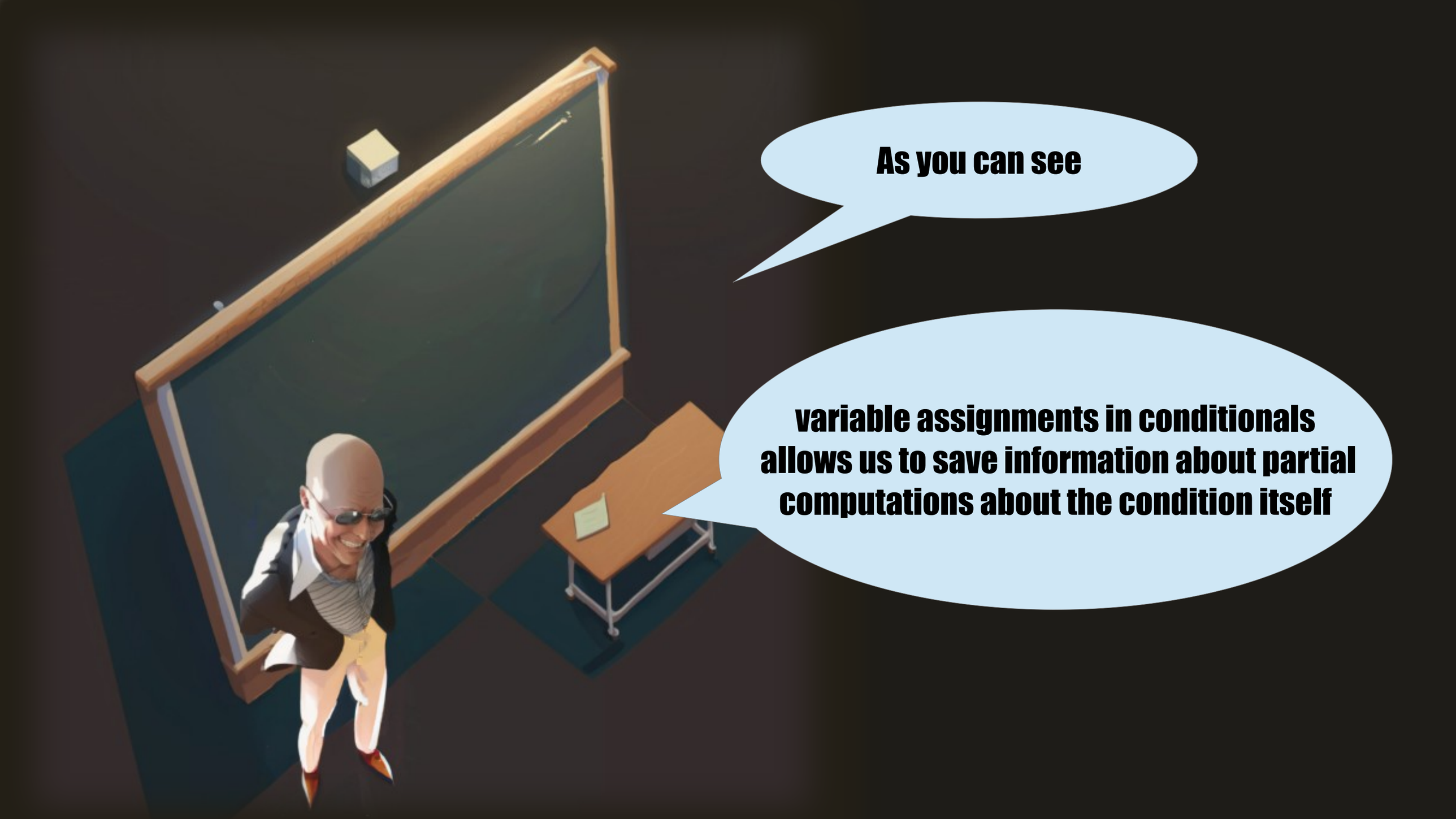
```
int input = 2;
int res   = 0;
int value = 0;
if( (value=2*10)>40 ){ res = 10; }
assert res == 0;
assert value == 20;
```

We update value to 20

value = 20 is now just ‘20’

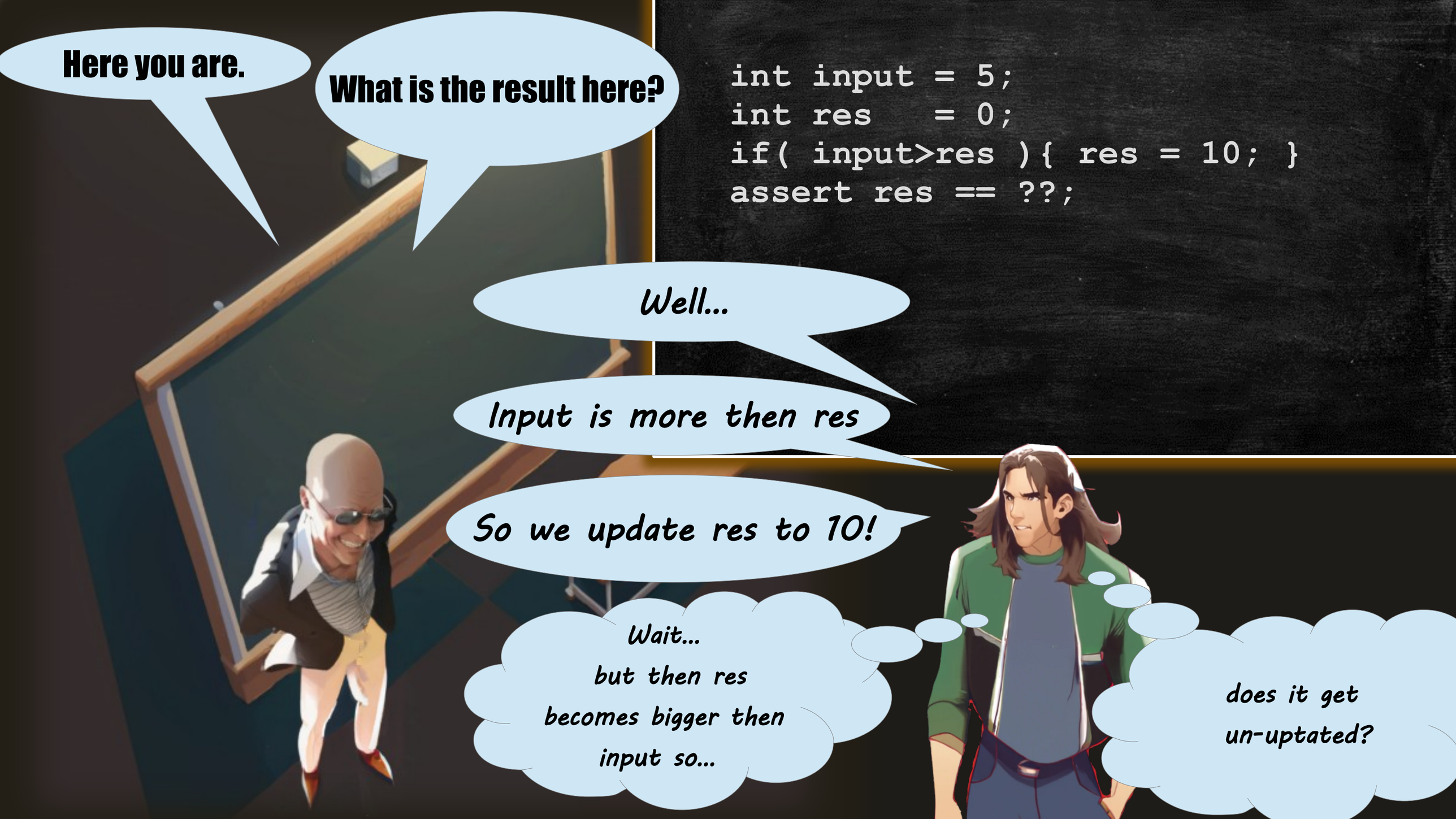
20 is not bigger then 40
so we get false.

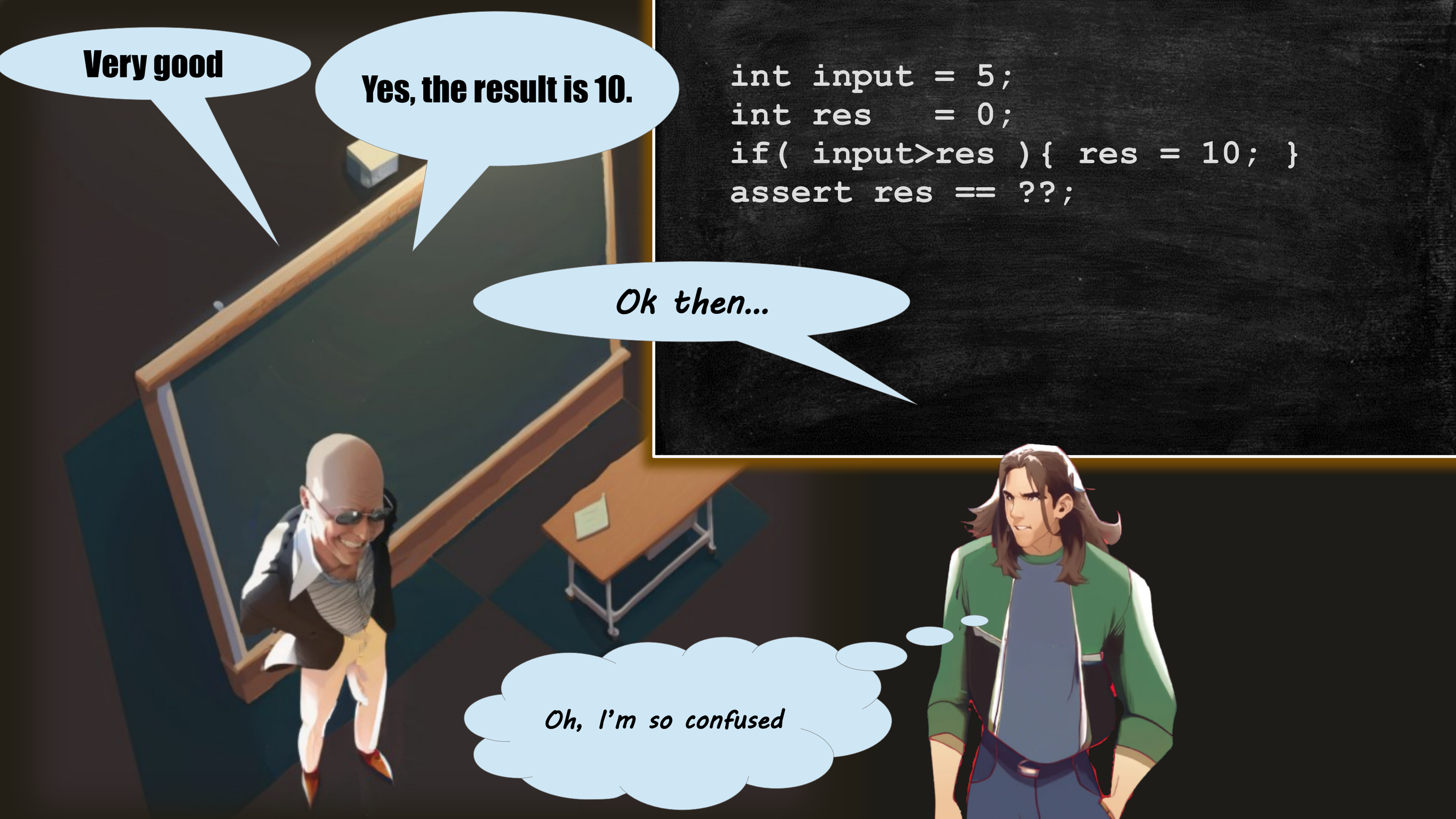
with false, we skip the if body


As you can see
variable assignments in conditionals allows us to save information about partial computations about the condition itself


So, you, .. come here!

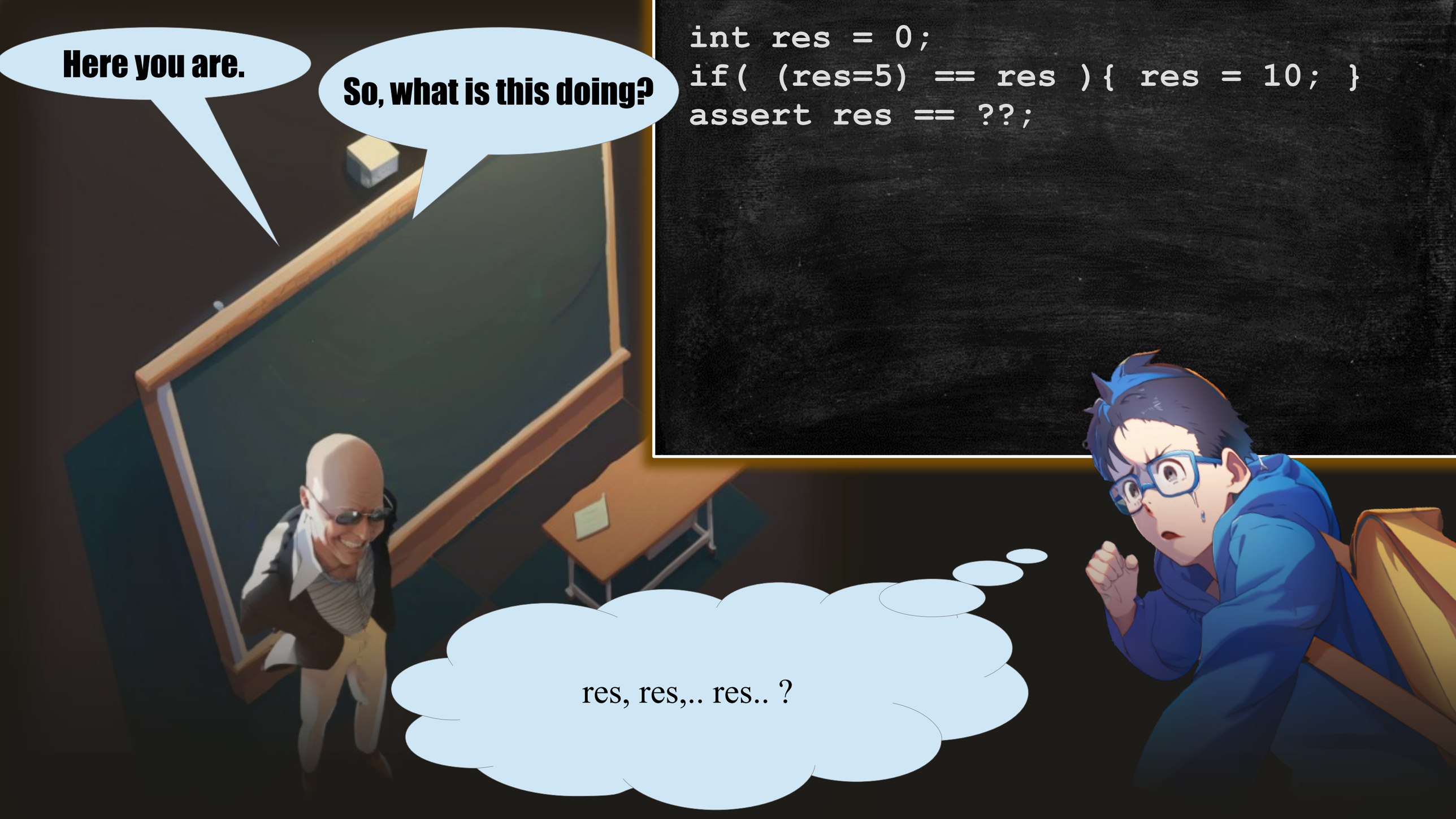

So, you, .. come here!
Do you mean... Me?!?

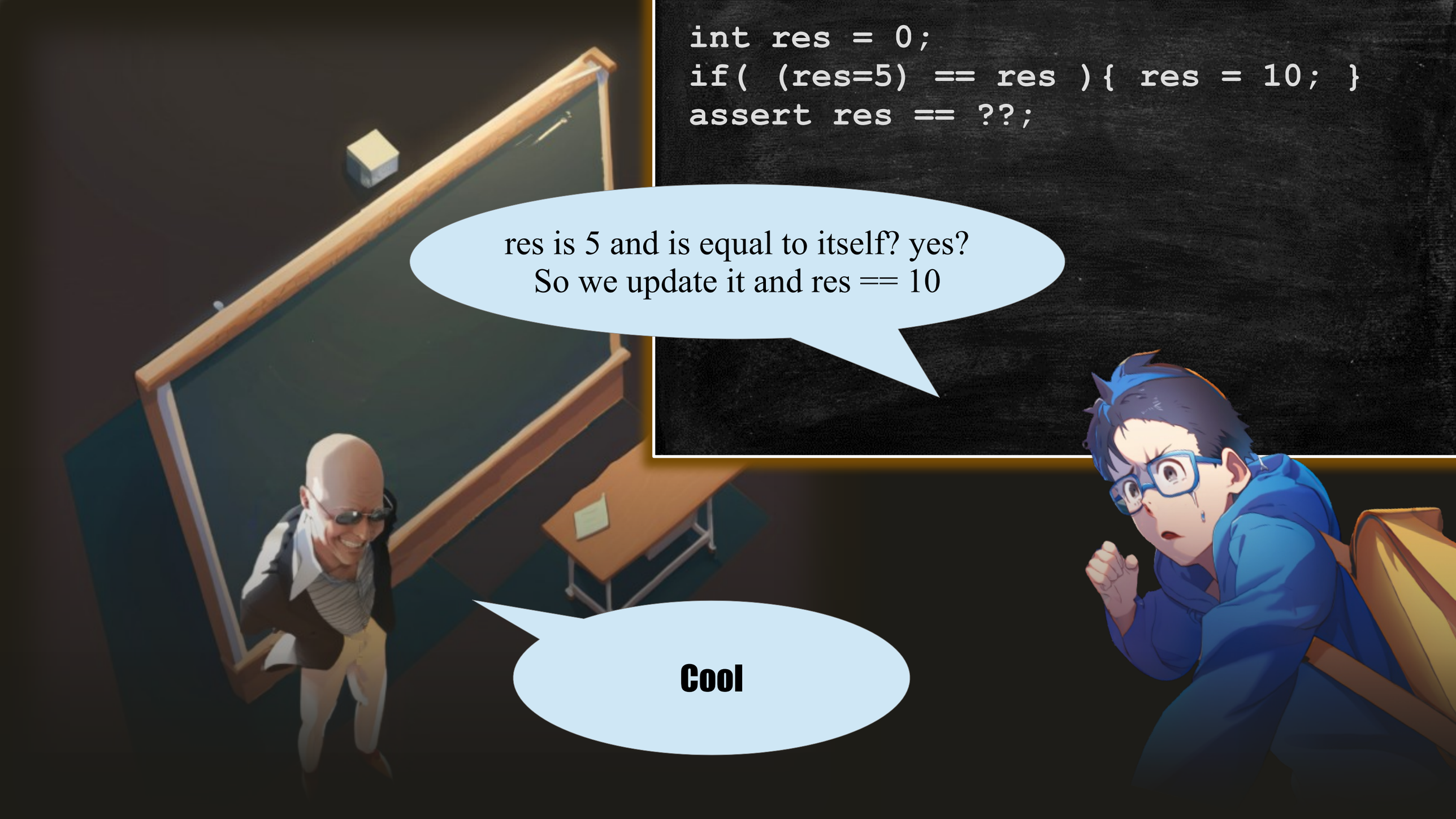

Here you are.
What is the result here?
int input = 5;
int res = 0;
if(input>res){ res = 10; }
assert res == ??;
Well...
Input is more then res
So we update res to 10!
Wait...
but then res
becomes bigger then
input so...
does it get
un-uptated?

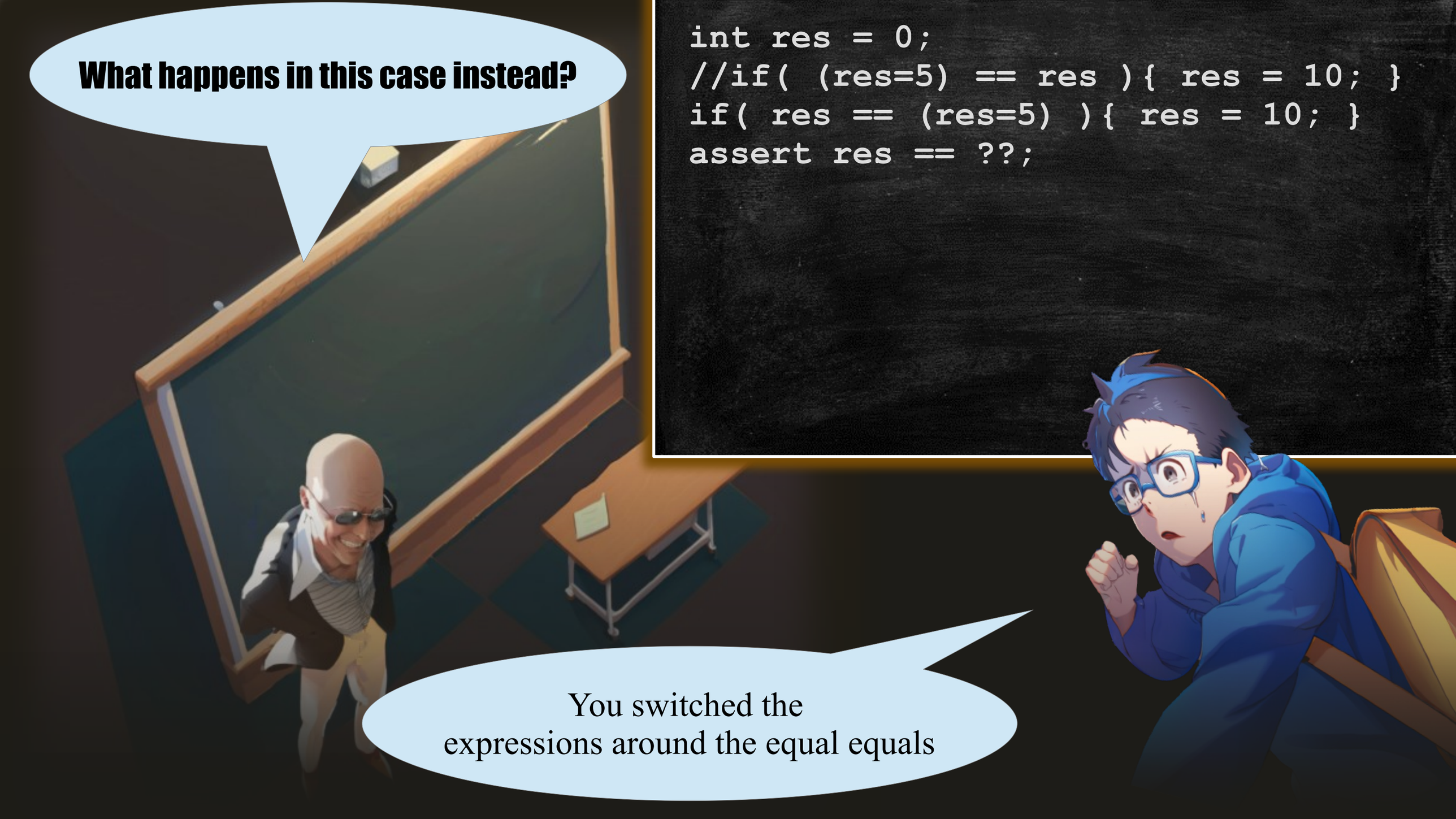

Very good
Yes, the result is 10.
int input = 5;
int res = 0;
if(input>res){ res = 10; }
assert res == ??;
Ok then...
Oh, I'm so confused


Me?!?
Next, you above!


This is it,
he is going to take revenge!


Here you are.
So, what is this doing?
int res = 0;
if((res=5) == res){ res = 10; }
assert res == ??;
res, res,.. res.. ?


int res = 0;
if((res=5) == res){ res = 10; }
assert res == ??;
res is 5 and is equal to itself? yes?
So we update it and res == 10
Cool


What happens in this case instead?
int res = 0;
//if((res=5) == res){ res = 10; }
if(res == (res=5)){ res = 10; }
assert res == ??;
You switched the
expressions around the equal equals

So?
what is the result?

```
int res = 0;
//if( (res=5) == res ){ res = 10; }
if( res == (res=5) ){ res = 10; }
assert res == ??;
```

It looks like is the same,
I mean, == is clearly a
commutative operation.

So?
what is the result?

```
int res = 0;
//if( (res=5) == res ){ res = 10; }
if( res == (res=5) ){ res = 10; }
assert res == ??;
```

Stop the video and try to answer!

Can you beat the
‘equals switcheroo’ ?

So?
what is the result?

```
int res = 0;
//if( (res=5) == res ){ res = 10; }
if( res == (res=5) ){ res = 10; }
assert res == ??;
```

It looks like is the same,
I mean, == is clearly a
commutative operation.


Wait a minute, would he even ask the question if it was the same?


There must be a difference then!


Or.. is he thinking that
I will think this,
and it is thus the same??


It is as if we were
playing poker!

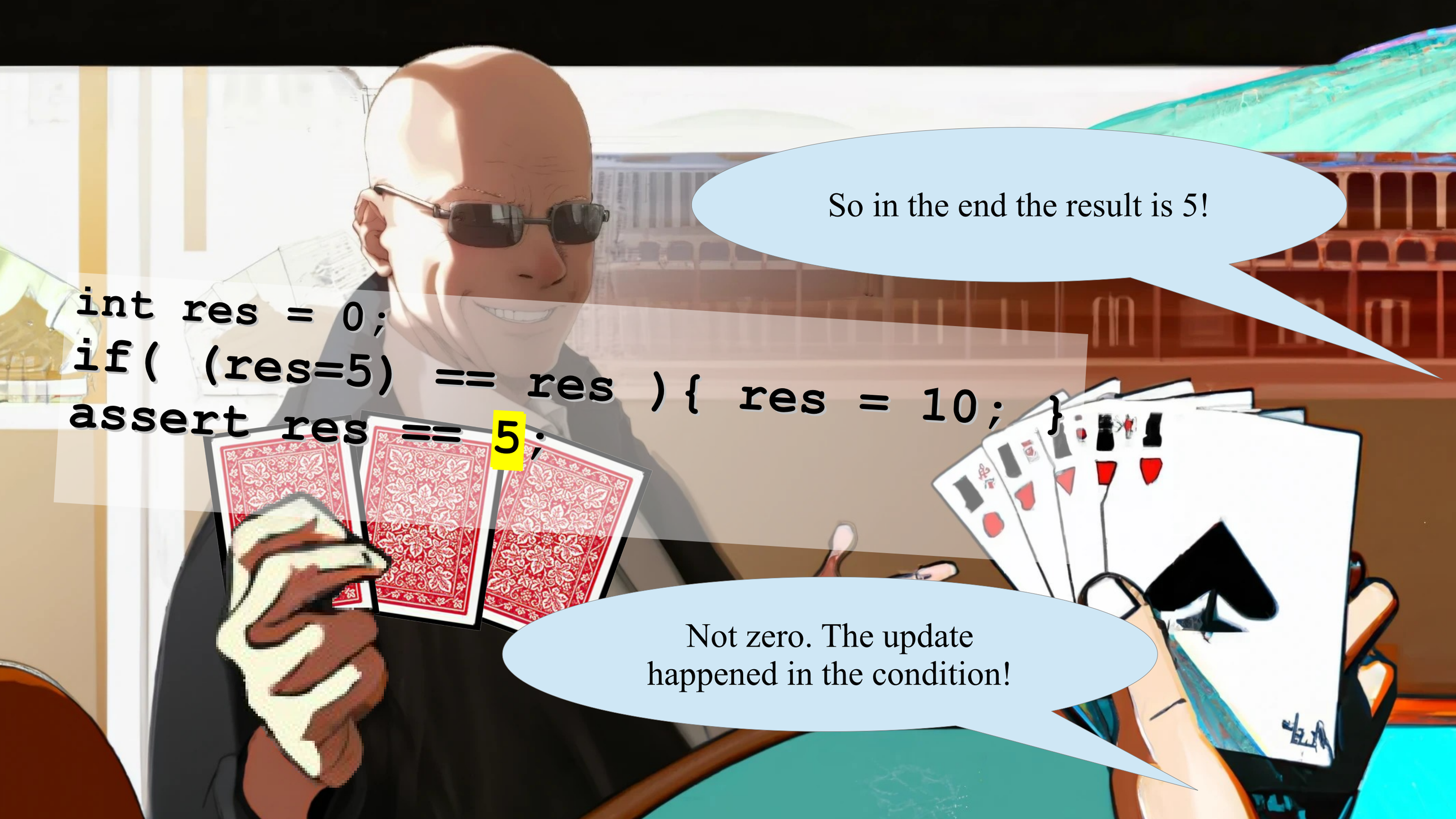

What is going on?
We ARE playing poker!
Another hallucination?

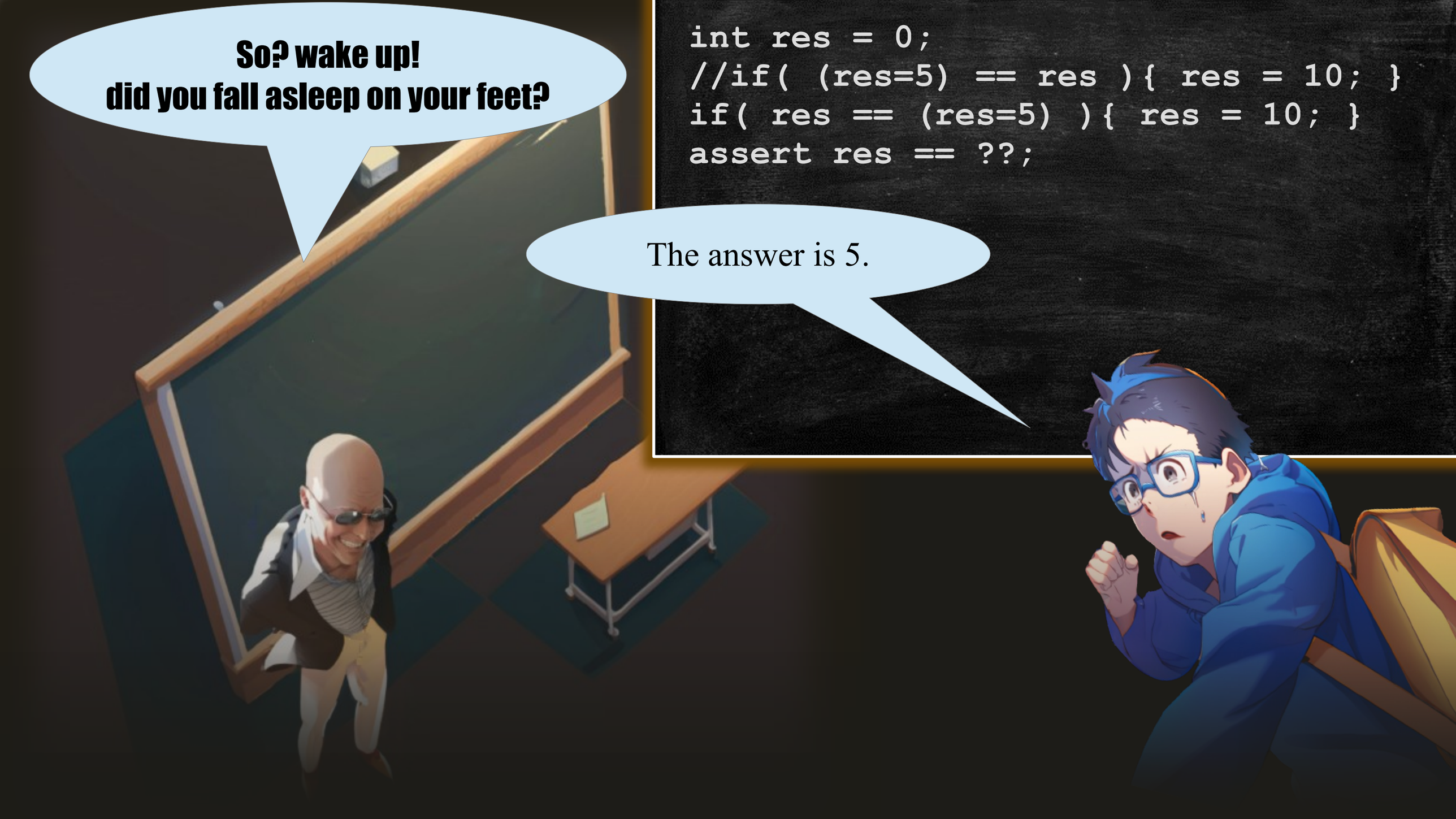

int res = 0;
if(res==(res=5)){res=10;}
Pupon has 3 cards in his hand.
They are the 3 res
By the way, my cards are not very good..

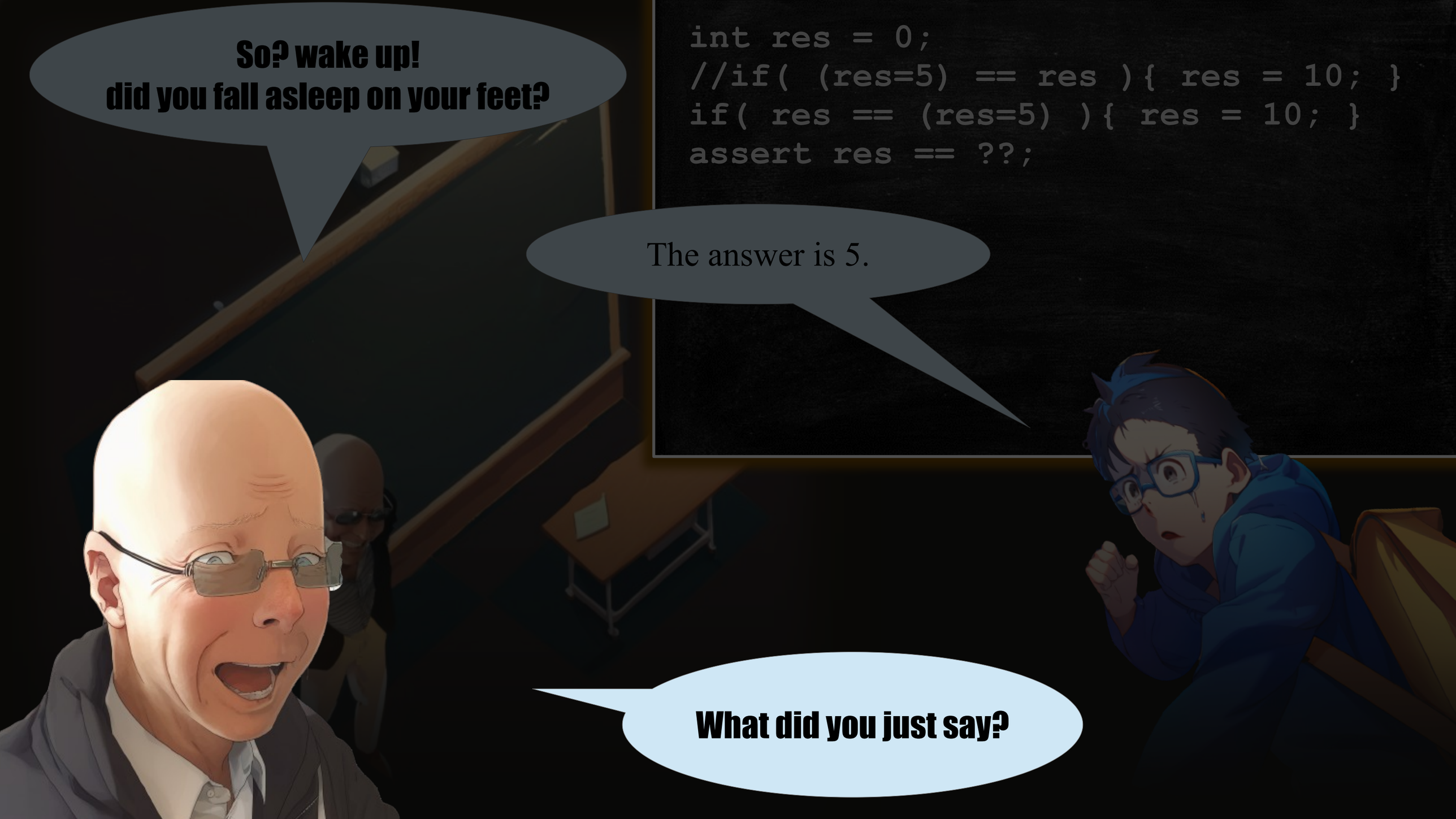

The first one must be zero.
Because at the start res is zero.
int res = 0;
if(res==(res=5)){res=10;}
0


int res = 5;
if(res==(res=5)){res=10;}
0
5
The first one must be zero.
Because at the start res is zero.
The second is five:
the variable update returns the update value.
Also res is now 5.


int res = 5;
if(res==(res=5)){res=10;}
0
5
The first card is evaluated already, so it is not changing
zero is not equal to 5, so we skip the body of the if!
So the last card is not played!


So in the end the result is 5!
int res = 0;
if((res=5) == res){ res = 10; }
assert res == 5;
Not zero. The update
happened in the condition!


So? wake up!
did you fall asleep on your feet?
int res = 0;
//if((res=5) == res){ res = 10; }
if(res == (res=5)){ res = 10; }
assert res == ??;
The answer is 5.


So? wake up!
did you fall asleep on your feet?
int res = 0;
//if((res=5) == res){ res = 10; }
if(res == (res=5)){ res = 10; }
assert res == ??;
The answer is 5.
What did you just say?


int res = 0;
//if((res=5) == res){ res = 10; }
if(res == (res=5)){ res = 10; }
assert res == ??;
The answer is 5!
Apparently equality is
not commutative in Java.