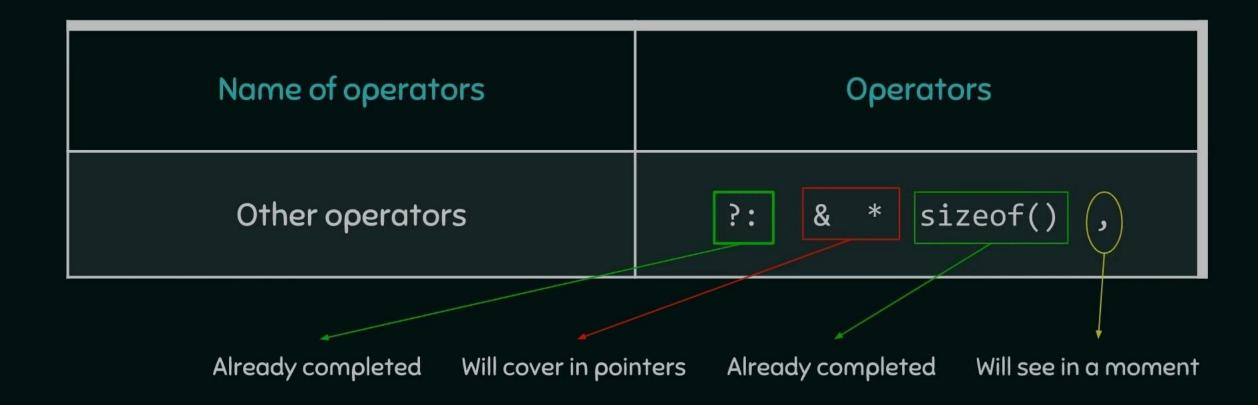
From lecture "Motivation and Introduction to operators in C"





Comma operator can be used as a "separator".

Example:

int
$$a = 3$$
, $b = 4$, $c = 8$;

=

int a = 3;
int b = 4;
int c = 8;

Multiple definitions in single line.



Comma operator can be used as an "operator".

```
int a = (3, 4, 8);
printf("%d", a);
```

Output: 8

Comma operator returns the rightmost operand in the expression and it simply evaluates the rest of the operands and finally reject them.



Comma operator returns the rightmost operand in the expression and it simply evaluates the rest of the operands and finally reject them.

Example:

int var = (printf("%s\n", "HELLO!"), 5);

printf("%d", var);

Output: HELLO!

5

It will simply not rejected. First evaluated and then rejected.

This value will be

returned to var

after evaluating

the first operand



Comma operator is having least precedence among all the operators available in C language.

Example 1:

```
int a;
a = 3, 4, 8;

printf("%d", a);

int a;
(a = 3), 4, 8;

printf("%d", a);
```

Output: 3



Comma operator is having least precedence among all the operators available in C language.

Example 2:

```
int a = 3, 4, 8;
printf("%d", a);
```

Here comma is behaving like a separator.

Output: error

Error!

int a = 3; int 4; int 8;



Comma operator is having least precedence among all the operators available in C language.

```
Example 3:
```

```
int a;
a = (3, 4, 8);
printf("%d", a);
```

OR

```
int a = (3, 4, 8);
printf("%d", a);
```



Bracket has the highest precedence than any other operator

Output: 8

HOMEWORK PROBLEM

What is the output of the following C program fragment?

```
#include<stdio.h>
int main()
    int var;
    int num;
    num = (var = 15, var + 35);
    printf("%d", num);
    return 0;
```

- a) 15
- b) 50
- c) No output
- d) Error