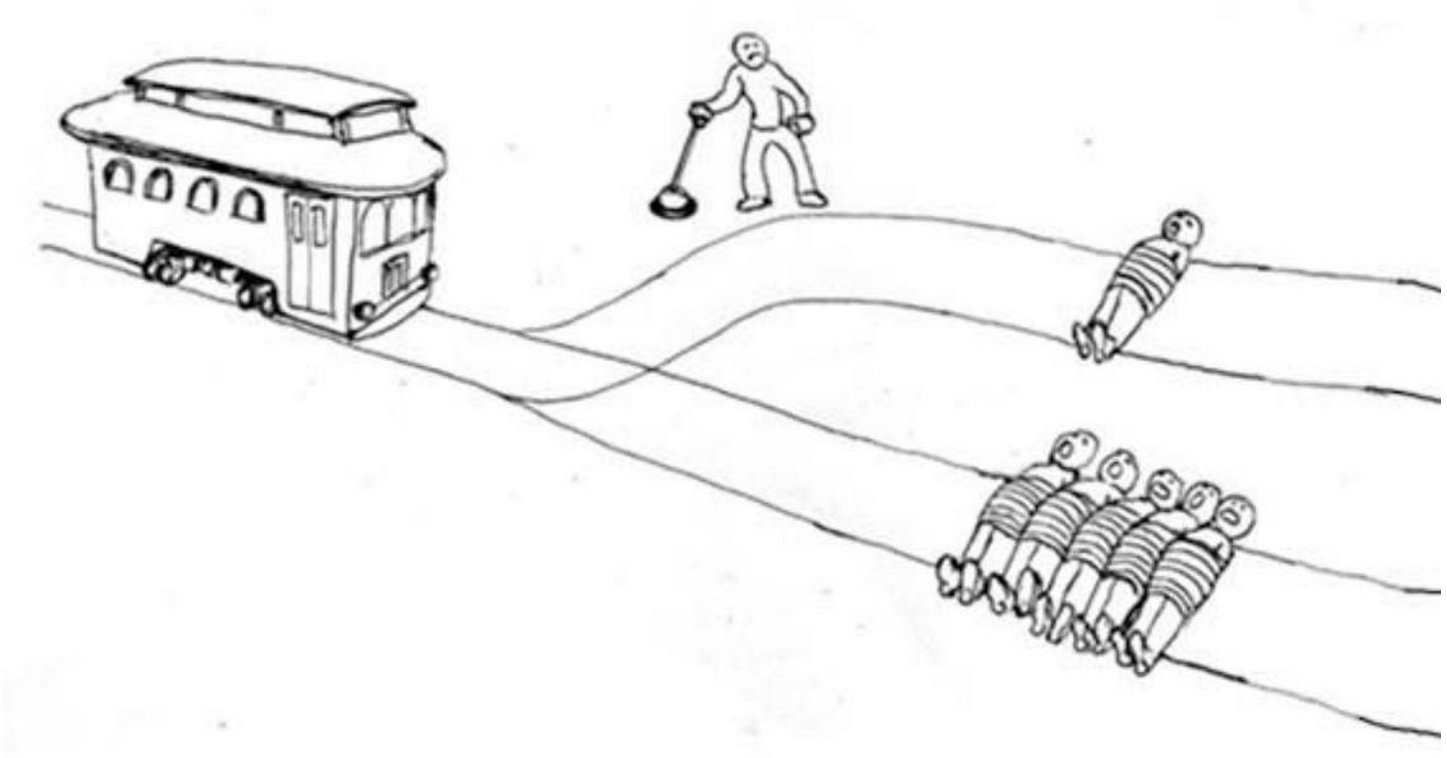
Operators and conditionals

Adapted from materials by Dr. Carrier



Basic operators

Assignment: `int x = 7;`

Basic operators

Assignment: `int x = 7;`

Arithmetic:

- Basics: `+, -, /, *`

Basic operators

Assignment: `int x = 7;`

Arithmetic:

- Basics: `+, -, /, *`
- Modulo: `%`

Basic operators

Assignment: `int x = 7;`

Arithmetic:

- Basics: `+, -, /, *`
- Modulo: % (Remainder)

Basic operators

Assignment: `int x = 7;`

Arithmetic:

- Basics: `+, -, /, *`
- Modulo: `%` (Remainder)
- Modified assignment (do op and store):
  `+=, -=, *=, /=, %=`

Basic operators

Assignment: `int x = 7;`

Arithmetic:

- Basics: `+, -, /, *`
- Modulo: % (Remainder)
- Modified assignment (do op and store):
  - `+=, -=, *=, /=, %=`
- Increment / decrement: `++, --`

Boolean operators

Comparison: ==, !=, >, <, >=, <=

Boolean operators

Comparison: ==, !=, >, <, >=, <=

Boolean operations:

- Not: !
- And: &&
- Or: ||

Boolean operators

Comparison: ==, !=, >, <, >=, <=

Boolean operations:

- Not: !
- And: &&
- Or: ||

Note: These are different than bitwise operators:

- Not (inverse): ~
- Bitwise and: &
- Bitwise or: |
- Bitwise xor: ^

A note on `bool`

C didn’t have a bool type at first

Implemented in C99

You need to `#include <stdbool.h>`

We can always just use ints!

Control statements

Control statements

```
if(expression){
...
}
else if(expression){
...
}
else {
...
}
```

Control statements

```
if(expression){
...
}
else if(expression){
...
}
else {
...
}
```

No braces needed if you have only one statement:

```
if(expression) statement;
```

Ternary operator

```
expression ? expression1 : expression2
```

If expression is true, replace with expression1
Else, replace with expression 2

Order of operations (precendence)

Precedence	Operator	Description	Associativity
1	++ --	Suffix/postfix increment and decrement	Left-to-right
	( )	Function call	
	[ ]	Array subscripting	
	.	Structure and union member access	
	->	Structure and union member access through pointer	
	(*type*){*list*}	Compound literal(C99)	
2	++ --	Prefix increment and decrement[note 1]	Right-to-left
	+ -	Unary plus and minus	
	! ~	Logical NOT and bitwise NOT	
	(*type*)	Cast	
	*	Indirection (dereference)	
	&	Address-of	
	sizeof	Size-of[note 2]	
	_Alignof	Alignment requirement(C11)	
3	* / %	Multiplication, division, and remainder	Left-to-right
4	+ -	Addition and subtraction	
5	<< >>	Bitwise left shift and right shift	
6	< <=	For relational operators < and ≤ respectively	
	> >=	For relational operators > and ≥ respectively	
7	== !=	For relational = and ≠ respectively	
8	&	Bitwise AND	
9	^	Bitwise XOR (exclusive or)	
10	|	Bitwise OR (inclusive or)	
11	&&	Logical AND	
12	||	Logical OR	
13	?:	Ternary conditional[note 3]	Right-to-left
14[note 4]	=	Simple assignment	
	+= -=	Assignment by sum and difference	
	*= /= %=	Assignment by product, quotient, and remainder	
	<<= >>=	Assignment by bitwise left shift and right shift	
	&= ^= |=	Assignment by bitwise AND, XOR, and OR	
15	,	Comma	Left-to-right