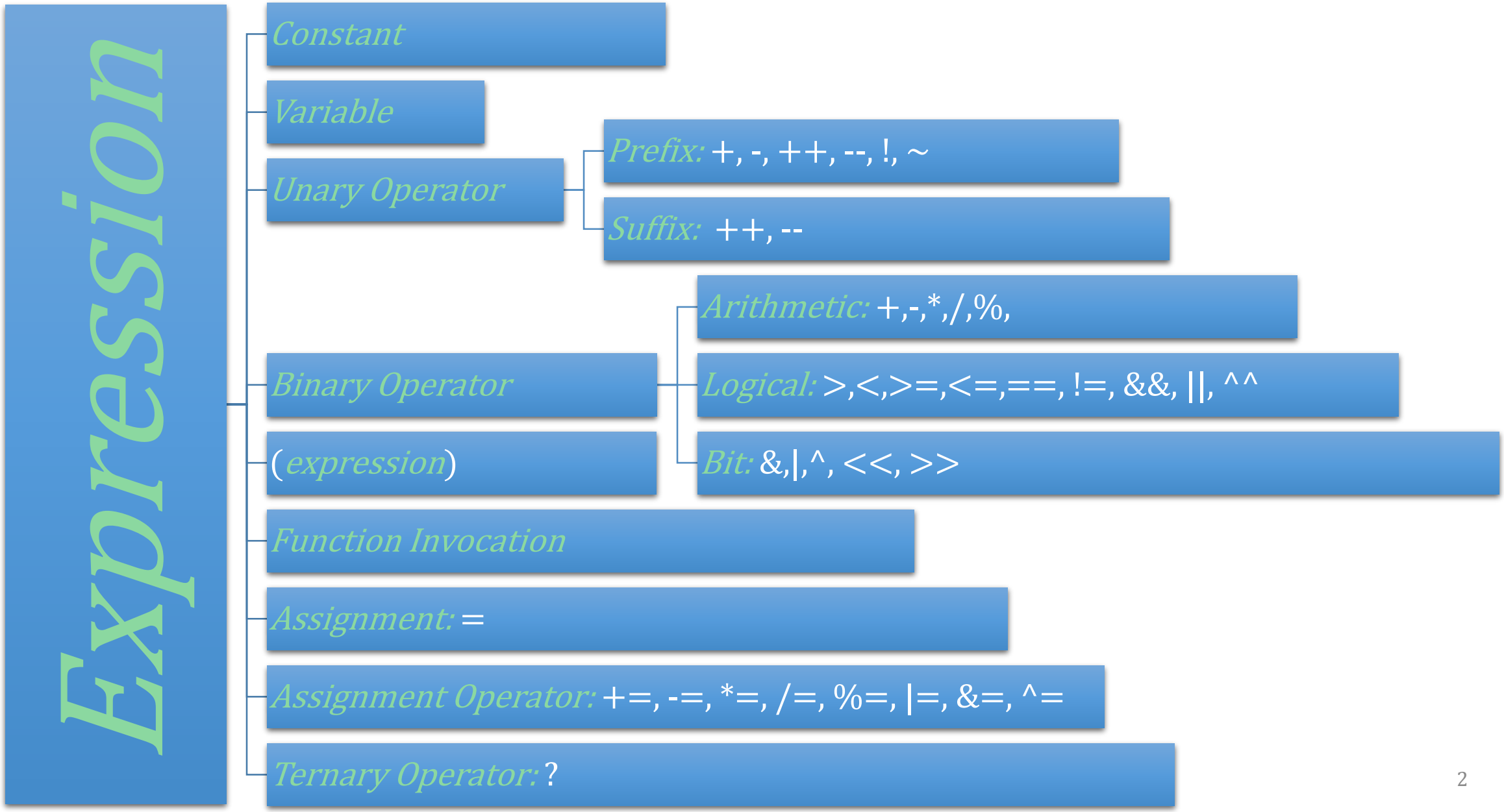
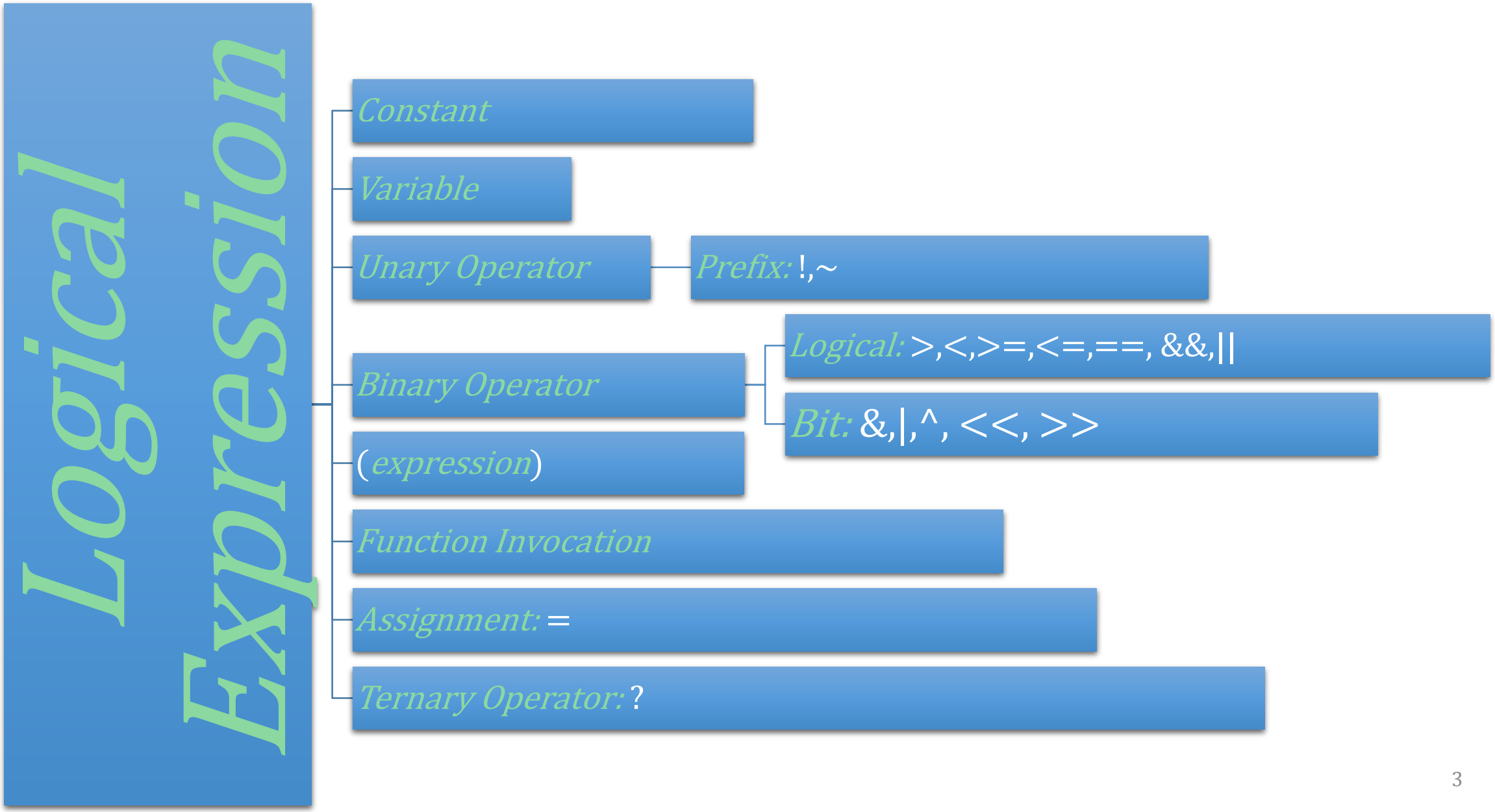
Logical and Bitwise Expressions

The truth value will set you free.


Expression
Constant
Variable
Unary Operator
Prefix: +, -, ++, --, !, ~
Suffix: ++, --
Binary Operator
Arithmetic: +,-,*,/,%,
Logical: >,<,>=,<=,==, !=, &&, ||, ^^
Bit: &,|,^, <<, >>
(expression)
Function Invocation
Assignment: =
Assignment Operator: +=, -=, *=, /=, %=, |=, &=, ^=
Ternary Operator: ?


Logical Expression
Constant
Variable
Unary Operator
Prefix: !,~
Binary Operator
Logical: >,<,>=,<=,==, &&,||
Bit: &,|,^, <<, >>
(expression)
Function Invocation
Assignment: =
Ternary Operator: ?

Logical Expressions

C Numbers and Logic

- Logic deals with two values: True and False
- Numbers have many values
- In C, zero is logically “False”
  - Expressions which evaluate to “False” have a value of 0
- In C, every number OTHER than zero is logically “True”
  - Both positive and negative numbers are “True”
  - Logically, -32,451 and 1 and 345 all are “True”
  - Expressions which evaluate to “True” have a value of 1

Logical (Boolean) Variables

Option 1

- Use an integer variable
- Use '0' for true and '1' for false

```
int firstTime=1;
myFunction(firstTime);
firstTime=0;
myFunction(firstTime);
```

Option 2

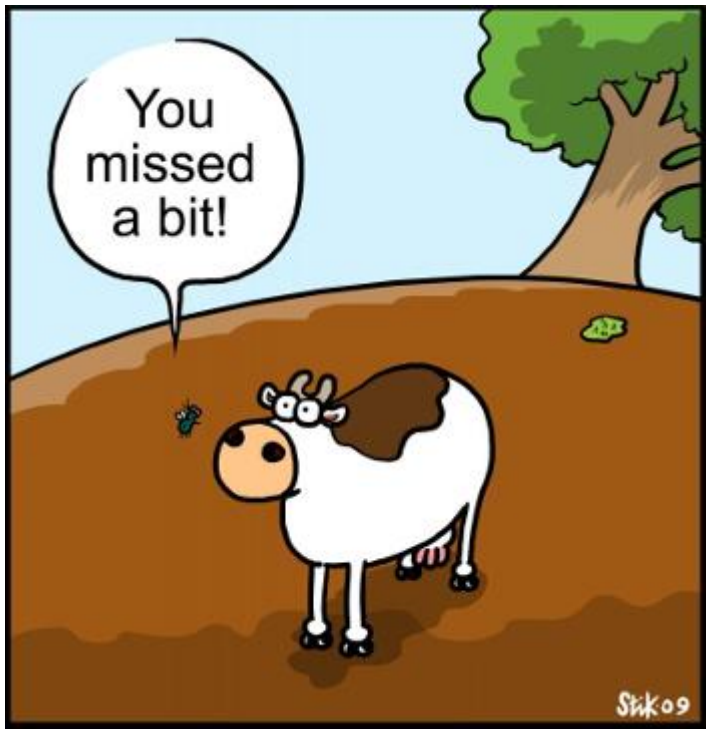
- Use library: "stdbool.h"
  - Includes data type "bool"
  - Includes values true/false

```
#include <stdbool.h>
bool firstTime=true;
myfucntion(firstTime);
firstTime=false;
myfunction(firstTime);
```

Example Logical Binary Operators

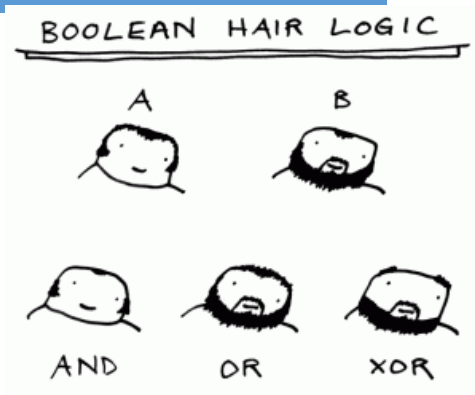
```
int x=7; int y=-3; int z=0;


z=(x>y);
z=(x>0) && (y<0);
z=( 6==x );
```



Logical Truth Tables

A	B	A || B	A&&B
F	F	F	F
F	T	T	F
T	F	T	F
T	T	T	T



Condition

- Expression interpreted as a logical value

```
int x=9;
```

```
x>10
x==6
(x-9)
x && (113/x < 12)
```

Short Circuit Evaluation

- Given: <cond1> && <cond2>
  - If <cond1> evaluates to False, <cond2> is not evaluated!
- Given <cond1> || <cond2>
  - If <cond1> evaluates to True, <cond2> is not evaluated!

Ternary operator

<condition> ? <true_expr> : <false_expr>

Evaluate condition…

if true, evaluate <true_expr>

if false, evaluate <false_expr>

```
y=x?3/x:0; /* If x=7, y=2… if x=0, y=0 */
printf("B variable is %s\n",B?"true":"false");
hamlet=question?bb:!bb;
```

BITWISE Operators

Logical vs. Bitwise

Logical

- The entire variable has a single truth value
- Logical operators work on two variables (two truth values)
- A logical operator performs one logical operation
- Use double operators… && ||

BitWise

- Each bit in the variable has a single truth value
- BitWise operators work on columns of bits
- A bitwise operators performs multiple logical operations … one for each bit position in the arguments
- Use single operators & | ^

Example

Logical

```
int x=x0000 FFFF;
int y=xFFFF 00FF;
int z = x && y;
```

```
x!=0, so x is true
y!=0, so y is true
z = true && true
z = true = 1
```

Bitwise

```
int x=x0000 FFFF;
int y=xFFFF 00FF;
int z = x & y;
```

```
x=x0000 0000 0000 0000 1111 1111 1111 1111
y=x1111 1111 1111 1111 0000 0000 1111 1111
&-----------------------------------
z=x0000 0000 0000 0000 0000 0000 1111 1111
```

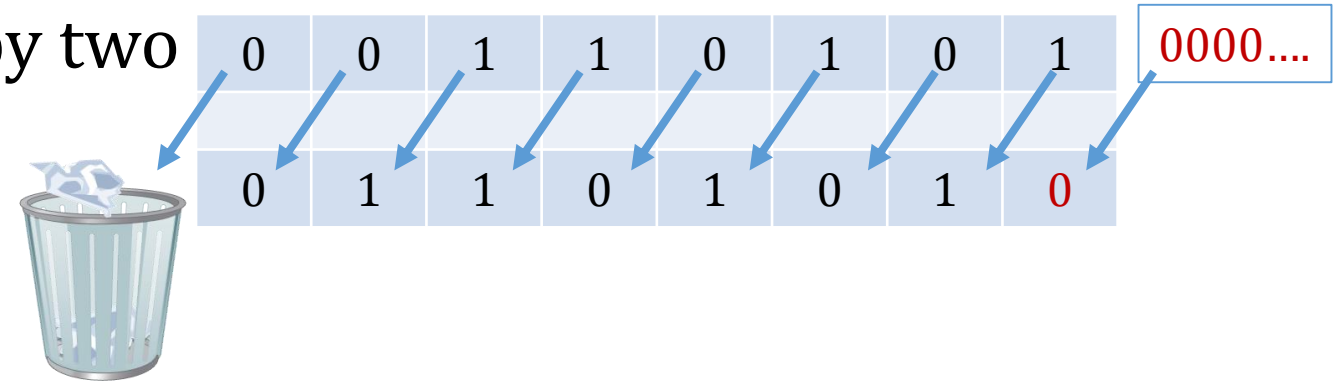
Bitwise Operators

- ~ - “invert” operator… flips each bit in the argument
  - (different from !... logical “not”… inverts the logical value of a variable)
- & - and operator… 1 if both bits are 1.
- | - or operator… 1 if either bit is 1.
- ^ - exclusive or operator… 1 if bits are different

Bit Shifting

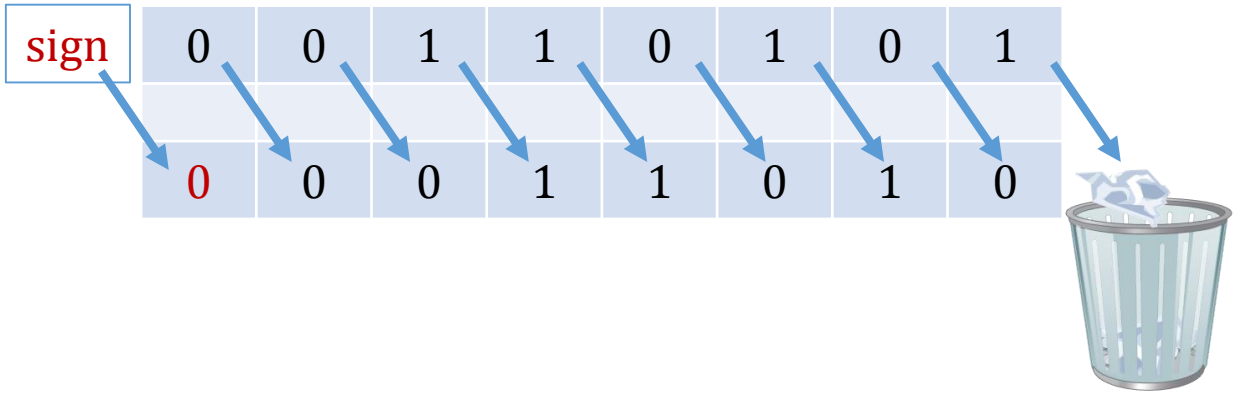
- Shift Left – Same as multiply by two

```
signed char x=53;
signed char y=x<<1;
```



- Shift Right – Same as divide by two (almost)

```
signed char x=53;
signed char y=x>>1;
```



"Bit Twiddling"

- Combination of bitwise operations and shifting
- Enables manipulation of multiple bits
- Often very "clever" (or confusing)
- Examples…

```
if ((x^y)<0) // do x and y have opposite signs?
for(c=0;v;v>>=1) c+=v&1; // count bits in v
```

Resources

- Programming in C, Chapter 3
- WikiPedia: Operators in C and C++ (https://en.wikipedia.org/wiki/Operators_in_C_and_C%2B%2B)
- Wikipedia: Short Circuit Evaluation (https://en.wikipedia.org/wiki/Short-circuit_evaluation)
- Wikipedia: Bit manipulation (https://en.wikipedia.org/wiki/Bit_manipulation)