Review Questions

- What is the value (and encoding) of **`TMin`** for a fictional 6-bit wide integer data type?
- For `unsigned char uc = 0xA1;`, what are the produced data for the cast **`(unsigned short)uc`**?
- What is the result of the following expressions?
  - **`(signed char)uc >> 2`**
  - **`(unsigned char)uc >> 3`**

Why Does Two's Complement Work?

- For all representable positive integers x, we want:

$$\begin{array}{r} \textit{bit representation of } x \\ +\ \textit{bit representation of } -x \\ \hline 0 \end{array}$$ (ignoring the carry-out bit)

  - What are the 8-bit negative encodings for the following?

```
  00000001      00000010      11000011
+ ????????    + ????????    + ????????
  00000000      00000000      00000000
```

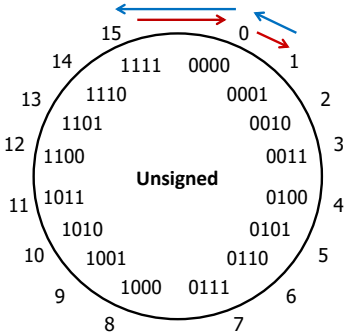

Overflow: Unsigned

- **Addition:** drop carry bit (-2^N)

```
  15        1111
+  2      + 0010
  17        10001
   1
```

- **Subtraction:** borrow ($+2^N$)

```
   1        10001
-  2      - 0010
  -1        1111
  15
```



$\pm 2^N$ because of modular arithmetic

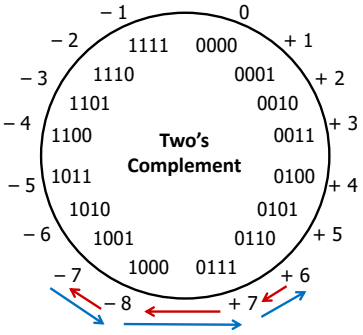
Overflow: Two's Complement

- **Addition:** (+) + (+) = (−) result?

```
   6        0110
+  3      + 0011
  -7        1001
  -7
```

- **Subtraction:** (−) + (−) = (+)?

```
  -7        1001
-  3      - 0011
  -10       0110
   6
```



For signed: overflow if operands have same sign and result's sign is different

Shift Operations (Review)

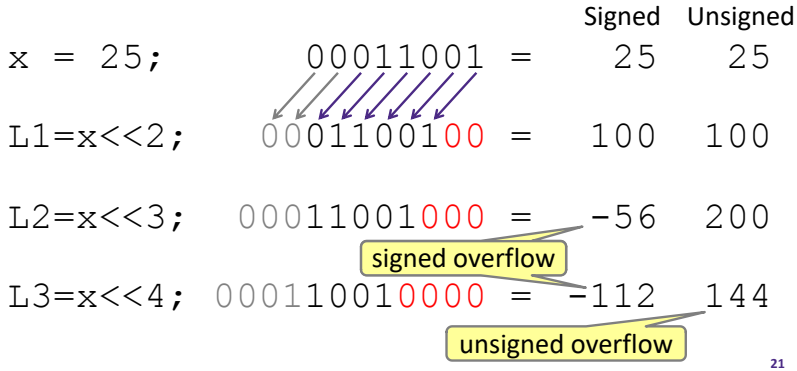
- Throw away (drop) extra bits that "fall off" the end
- Left shift (x<<n) bit vector x by n positions
  - Fill with 0's on right
- Right shift (x>>n) bit-vector x by n positions
  - Logical shift (for unsigned values)
    - Fill with 0's on left
  - Arithmetic shift (for signed values)
    - Replicate most significant bit on left (maintains sign of x)

	x	0010 0010
	x<<3	0001 0000
logical:	x>>2	0000 1000
arithmetic:	x>>2	0000 1000

	x	1010 0010
	x<<3	0001 0000
logical:	x>>2	0010 1000
arithmetic:	x>>2	1110 1000

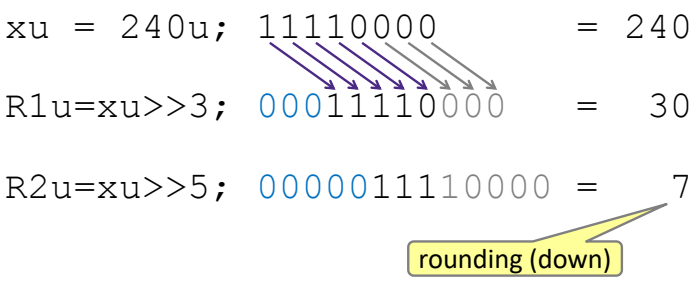
Left Shifting Arithmetic 8-bit Example

- No difference in left shift operation for unsigned and signed numbers (just manipulates bits)
  - Difference comes during interpretation: $x*2^n$?



Right Shifting Arithmetic 8-bit Examples

- **Reminder:** C operator >> does *logical* shift on unsigned values and *arithmetic* shift on signed values
  - Logical Shift: $x/2^n$?



Right Shifting Arithmetic 8-bit Examples

- **Reminder:** C operator >> does *logical* shift on unsigned values and *arithmetic* shift on signed values
  - Arithmetic Shift: $x/2^n$?