

Lecture Review

1. What happens if you store a multi-byte integer in a single-byte integer?
 - Truncation: The higher-order bytes are discarded, and only the least significant byte (the lowest 8 bits, equivalent to value & 0xFF) is kept.
 - Value & Sign Alteration: If the original value exceeds what an 8-bit integer can represent, data is lost. In signed types, the most significant remaining bit (bit 7) is interpreted as the sign bit, which can turn a positive number into a negative number.
2. Is a 32-bit integer enough to store a number in the trillions?
 - One trillion is $10^{12} = 1,000,000,000,000$. A signed 32-bit integer (`int32_t`) has a maximum value of $2^{31} - 1 = 2,147,483,647$ (~2.15 billion). An unsigned 32-bit integer (`uint32_t`) has a maximum value of $2^{32} - 1 = 4,294,967,295$ (~4.29 billion). Since $4.29 \times 10^9 < 10^{12}$, the answer is no.
3. If I have an integer made of 11 bytes, what integer types can I store it in?
 - An 11-byte integer requires $11 \times 8 = 88$ bits. For Standard C integer primitives, they max out at 8 bytes / 64 bits (`long long`, `int64_t`, or `uint64_t`), which all fits in this integer. An array of bytes (e.g., `uint8_t[11]`) or a custom struct of integers may also fit.