Something More Fun

- I write integers 1, 2, \ldots, 49, 50 on board.
- You repeat until one number left:
  - Select any two distinct \( a, b \) on board.
  - (If same number written twice, can choose for \( a, b \)).
  - Erase \( a, b \) from board.
  - Write \( |a - b| \) on board.

- I wonder:
  - Can you get an even number remaining?

\[
\sum_{i=1}^{50} i = \frac{25(51)}{2}
\]
2 Something More Fun

- I write integers 1, 2, \ldots 49, 50 on board
- You repeat until one number left:
  - Select any two distinct $a, b$ on board
    (if same number written twice, can choose for $a, b$)
  - Erase $a, b$ from board.
  - Write $|a - b|$ on board

- I wonder:
  - Can you get a negative number remaining?
    \[ \text{no} \]

2 Something More Fun

- I write integers 1, 2, \ldots 49, 50 on board
- You repeat until one number left:
  - Select any two distinct $a, b$ on board
    (if same number written twice, can choose for $a, b$)
  - Erase $a, b$ from board.
  - Write $|a - b|$ on board

- I wonder:
  - What number(s) could be left when you are done?
    \[ \text{Invariant: only } [0, 50] \text{ on board} \]
Reminder: recursion exists

```c
unsigned factorial(unsigned n);

unsigned factorial(unsigned n)
{
    if ( n <= 1 )
        return 1;
    // else:
    return n * factorial(n-1);
}
```
5. Programs to solve puzzles?

- You want to buy \( n \geq 12 \) cans of soda
  - Packs of soda are 4 or 5 cans each
  - How many of each pack to buy?
- You have a \( 2^n \times 2^n \) chessboard
  - One square is removed
  - You have a lot of 3-square L shaped dominoes
  - How can you cover the squares with dominoes?
- You have an \( n \times n \) chessboard
  - Please \( n \) queens so no two threaten one another

6. How would you solve this puzzle?

- Assign a unique digit to each letter
- Make each equation true.

\[
\begin{array}{c}
P O T \\
+ D A N \\
\hline
B I B
\end{array}
\]

\[
\begin{array}{c}
2 \times 1 \times 3 \\
\hline
2 \times 6 \times 5
\end{array}
\]
Buying soda

- You want exactly \( n \geq 12 \) cans of soda
- Store has enough 4-packs and 5-packs
- Packs cannot be divided

Writing the function

```c
void buySoda(unsigned n, unsigned &num4Packs, unsigned &num5Packs)
{
    if (12 == n)
    {
        num4Packs = 3;
        num5Packs = 0;
        return;
    }
    // Not written but should: 13, 14, 15
    else
    {
        buySoda(n-4);
        num4Packs += 2;
        return;
    }
}
```
Tiling a $2^n \times 2^n$ chessboard

Tiling a $2^n \times 2^n$ chessboard