### Lecture 2: Recursion I

Soda Cans, TileChess

#### 3. Something More Fun

- I write integers 1, 2, … 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

| a | b | |a−b| |
|---|---|---|---|
| odd | even | odd | even |
| even | odd | even | odd |

**Invariant**

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

**Parity**

\[
\sum_{i=1}^{50} \frac{25 - (5i)}{2}
\]

is odd
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?
    
    \textbf{No}

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

invariant: all \# on board: $c \in [0, 50]$

- I wonder:
  - What number(s) could be left when you are done?
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.

\[
\begin{align*}
\text{DOT} & + \text{PAN} \\
\phantom{\text{DOT}} & + \phantom{\text{PAN}} \\
\hline
\text{B1B} & 
\end{align*}
\]
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;
    }
    else if (13 == n)
    {
        num4Packs = 2;
        num5Packs = 1;
        return;
    }
    // Todo: 2 more base (14, 15)
    buySoda(n - 4);
    num4Packs++;
    return;
}
```
Tiling a $2^n \times 2^n$ chessboard