1 Random Math Puzzle!

1.1 Problem

I got this problem from a friend. You could easily have a computer solve this problem – but there's also a very elegant mathematical solution - have fun! ;)

On a table are nine face-up cards. Each card displays a different number from the list: 2, 4, 8, 16, 32, 64, 128, 256, 512. Alice and Bob alternately take turns. Each picks a single card on a turn, and Alice goes first. Once a card is chosen it cannot be selected again. The game ends when someone wins or all the cards have been selected. The winner is the first person to collect a set of 3 cards with a product of 32,768. If neither player does this, the game ends in a draw. If both play optimally, does either player have a winning strategy? Or does the game always end in a draw? How should you play this game?

The solution starts on the next page. The computer solution was written by me, who, as a computer geek, of course, defaulted to a minimax algorithm. The mathematical solution was provided by my friend who sent me this problem, who's a math major at Princeton University.

1.2 Computer Solution

Here's the minimax code to solve this, written in Python. I like this solution since it's more generalizable to arbitrary numbers than the mathematical one. Notice that all the numbers given are powers of 2, so the core idea to win is to obtain a subset of three in $[1, 9]$ such that the three numbers add to 15.

```
import copy

def filt(L, p):
    result = []
    for x in L:
        if p(x):
            result.append(x)
    return result

def moveA(A, B, L, target):
    # no moves left
    if len(L) == 0: return "tie"
    # try to win now
    for i in A:
        for j in A:
            if i != j and (target - i - j) in L:
                return "A wins"
    # try to keep B from winning
    for i in B:
        for j in B:
            if i != j and (target - i - j) in L:
                newA = copy.copy(A)
                newA.append(target-i-j)
                newL = filt(L, lambda x: x != (target - i - j))
                return moveB(newA, B, newL, target)
    # try to win sometime later
    for i in L:
        newA = copy.copy(A)
        newA.append(i)
        newL = filt(L, lambda x: x != i)
        result = moveB(newA, B, newL, target)
        if result == "A wins": return result
    # try to not lose
    for i in L:
        newA = copy.copy(A)
        newA.append(i)
        newL = filt(L, lambda x: x != i)
        result = moveB(newA, B, newL, target)
        if result != "B wins": return result
    # if your only choice is to lose
```

```
    i = L[0]
    newA = copy.copy(A)
    newA.append(i)
    newL = filt(L, lambda x: x != i)
    return moveB(newA, B, newL, target)

def moveB(A, B, L, target):
    # no moves left
    if len(L) == 0: return "tie"
    # try to win now
    for i in B:
        for j in B:
            if i != j and (target - i - j) in L:
                return "B wins"
    # try to keep A from winning
    for i in A:
        for j in A:
            if i != j and (target - i - j) in L:
                newB = copy.copy(B)
                newB.append(target-i-j)
                newL = filt(L, lambda x: x != (target - i - j))
                return moveA(A, newB, newL, target)
    # try to win sometime later
    for i in L:
        newB = copy.copy(B)
        newB.append(i)
        newL = filt(L, lambda x: x != i)
        result = moveA(A, newB, newL, target)
        if result == "B wins": return result
    # try to not lose
    for i in L:
        newB = copy.copy(B)
        newB.append(i)
        newL = filt(L, lambda x: x != i)
        result = moveA(A, newB, newL, target)
        if result != "A wins": return result
    # if your only choice is to lose
    i = L[0]
    newB = copy.copy(B)
    newB.append(i)
    newL = filt(L, lambda x: x != i)
    return moveA(A, newB, newL, target)

print("Answer:", moveA([], [], list(range(1, 9 + 1)), 15))
```

1.3 Mathematical Solution

Notice that all the numbers given are powers of 2, so the core idea to win is to obtain a subset of three in $[1, 9]$ such that the three numbers add to 15. So writing out these numbers on a 3 by 3 grid, we have:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Note that any three-in-a-row that goes through the center 5 yields the solution. This, plus the triples $(2, 4, 9), (3, 4, 8), (1, 6, 8), (2, 6, 7)$ provide all solutions. Hence, we can devise the following by keeping 5 at the center, but moving around the edge entries to yield the following 3 by 3 grid.

$$\begin{pmatrix} 4 & 3 & 8 \\ 9 & 5 & 1 \\ 2 & 7 & 6 \end{pmatrix}$$

Now, we see that winning the game is the exact same as winning tic-tac-toe on the above board, i.e. this game is isomorphic to tic-tac-toe. Since tic-tac-toe can always end in a tie if both players optimize, the answer is a tie.