# Assignment # 2

**(290 points)**

*NOTE: Take the print out of the assignment and read the Questions once and keep working on this while walking, playing, even during dreaming (I call it, chaltay phirtay soochna: You should make your habit of this now; as a lot of DSA assignments related works will involve thinking and you should develop this brain habit of working in the background, coding the programs will be hardly of 10-20% of the time).*

## Problem 1 (10 points)

- Five pirates on an island have one hundred gold coins to split among themselves. They divide the loot as follows: The senior pirate proposes a division, and everyone votes on it. Provided at least half the pirates vote for the proposal, they split the coins that way. If not, they kill the senior pirate and start over. The most senior (surviving) pirate proposes his own division plan, and they vote by the same rules and either divide the loot or kill the senior pirate, as the case may be. The process continues until one plan is accepted.
  Now Suppose you are the most senior pirate ;) What division do you propose? (The pirates are all extremely logical and greedy, and all want to live).

- (BONUS) What will be your generalized strategy if there are N people in total. **(10 points)**

## <u>Mathematical Induction Continues</u>

## Problem 3 (10 points)

Assume we have n people, say $P_1, \ldots, P_n$ at a party. Furthermore, for every two persons, they either shake hands during the party, or they do not (well duh!). Prove that one can always, regardless of whom shook hands with whom, arrange these n people into two lines such that the following holds. In the first line, each person has shaken hands with the person immediately behind them in the line. In the second line, each person has not shaken hands with the person immediately behind them in the line. Note that one of these lines could be empty, i.e., all the n people are in only one of these lines.

## Problem 4 (15 points)

- Prove that $F_{3n}$ is even for all $n \geq 1$, where $F_0 = 0$, $F_1 = 1$, and $F_n = F_{n-1} + F_{n-2}$ for all $n > 1$.
- Here's an inductive proof that $n(n + 1)$ is odd: we show that $n(n + 1)$ is odd, assuming that $(n - 1)(n)$ is odd. Note that $n(n + 1) = (n - 1)(n) + 2n$. Since $2n$ is even, and by inductive hypothesis, $(n - 1)(n)$ is odd, and since the sum of an even and an odd number is odd, we conclude that $n(n+1)$ is odd. Clearly, something is wrong in the above proof, since for $n = 10$, we have $n(n + 1) = 110$, which is even. What exactly is wrong?

## Problem 5 (30 points)

- Use induction to prove that the sum of the (base-10) digits in a number x is divisible by 9 **if and only if** x is divisible by 9. (if and only if means you have to show in both directions).
- We can place a square using match sticks. Prove that to make n squares in a row you need exactly 3n + 1 sticks.
- Remember we played the game of Cookies (where you have a row of cookies: and there are two players who eat some number of cookies at max 3 at a time, whoever eats the last cookie, WINS). Prove USING MATHEMATICAL INDUCTION THAT if there are 4M(multiple of 4) number of cookies then Player 2 always has a winning strategy OTHERWISE Player 1 always has a winning strategy.

## Problem 7 (15 points)

- Prove that the Weak Principle of Mathematical Induction implies the Strong Principle of Mathematical Induction.
- Prove that the Strong Principle of Mathematical Induction implies the Weak Principle of Mathematical Induction.
- Show how RECURSION and Mathematical Induction are related to each other (one page should be suffice).

## Challenge Set 2 (Recursion related to problems) (70 points)

1. Write the recursive and iterative code for Fibonacci Number computation.
    - Analyze why Iteration is working so fast as compare to recursive implementation of Fibonacci Numbers.
    - Use **Memorization Technique** to make the recursive algorithm fast.
2. Write the recursive and iterative code for Computing the TriSum sequence:
   1, 2, 3, 6, 11, 20, 37, ........
    - Write the recursive mathematical formulation.
    - Write the recursive code for the Sequence generator
    - Analyze what will be its time complexity (the approximate number of time the recursive call will be called.
    - Give the Dynamic Programming solution to avoid the recalculation of the same TriSum number again and again.

3. Write a recursive function for computing the factorial of a number N.
4. **Write a recursive function using the recursive definition of *Permutation, Combination (Formula).***
5. **Write a recursive Linear Search function.**
6. **Write the Recursive Binary Search and test it on a huge Data.**
7. Write a Recursive function to compute POWER(X, Y, M) compute $X^Y$ % M ($X^Y$ modulo M).

8. Write a Program which wants to do multiplication of AxB imagine all A and B are n bit strings and there is module available ADD(X,Y) and you want to write this MULT(X,Y) using ADD(X,Y) How you will going to write this module. Write the module such that It takes minimum number of steps, obviously Calling ADD Y times is very bad idea and unacceptable.

## Advanced Problems (RECURSION)      (140 points, each of 20 points)

9. Write a Recursive Pseudocode to find the Determinant of a NxN Matrix and Drive its Time Complexity(Big O is fine with me)
10. (Bonus) Find a problem which is doable using Recursion but not possible with Iterations. Of course other than Permutation question.
11. **Permuting a String.**
12. **FAST POWERING** again POWER(X, Y, M) compute $X^Y$ % M ($X^Y$ modulo M) with ONLY LOG(Y) multiplications allowed.
13. **8 Queen Problem**
    - **Prove 9 Queen within 8x8 box cannot be placed such that every Queen is not in threat by the other.**
14. **Tower Of Hanoi**
15. **Extended Prison-Break:** Given a two dimensional world containing containing many prisoners, A TWO DIMENSIONAL ARRAY containing zeroes(free space), 1's (walls' bricks) and 2's (prisoner position) and 3's (location of escape gate). Assume that wall could be multi layered. Design an algorithm which reads a file (a two dimensional array of Two dimensional world) and tell about each prisoner whether he could free. Assuming that a prisoner can only move vertically and horizontally through ONLY AND ONLY free space (zeroes).