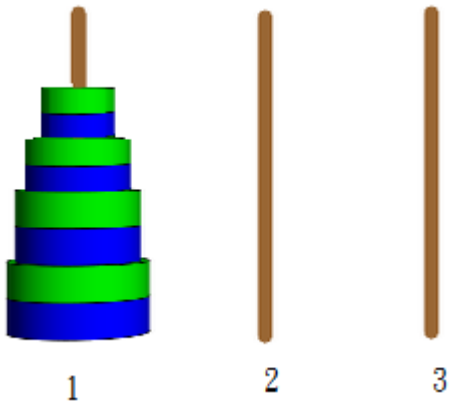
1. Rewrite linear search using recursion.
2. Rewrite bubble sort using recursion.
3. Rewrite selection sort using recursion.
4. Mr. Smith has several sums of money that he would like to equally deposit into two bank accounts. Given the sums of money as input, you are to write a program to help Mr. Smith determine if his goal is achievable. If it is achievable, the program must show Mr. Smith into which account each sum of money goes. If it is not achievable, the program should state the unfortunate conclusion. Input consists of two lines, the first of which is an integer N showing how many sums of money there are, followed by N integers on the second line.

5. Write a recursive function to move n disks from peg 1 to peg2, using peg3 as an intermediate peg. These disks come in pairs, each pair being of the same radius but different colors as shown below. Moving rules: no disk of a larger radius can be placed on top of a smaller disk during the process, and no blue disk can be placed on top of a green disk of the same radius.




6. **THE STONE GAME**

**Game description:** This program deals with a game, the Stone Game, which proceeds as follows: initially, there are three piles of stones, each pile with an arbitrary number of stones greater than two. Two players take turn to remove either one or two stones from a pile. Whoever removes the last stone wins the game.

**Program description:** The program randomly generates the number of stones for each pile, and for the sake of finishing the game in a reasonable amount of time, each pile shall not contain more than 12 stones. The computer will play against a human opponent and will let the user decide who is to move first. Once the game starts, the computer will try its best to win the game; that is, when there is a winning move, the computer must make this move.During the course of the game, the program should somehow display a graphic representation of the piles of stones. The computer will prompt the user for input when it is the user's move, and it will display a message, such as "I am thinking", when it is looking for a best move. At the end of the game, the program must announce the winner. There is no possibility of a draw for this game.

**Program design:** A state of the game is characterized by the numbers of stones in the piles. A state is said to be a forced-win state if one of its sub-states leads to a loss. In other words, a player is in a forced-win state if there exists a move he can make such that no matter what his opponent does afterwards, his opponent is bound to lose. A state is a forced-loss state if all its sub-states are forced-win states. The computer should always make a move that produces a forced-loss state for the human player whenever possible. If there is no winning move, then the computer will choose the move that removes the least number of stones from the piles. The program terminates when the last stone is removed and the winner is declared. The following is an example of the sequence of moves made during the game:

2 1 1 forced-loss → 0 1 1 forced-win → 0 0 1 forced-loss → 0 0 0 win and game over

0 1 0 forced-loss → 0 0 0 win and game over

Employ recursion to compute the win/loss status of a game state and to help decide what move to make by the computer.