(1) Let there be three integer variables count, K, and L. Let them be set to zero. Then there's a for loop with integer integer i set to zero. The integer i is incremented by one. In the for loop There are five comparison operators in the for loop. The first is an if statement. IF the array at index i equals array length - 5 - K, then increment the count by one. The next if statement asks if the count is greater than 1, then print array at index i, then increase L by 1. The third ~~is~~ if the integer i equals array length -1, then set i = 0, increase K by 1, and set count equal to zero. The fourth if statement asks if ~~K ==~~ K equals array length minus five, then ~~equals~~ set i to array length. The last if statement asks if L equals five. IF true then set i to array length.

(2) I declare an integer variable sum to zero. Then I make a for loop with integer i set to zero. If the integer is less than n, increment i by one. In the for loop, the sum is incremented ~~till~~ at the List in the index at i. Then the sum equals the sum multiplied by 2.

(3) We take two Queues, A and B. The A stack pushes data into it, using the push(), which is an O(1) operation. The pop function removes data from A and then it adds it to the B stack, which is an O(n) operation.

(4.) We can use two stacks A and B. Inserting just requires pushing in elements into A, which is an O(1). Then pushing the smallest element into B. Popping the elements requires just removing from the B stack, which is an O(1) operation. Then find min is also O(1), since it just returns the element in B.

(5.) An integer is contained in 4 bytes. A stack contains at most 1 megabyte on most commercial computers or 8 megabytes on Linux computers. The numbers dealing with the Fibonacchi sequence are ~~exponent~~ millions of integers, when N = 50. So when we multiply 4 by the ~~exponential~~ integer ~~number~~ that's a million, we see that we have surpassed the ~~max~~imum space ~~needed to~~ alloted to a stack.