Your assignment is to write a MIPS program that inputs a number \( n \) from the user, and then outputs the 0\(^{th} \) to \( n \)\(^{th} \) Leonardo numbers*. Leonardo numbers are similar to Fibonacci numbers, and are defined recursively such that:

\[
L_0 = 1 \\
L_1 = 1 \\
L_i = L_{i-1} + L_{i-2} + 1
\]

Thus:

- If the user enters 0, your program must print “1”.
- If the user enters 1, your program must print “1, 1”.
- If the user enters 5, you print “1, 3, 5, 9, 15”.
- And so on...

Note that you don’t actually need to use recursion. (In fact, you probably don’t know how to recurse yet.) Sketch out how you’d solve this with a simple loop, like you were programming this in Java or C, and then implement it in assembly. Also, do not use an array. Simply print out each number as you calculate it.

The final program you turn in should be called `leonardo.asm`.

*Leonardo numbers are primarily used by a sorting algorithm called smoothsort, written by Edsger Dijkstra in 1981. It is a variation of heapsort, but its time complexity approaches \( O(n) \) on mostly-sorted data.