1. Mini-project regarding the Fibonacci function:
   1.1 practice converting loops into recursive functions.
   1.2 proving correctness of algorithms.

2. In-class discussion of the solution.
fun fib :: "nat ⇒ nat"  where
  "fib 0 = 0"  |
  "fib (Suc 0) = 1"  |
  "fib (Suc(Suc x)) = fib x + fib (Suc x)"
Iterative Fibonacci Algorithm

- The fib function is inefficient because it redundantly computes the same fibonacci number over and over.
- The following iterative algorithm computes Fibonacci numbers in linear time (textbook page 317).

```
procedure iterative_fibonacci(n)
    if n = 0 then
        y := 0
    else
        x := 0
        y := 1
        for i := 1 to n - 1
            z := x + y
            x := y
            y := z
        return y
```
1. Implement a recursive version of the iterative fibonacci algorithm using accumulator passing style.

2. Prove that your recursive function produces the same output as fib.