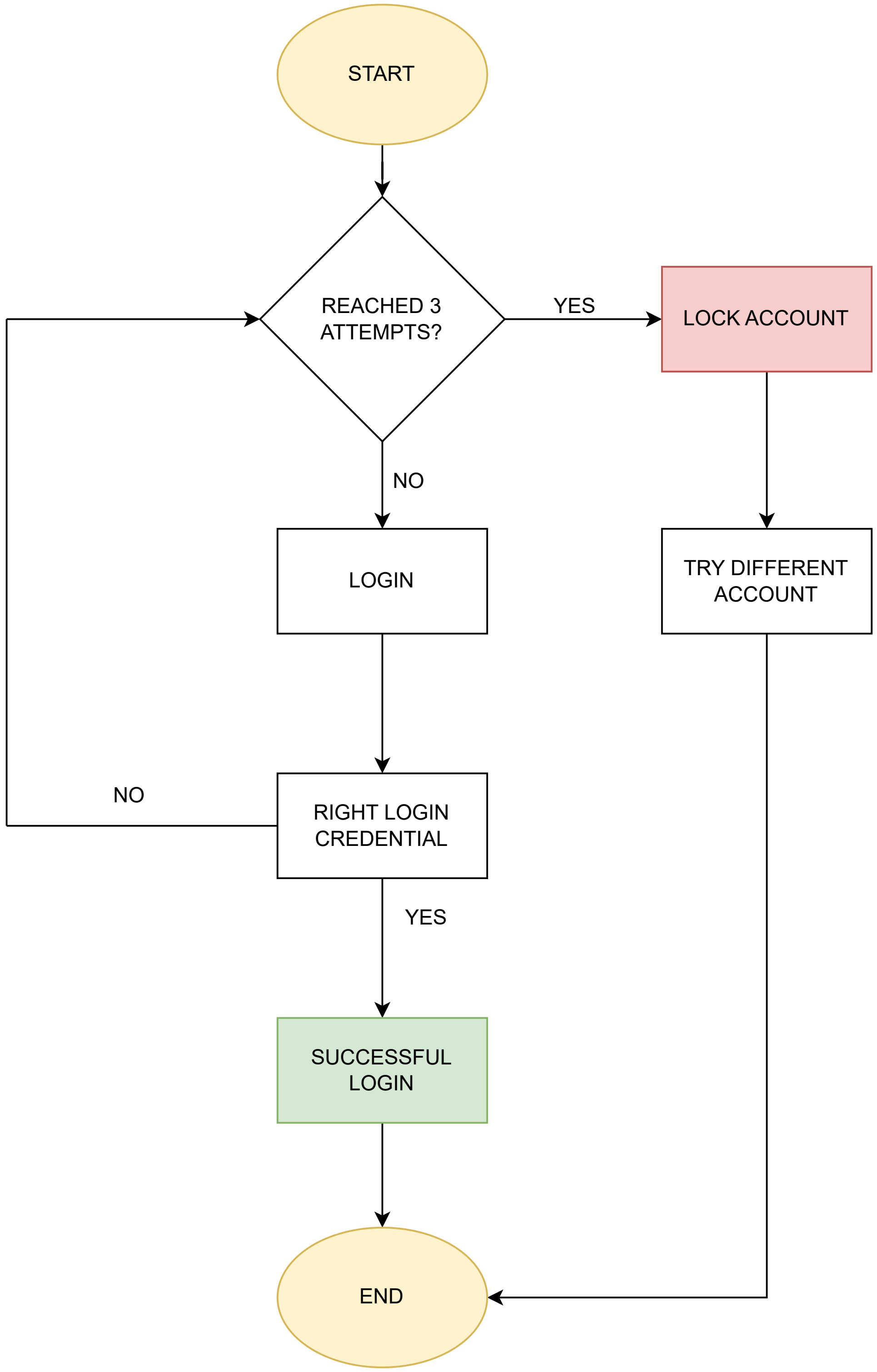
```
BEGIN

input a number
divided the number by 2

if remainder = 0
output = number*number

else
output = number*number*number

END
```


START
REACHED 3 ATTEMPTS?
YES
LOCK ACCOUNT
NO
LOGIN
TRY DIFFERENT ACCOUNT
NO
RIGHT LOGIN CREDENTIAL
YES
SUCCESSFUL LOGIN
END

```
BEGIN

Take input = n which is a positive integer
Initialize factorial = 1
For i = 1 to n:
    factorial = factorial * i
Output factorial

END
```

Modules are self-contained units. A module interface defines an element (variable or function) Modular design benefits us in multiple ways by reusing in different projects and isolating functionality which simplifies debugging and updates moreover we can add more features easily.

```
BEGIN

Input n as an integer
if n < 0
    output wrong input
else if  n == 0
    output 0
else if n == 1
     output 1
else
    Output =  (Fibonacci_Series(n - 1) +
    Fibonacci_Series(n - 2))

END
```

Modules are self-contained units. A module interface defines an element (variable or function) Modular design benefits us in multiple ways by reusing in different projects and isolating functionality which simplifies debugging and updates moreover we can add more features easily.