1

Alexandra Yaunilovitch
Abigail Rechkin

Homework 1

We found that only Algorithm F0 works and Algorithms F1, F2, and F3 do not work.

Since we are using 3 variables i, j, k to represent consecutive Fibonacci numbers, m must be decremented by 3 as is done in Algorithm F0. However, F1 decrements m by 2 instead. For an input of n=3 in F1, an output of 2 is expected while the actual output is 3. Algorithm F2 will return an output of 1 (k=i+j) because the values of i and j are always i=0 and j=1. Thus, for an input of n=3, F2 returns an output of 1 instead of 2. Algorithm F3 does not have a reliable loop condition (while m!=3) because for any input n that is not divisible by 3 or for input n = 3 , the program will enter an infinite loop. For an input of n=2, the program does not terminate.

As stated earlier, Algorithm F0 is valid. So, we will provide a program correctness proof on the following page.

Algorithm F0 Program Correctness

```
1  procedure F0(n)
2     i ← 0, j ← 1, k ← 1, m ← n
3     while ( m >= 3) do
4        m ← m-3
5        i ← j + k
6        j ← i + k
7        k ← i + j
8     if m=0 then
9        return i
10    else
11       if m=1 then
12          return j
13       else
14          return k
```

The expected input n is an integer that is greater than or equal to 2. The expected output is F_n, which is the n^{th} Fibonacci number.

Our proposed loop invariants are:

1) $m \geq 0$ and is an integer
2) $i = F_{n-m}$
3) $j = F_{n-m+1}$
4) $k = i + j$

Proof by induction:

Let m_t, i_t, j_t, k_t be the values of $m, i, j, \& \ k$ right before line 3 (the while test) is executed for t^{th} iteration. We prove this by induction on t.

<u>Base case</u>: $t=1$

On line 2, $i_1 = 0$, $j_1 = 1$, $k_1 = 1$, and $m_1 = n$. Since the $n \geq 2$ and $n \in \mathbb{Z}$, we conclude that $m_1 \geq 0$ and is an integer.

$i_1 = 0 = F_{n-m} = F_{n-n} = F_0$ which holds.

$j_1 = 1 = F_{n-m+1} = F_{n-n+1} = F_1$ which holds.

$k_1 = i_1 + j_1 = 0 + 1 = 1$ which holds.

Inductive step:

Inductive hypothesis: Assume for $t \geq 1$,

1) $m_t \geq 0$ and is an integer
2) $i_t = F_{n-m_t}$
3) $j_t = F_{n-m_t+1}$
4) $k_t = i_t + j_t$

Consider the $(t+1)^{st}$ test of the while condition.
We need to show:

1) $m_{t+1} \geq 0$ and is an integer
2) $i_{t+1} = F_{n-m_{t+1}}$
3) $j_{t+1} = F_{n-m_{t+1}+1}$
4) $k_{t+1} = i_{t+1} + j_{t+1}$

Since line 3 is about to be executed for the $(t+1)^{st}$ time, its condition was true during the t^{th} execution. We know $m_t \geq 3$ and on line 4, $m_{t+1} = m_t - 3$ which can be rearranged as $m_{t+1} + 3 = m_t$. Substituting that for m_t, we have $m_{t+1} + 3 \geq 3$ which is $m_{t+1} \geq 0$. From our IH, we know that m_t is an integer and thus $m_{t+1} = m_t - 3$ is also an integer by closure. So, loop invariant 1 holds.

On line 5, $i_{t+1} = j_t + k_t$ and from IH, we can substitute into this expression $j_t = F_{n-m_t+1}$ and $k_t = i_t + j_t$. So, $i_{t+1} = F_{n-m_t+1} + i_t + j_t$. Again, through substitution from IH,
$i_{t+1} = F_{n-m_t+1} + F_{n-m_t} + F_{n-m_t+1}$. From line 4, we know $m_{t+1} = m_t - 3$ which is $m_{t+1} + 3 = m_t$.

$i_{t+1} = F_{n-(m_{t+1}+3)+1} + F_{n-(m_{t+1}+3)} + F_{n-(m_{t+1}+3)+1}$ by substitution

$i_{t+1} = F_{n-m_{t+1}-3+1} + F_{n-m_{t+1}-3} + F_{n-m_{t+1}-3+1}$ simplification

$i_{t+1} = \underbrace{F_{n-m_{t+1}-2} + F_{n-m_{t+1}-3}} + F_{n-m_{t+1}-2}$

$i_{t+1} = F_{n-m_{t+1}-1} + F_{n-m_{t+1}-2}$ by definition of Fibonacci

$i_{t+1} = F_{n-m_{t+1}}$ by definition of Fibonacci

So, invariant 2 holds.

On line 6, $j_{t+1} = i_{t+1} + k_t$. Based on the IH and the proven loop invariant $i_{t+1} = F_{n-m_{t+1}}$, we can substitute for i_{t+1} and k_t.

$j_{t+1} = F_{n-m_{t+1}} + i_t + j_t$

$j_{t+1} = F_{n-m_{t+1}} + F_{n-m_t} + F_{n-m_t+1}$ from IH

$j_{t+1} = F_{n-m_{t+1}} + F_{n-(m_{t+1}+3)} + F_{n-(m_{t+1}+3)+1}$ from $m_{t+1}+3 = m_t$

$j_{t+1} = F_{n-m_{t+1}} + \underbrace{F_{n-m_{t+1}-3} + F_{n-m_{t+1}-2}}$ simplify

$j_{t+1} = F_{n-m_{t+1}} + F_{n-m_{t+1}-1}$ def. of Fibonacci

$j_{t+1} = F_{n-m_{t+1}+1}$ def. of Fibonacci

So, invariant 3 holds.

On line 5, $i_{t+1} = j_t + k_t$. On line 6, $j_{t+1} = i_{t+1} + k_t$. Finally, on line 7, $k_{t+1} = i_{t+1} + j_{t+1}$. Thus, invariant 4 holds. Therefore, all 4 loop invariant hold □

Soundness:

If and when the loop terminates, based on invariant 1, we know $m \geq 0$ and is an integer. Furthermore, based on the loop condition, exiting the loop requires that $m < 3$. So, $m = 0$, $m = 1$, or $m = 2$.

Case $m=0$:

When $m=0$ on line 8, on line 9, we return i.
$i = F_{n-m}$ from invariant 2. Thus, substituting $m=0$,
$i = F_{n-0} = F_n$ which is the correct return value.

Case $m=1$:

The condition on line 8 is false, so we continue to line 11 which leads to returning j on line 12.
$j = F_{n-m+1}$ from invariant 3. Thus, substituting $m=1$,
$j = F_{n-1+1} = F_n$ which is the correct return value.

Case $m=2$:

Since the conditions on line 8 and 11 are false, the algorithm returns k on line 14. $k = i + j$ from invariant 4. By substituting invariants 2 and 3 into this expression, $k = F_{n-m} + F_{n-m+1}$. By definition of Fibonacci, and $m=2$:

$$k = F_{n-2} + F_{n-2+1} = F_{n-2} + F_{n-1}$$
$$k = F_n \text{ which is the correct return value.} \quad \square$$

Termination

On line 4, m is always decremented with each iteration of the loop. Meaning, the loop must exit once $m<3$. Then, based on the proven loop invariant 1, we know m can be one of 3 values: $m=0$, $m=1$, or $m=2$. As shown in the soundness argument, the algorithm will terminate for each of the 3 possible values of m. ■