

Algorithm Correctness Problems

Problem 1.

```

AL1(M)
1. P = 1
2. I = 1
3. while (I < 2*M)
4.     P = P * I
5.     I = I + 2
6. return P

```

Let p_n and i_n be the values of P and I at line 3 after the n th iteration.

Question 1. Write the recursive relation for i_n and the four first values i_0, i_1, i_2, i_3 . What is the (non-recursive) loop invariant for i_n ?

The recursive relation for i_n is derived from line 5 and is

$$i_n = i_{n-1} + 2.$$

The four first values are $i_0 = 1, i_1 = 3, i_2 = 5, i_3 = 7$. Obviously, i_n is the $n + 1$ th positive odd integer and the loop invariant for it is

$$i_n = 2n + 1.$$

Question 2. Write the recursive relation for p_n and the five first values p_0, p_1, p_2, p_3, p_4 . What is the (non-recursive) loop invariant for p_n for $n \geq 1$?

The recursive relation for p_n is derived from line 4 and is

$$p_n = p_{n-1} i_{n-1}.$$

The four first values are

$$\begin{aligned}
 p_0 &= 1, \\
 p_1 &= p_0 i_0 = 1 \cdot 1 = 1, \\
 p_2 &= p_1 i_1 = 1 \cdot 3, \\
 p_3 &= p_2 i_2 = (1 \cdot 3) \cdot 5, \\
 p_4 &= p_3 i_3 = (1 \cdot 3 \cdot 5) \cdot 7.
 \end{aligned}$$

Apparently p_n is the product of the $n - 1$ first odd integers; the loop invariant for it is

$$p_n = \prod_{j=0}^{n-1} (2j + 1), \quad n \geq 1.$$

Question 4. Give a proof by induction for the previous loop invariant.

Base case: For $n = 1$ we have

$$p_n = 1 = \prod_{j=0}^0 (2j + 1),$$

so the base case is true.

Inductive step. Suppose that the formula is true for $n = k$. We will prove that it is also true for $n = k + 1$.

$$\begin{aligned} p_{k+1} &= p_k i_k \\ &= \left(\prod_{j=0}^{k-1} (2j + 1) \right) \cdot (2k + 1) \\ &= \prod_{j=0}^k (2j + 1). \end{aligned}$$

So the formula is true for $n = k + 1$ and the induction is complete.

Question 5. Prove that the algorithm terminates. What is the return value of this algorithm?

We have $i_n = 2n + 1$, which is a strictly increasing sequence of integers and so there is a minimum index k for which

$$i_k = 2k + 1 \geq 2m.$$

For this value i_k of I the condition of the loop is not satisfied and the algorithm will terminate. Since the first odd number that is greater than $2m$ is $2m + 1$ we conclude that $i_k = 2k + 1 = 2m + 1$ and $k = m$. The return value is p_k and it will be

$$p_k = p_m = \prod_{j=0}^{m-1} (2j + 1).$$

Problem 2. Give a (full) proof of correctness for the following algorithm for the following conditions:

Precondition: `elements` is an ascending order set of size `size`.

Postcondition: If `to_find` is in `elements` the algorithm returns a position in which `to_find` is located, otherwise it returns `NOT_FOUND`.

```

    binary_search( to_find, elements[], size)
1.  lowpoint = 1
2.  highpoint = size
3.  while lowpoint < highpoint
4.      pivot = lbound(( lowpoint+ highpoint)/ 2)
5.      if (to_find > elements[pivot])
6.          lowpoint = pivot + 1
7.      else
8.          highpoint = pivot
9.  if to_find = elements[lowpoint] then
10.     return lowpoint
11. else
12.     return NOT_FOUND

```

Answer: Let l_n, h_n, p_n be the values of `lowpoint`, `highpoint` and `pivot` after the n th iteration. Moreover let $f, m, e[i]$ be the values of `to_find`, `size` and `elements` at the i th position. We will denote S the set of the values in `elements`.

We interpret the algorithm to following relations:

1. $l_0 = 1$
2. $h_0 = m$
4. $p_n = \lfloor \frac{l_{n-1} + h_{n-1}}{2} \rfloor$
6. $(f > e[p_n]) \rightarrow (l_n = p_n + 1)$
8. $(f \leq e[p_n]) \rightarrow (h_n = p_n)$

The last two lines define two cases in the loop:

Case 1. $f > e[p_n]$. Then $l_n = p_n + 1 = \lfloor \frac{l_{n-1} + h_{n-1}}{2} \rfloor + 1$ and $h_n = h_{n-1}$.

Case 2. $f \leq e[p_n]$. Then $h_n = p_n = \lfloor \frac{l_{n-1} + h_{n-1}}{2} \rfloor$ and $l_n = l_{n-1}$.

Step 1, the Loop Invariant. The loop invariant should express what is the purpose of the loop; here the loop is trying to "trap" the value f which we want to locate between the two values $e[l_n], e[h_n]$. If $f \in S$ then we will have $e[l_n] \leq f \leq e[h_n]$. So the loop invariant should be

$$(f \notin S) \vee (e[l_n] \leq f \leq e[h_n]). \quad (1)$$

Proof by induction. First we observe that if $f \notin S$ then the assertion is always true; we will examine the case $f \in S$.

Base case: $n = 0$. Since $f \in S$ and S is a sorted set we have

$$e[l_0] = e[1] \leq f \leq e[m] = e[h_0],$$

so (1) holds for $n = 0$.

Inductive step: Suppose (1) holds for $n = k$, we will prove it holds for $n = k + 1$. We have two cases:

Case 1. $f > e[p_{k+1}]$. Then $f \geq e[p_{k+1} + 1]$ because we have assumed that $f \in S$ and S is an ordered set. We have $l_{k+1} = p_{k+1} + 1$, so $f \geq e[l_{k+1}]$. On the other hand we have $h_{k+1} = h_k$ and $f \leq e[h_k] = e[h_{k+1}]$ from our assumption.

Case 2. $f \leq e[p_{k+1}]$. Then $l_{k+1} = l_k$, so $f \geq e[l_{k+1}]$ from our assumption. On the other hand we have $h_{k+1} = p_{k+1}$, so $f \leq e[h_{k+1}]$.

In both cases (1) holds for $n = k + 1$ and the induction is completed.

Step 2, The Termination. The quantity that strictly decreases in this loop is the distance

$$d_n = h_n - l_n.$$

We can see this examining the two cases:

Case 1. $f > e[p_n]$. Then $l_n = p_n + 1 = \lfloor \frac{l_{n-1} + h_{n-1}}{2} \rfloor + 1$ and $h_n = h_{n-1}$. So

$$\begin{aligned} d_n &= h_n - l_n \\ &= h_{n-1} - \left(\lfloor \frac{l_{n-1} + h_{n-1}}{2} \rfloor + 1 \right) \\ &< h_{n-1} - \frac{l_{n-1} + h_{n-1}}{2} \\ &= \frac{h_{n-1} - l_{n-1}}{2} \\ &= \frac{d_{n-1}}{2}. \end{aligned}$$

Case 2. $f \leq e[p_n]$. Then $h_n = p_n = \lfloor \frac{l_{n-1} + h_{n-1}}{2} \rfloor$ and $l_n = l_{n-1}$. So

$$\begin{aligned} d_n &= h_n - l_n \\ &= \lfloor \frac{l_{n-1} + h_{n-1}}{2} \rfloor - l_{n-1} \\ &\leq \frac{l_{n-1} + h_{n-1}}{2} - l_{n-1} \\ &= \frac{h_{n-1} - l_{n-1}}{2} \\ &= \frac{d_{n-1}}{2}. \end{aligned}$$

In both cases we have that $d_n < d_{n-1}$, so we have a strictly decreasing sequence and there is a minimum index k for which we have $d_k \leq 0$. For this index we have

$$d_k = h_k - l_k \leq 0 \Rightarrow h_k \leq l_k,$$

and the condition of the loop $h_n > l_n$ is not satisfied, so the loop will terminate.

Step 3, The Return Value. Next we want to find the value of d_k . Since k is the first index for which $d_k \leq 0$ we have $d_{k-1} > 0$. We can see that

$$l_{k-1} \leq p_k < h_{k-1}$$

so in both cases we have

$$l_k \leq h_k$$

which means that $d_k \geq 0$ and because $d_k \leq 0$ we have finally

$$d_k = 0.$$

So we have $h_k = l_k$ and if we write the loop invariant for this index we have

$$(f \notin S) \vee (e[l_k] \leq f \leq e[h_k] = e[l_k]).$$

If $f \in S$ then $e[l_k] \leq f \leq e[l_k]$ must be true, so $f = e[l_k]$ and the return value l_k is the index where f was found. Reversevly, if $f \neq e[l_k]$ means that $f \notin S$ because the loop invariant is always true; in this case we return a `NOT_FOUND` value. This proves that the postcondition is true and this completes the proof of correctness. ■