The Australian National University
School of Computing
Dirk Pattinson and Victor Rivera

Semester 2, 2021
Practical Session 6

Foundations of Computation

The practical contains a number of exercises designed for the students to practice the course content. During the practical session, the tutor will work through some of these exercises while students will be responsible for completing the remaining exercises in their own time. There is no expectation that all the exercises will be covered in the practical session.

Covers: Lecture Material Week 6

At the end of this tutorial, you will be able to prove the partial and total correctness of imperative programs using Hoare Logic.

Exercise 1 — Hoare Logic: Simple Loops

Consider the Hoare triple

$$\{P\}\ \texttt{while x>a do x:=x-1}\ \{x = 0\}$$

where P is an (as of yet unspecified) predicate.

1. find the weakest precondition P (in terms of x and a) that makes this Hoare-triple true. Justify your answer briefly (no formal proof required).
2. Give a proof of validity of this triple using the precondition P that you have identified above.

Solution.

1. for $x = 0$ to hold in the post $a = 0$, if x is negative initially, then the loop will not b ondition will also be false. We therefore require $P = (a = 0) \wedge (x \geq 0)$.
2. We give a proof of $\{(a = 0) \wedge (x \geq 0)\}$ while x>a do x:=x-1 $\{x = 0\}$.
   For the invariant I we require that
   - $\{I \wedge x > a\}\ \texttt{x} := \texttt{x} - 1\ \{I\}$, i.e. we can prove the premise of the while rule
   - $a = 0 \wedge x \geq 0 \to I$, i.e. the overall precondition implies the conclusion of the while-rule.
   - $I \wedge \neg(x > a) \to x = 0$, i.e. postcondition of the while rule implies the overall intended postcondition.

   As invariant, we therefore chose $I = (a = 0) \wedge (x \geq a)$. This gives the following proof:

   1. $\{a = 0 \wedge x - 1 \geq a\}\ \texttt{x} := \texttt{x} - 1\ \{(a = 0) \wedge (x \geq a)\}$ (Assignment)
   2. $(a = 0) \wedge (x \geq a) \wedge (x > a) \to a = 0 \wedge (x - 1 \geq a)$ (Basic Maths)
   3. $\{a = 0 \wedge x \geq a \wedge x > a\}\ \texttt{x} := \texttt{x} - 1\ \{a = 0 \wedge x \geq a\}$ (Precondition Strengthening, 2, 1)
   4. $\{a = 0 \wedge x \geq a\}$ `while x>a do x:= x-1` $\{a = 0 \wedge x \geq a \wedge \neg(x > a)\}$ (While Loop, 3)
   5. $a = 0 \wedge x \geq a \wedge \neg(x > a) \to x = 0$ (Basic Maths)
   6. $\{a = 0 \wedge x \geq a\}$ `while x>a do x:= x-1` $\{x = 0\}$ (Postcondition Weakening, 4, 5)

Exercise 2 — Find the Invariant

Consider the following Hoare triple:

```
{ i = 0 /\ s = 0 /\ n >= 0 }
  while (i < n) do
    i := i + 1;
    s := s + i
{ s = n * (n+1) / 2 }
```

1. Complete the table below

loop iteration	0	1	2	3	4
i	0				
s	0				

by filling in the values of i and s *after* the first, second, etc. iteration of the loop. The values for the 0th loop iteration are the initial values of i and s that are given by the precondition.

Solution. We complete the table as follows:

loop iteration	0	1	2	3	4
i	0	1	2	3	4
s	0	1	$1+2$	$1+2+3$	$1+2+3+4$

2. Using the table above, derive an invariant I, that is, a relation between s and i, that holds both before the loop is being entered, and after each iteration of the loop body.

Solution. With a view to the postcondition, we note that $I \equiv s = 1+2+\ldots+i$. Using the fact that $\sum_{k=1}^{i} k = i*(i+1)/2$, we can write the invariant as $I \equiv s = i*(i+1)/2$. Another possible invariant is $I \equiv s = i*(i+1)/2 \wedge i \leq n$.

3. Check whether the invariant that you have found is strong enough so that – together with the negation of the loop condition – implies the desired postcondition, and if not, modify the invariant accordingly.

That is, verify whether $I \wedge \neg(i < n) \rightarrow s = n*(n+1)/2$, and modify the invariant if this is not the case.

Solution. For the invariant $I \equiv s = i*(i+1)/2$ we would need to prove that

$$s = i*(i+1)/2 \wedge \neg(i < n) \rightarrow s = n*(n+1)/2$$

which is not true (e.g. for $i = 1$ and $s = 1$ and $n = 0$ we have both $s = i*(i+1)/2$ (as $1 = 1*(1+1)/2$) and $\neg(i < n)$ (as it is false that $1 < 0$), but we clearly do *not* have that $s = n*(n+1)/2$, as $1 \neq 0*1/2$.

The missing piece of informa *actly* equal to n, whereas the negation of the loop conditio pn of the loop, we know that $i \leq n$ which is what we $= i*(i+1)/2 \wedge i \leq n$.

For this modified invariant, we obtain that $I \wedge \neg(i < n) \rightarrow s = n \quad (n+1)/2$, because assuming $I \wedge \neg(i < n)$ we have that

- $s = i*(i+1)/2 \wedge i \leq n$ (by writing out I)
- $i \geq n$ (by expanding out $\neg(i < n)$)
- therefore $i = n$, as both $i \leq n$ and $i \geq n$
- and hence $s = n*(n+1)/2$ given that $s = i*(i+1)/2$ and $i = n$.

4. Hence, or otherwise, give a Hoare Logic proof of the Hoare triple above.

Solution. The complete proof is as follows:

1. $\{s+i = i(i+1)/2 \wedge i \leq n\}$ `s := s + i` $\{s = i(i+1)/2 \wedge i \leq n\}$ (Assignment)
2. $\{s+(i+1) = (i+1)(i+1+1)/2 \wedge i+1 \leq n\}$ `i := i + 1` $\{s+i = i(i+1)/2 \wedge i \leq n\}$ (Assignment)
3. $s = i(i+1)/2 \wedge i \leq n \wedge i < n \rightarrow s+(i+1) = (i+1)(i+1+1)/2 \wedge i+1 \leq n$ (Basic Maths)
4. $\{s = i(i+1)/2 \wedge i \leq n \wedge i < n\}$ i := i+1 $\{s+i = i(i+1)/2 \wedge i \leq n\}$ (Precondition Strengthening, 2, 3)
5. $\{s = i(i+1)/2 \wedge i \leq n \wedge i < n\}$ i := i+1; s := s + i $\{s = i(i+1)/2 \wedge i \leq n\}$ (Sequence, 1, 4)
6. $\{s = i(i+1)/2 \wedge i \leq n\}$ `program` $\{s = i(i+1)/2 \wedge i \leq n \wedge \neg(i < n)\}$ (While Loop, 5)
7. $s = i(i+1)/2 \wedge i \leq n \wedge \neg(i < n) \rightarrow s = n(n+1)/2$ (Basic Maths)
8. $\{s = i(i+1)/2 \wedge i \leq n\}$ `program` $\{s = n(n+1)/2\}$ (Postcondition Weakening, 6,7)
9. $s = 0 \wedge i = 0 \wedge n \geq 0 \rightarrow s = i(i+1)/2 \wedge i \leq n$ (Basic Maths)
10. $\{s = 0 \wedge i = 0 \wedge n \geq 0\}$ `program` $\{s = n(n+1)/2\}$ (Precondition Strengthening, 8, 9)

Exercise 3 **Find The Variant**

Consider the following Hoare triple, but now formulated in terms of total correctness:

```
[ i = 0 /\ s = 0 /\ n >= 0 ]
  while (i < n) do
    i := i + 1;
    s := s + i
[ s = n * (n+1) / 2 ]
```

1. Identify and state a *variant* E for the loop. Using the same invariant I as in the previous exercise, the variant should have the following two properties:

   - it should be ≥ 0 when the loop is entered, i.e. $I \wedge (i < n) \rightarrow E \geq 0$
   - it should decrease every time the loop body is executed, i.e. $[I \wedge b \wedge E = k]$ `body` $[I \wedge E < k]$

   where `body` stands for the body of the loop. You just need to state the variant, and do not need to prove the two bullet points above (yet).

   Solution. Since i is increasing by one each loop, and is counting up to n, then $n - i$ will decrease each loop. Hence, we chose the variant $E \equiv n - i$.

2. For the variant E that you have identified, give a proof of the premise of the while-rule for total correctness, i.e. give a Hoare-logic proof of $[I \wedge (i < n) \wedge E = k]$ `body` $[I \wedge E < k]$, and argue that $I \wedge (i < n) \rightarrow E \geq 0$.

   Solution. Recall that the invariant is $I \equiv s = i(i+1)/2 \wedge i \leq n$ and the loop condition is $b \equiv i < n$. This means that $I \wedge b$ is equivalent to $s = i(i+1)/2 \wedge i \leq n \wedge i < n$. In particular, this implies $n - i \geq 0$.

   We give the following Hoare-Logic proof:

   1. $[s + i = i(i+1)/2 \wedge i \leq n \wedge n - i < k]$ `s := s + i` $[s = i(i+1)/2 \wedge i \leq n \wedge n - i < k]$ (Assignment)
   2. $[s+i+1 = (i+1)(i+1+1)/2 \wedge i+1 \leq n \wedge n-(i+1) < k]$ `i := i + 1` $[s+i = i(i+1)/2 \wedge i \leq n \wedge n-i < k]$ (Assignment)
   3. $[s+i+1 = (i+1)(i+1+1)/2 \quad i+1 \quad n \quad n \quad (i+1) < k]$ `i := i + 1; s := s + i` $[s = i(i+1)/2 \wedge i \leq n \wedge n - i < k]$ (Se
   4. $s = i(i+1)/2 \wedge i$ $\leq n \wedge n - (i+1) < k$ (Basic Maths, similar t
   5. $[s = i(i+1)/2 \wedge i \leq n \wedge i < n \wedge n - i = k]$ `i := i +` $\wedge i \leq n \wedge n - i < k]$ (Precondition Strengthening, 3, 4)

3. Hence, or otherwise, give a Hoare-logic proof of $[I]$ `while (i < n) d` $]$.

   Solution. We continue the above proof.

   6. $I \wedge b \rightarrow E \geq 0$ (Basic Maths, argued above)
   7. $[I]$ `while (i< n) do body` $[I]$ (While Loop, 5, 6).

Exercise 4 **More While Loops**

Give a proof of the following Hoare-triples, where you may use that we have established the validity of $\{s = 2^i\}$ `i:=i+1; s:=s*2` $\{s = 2^i\}$ in Tutorial 5.

1. $\{s = 2^i\}$ `while i<n do i:=i+1; s:=s*2` $\{s = 2^i\}$

   Solution. We will use the validity of the Hoare triple above that $s = 2^i$ is invariant for the body of this loop. Write `Loop` to abbreviate the whole loop:

   1. $\{s = 2^i\}$ `i := i + 1; s := s * 2` $\{s = 2^i\}$ (Given)
   2. $((s = 2^i) \wedge (i < n)) \rightarrow (s = 2^i)$ (Basic Logic)
   3. $\{(s = 2^i) \wedge (i < n)\}$ `i := i + 1; s := s * 2` $\{s = 2^i\}$ (Precondition Strengthening, 1, 2)
   4. $\{s = 2^i\}$ `Loop` $\{(s = 2^i) \wedge \neg(i < n)\}$ (While Loop, 3)
   5. $((s = 2^i) \wedge \neg(i < n)) \rightarrow (s = 2^i)$ (Basic Logic)
   6. $\{s = 2^i\}$ `Loop` $\{s = 2^i\}$ (Postcondition Weakening, 4, 5)

2. $\{(s = 2^i) \wedge (i \leq n)\}$ `while i<n do i:=i+1; s:=s*2` $\{s = 2^n\}$

   Solution. We require that $I \wedge \neg b \rightarrow s = 2^n$, so choosing the same invariant $I \equiv s = 2^i$ as before is too weak, as $s = 2^i \wedge i \geq n \not\rightarrow s = 2^n$. We want to strengthen I as little as possible such that the implication does hold, and the extra information required to conclude $s = 2^n$ is that $i \leq n$, so we choose the invariant $I \equiv (s = 2^i) \wedge i \leq n$.

   Formally, again writing `Loop` to abbreviate the whole loop:

   1. $\{s * 2 = 2^i \wedge i \leq n\}$ `s := s * 2` $\{s = 2^i \wedge i \leq n\}$ (Assignment)
   2. $\{s * 2 = 2^{i+1} \wedge i + 1 \leq n\}$ `i := i + 1` $\{s * 2 = 2^i \wedge i \leq n\}$ (Assignment)
   3. $(s = 2^i \wedge i \leq n \wedge i < n) \rightarrow (s * 2 = 2^{i+1} \wedge i + 1 \leq n)$ (Basic Maths.)
   4. $\{s = 2^i \wedge i \leq n \wedge i < n\}$ `i := i + 1` $\{s * 2 = 2^i \wedge i \leq n\}$ (Precondition Strengthening, 2, 3)
   5. $\{s = 2^i \wedge i \leq n \wedge i < n\}$ `i := i + 1; s := s * 2` $\{s = 2^i \wedge i \leq n\}$ (Sequence, 1, 4)
   6. $\{s = 2^i \wedge i \leq n\}$ `Loop` $\{s = 2^i \wedge i \leq n \wedge \neg(i < n)\}$ (While Loop, 5)
   7. $(s = 2^i \wedge i \leq n \wedge \neg(i < n)) \rightarrow (s = 2^n)$ (Basic Maths.)
   8. $\{s = 2^i \wedge i \leq n\}$ `Loop` $\{s = 2^n\}$ (Postcondition Weakening, 6, 7)

Exercise 5 **Multiplication (Partial Correctness)**

Consider the following (annotated) program that multiplies two numbers by means of repeated addition:

```
{n >=  1}
  p := 0;
  i := 1;
  while (i <= n) do
    p := p+m;
    i := i+1
{p = m * n}
```


1. Identify the strongest mid-con

   ```
   {n >=  1}
     p := 0;
     i := 1;
   { M }
   ```

   is provable, and give a Hoare-logic proof of this fact.

2. Identify an *invariant* I for the loop. The invariant should have the following three properties:

   - it should be *strong enough* to imply the postcondition if the loop condition is false: $I \wedge \neg(i \leq n) \rightarrow p = m * n$
   - it should be *weak enough* so that it is implied by the mid-condition M, that is $M \rightarrow I$.
   - it should be an invariant, i.e $\{I \wedge (i \leq n)\}$ `body` $\{I\}$ should be provable.

   State the invariant, and give a Hoare-Logic proof of $\{I \wedge (i < n)\}$ `body` $\{I\}$ where `body` is the body of the while-loop.

3. Using the above, give a proof of the Hoare-triple given by the annotated program at the beginning of the exercise.

Solution.

1. Before we run the code, we have that $n \geq 1$, and after we run the code, we can guarantee that $n \geq 1$, $p = 0$ and $i = 1$. Hence, we choose the midcondition $M = n \geq 1 \wedge p = 0 \wedge i = 1$ and give the following proof.

   1. $\{n \geq 1 \wedge p = 0 \wedge 1 = 1\}$ `i := 1` $\{n \geq 1 \wedge p = 0 \wedge i = 1\}$ (Assignment)
   2. $\{n \geq 1 \wedge 0 = 0 \wedge 1 = 1\}$ `p := 0` $\{n \geq 1 \wedge p = 0 \wedge 1 = 1\}$ (Assignment)
   3. $\{n \geq 1 \wedge 0 = 0 \wedge 1 = 1\}$ `p := 0; i := 1` $\{n \geq 1 \wedge p = 0 \wedge i = 1\}$ (Sequence, 1, 2)
   4. $(n \geq 1) \rightarrow (n \geq 1 \wedge 0 = 0 \wedge 1 = 1)$ (Basic Maths)
   5. $\{n \geq 1\}$ `p := 0; i := 1` $\{n \geq 1 \wedge p = 0 \wedge i = 1\}$ (Precondition Strengthening, 3, 4)

2. We draw a table showing the state of the variables, as shown:

loop iteration	0	1	2	3	4
p	0	m	$2m$	$3m$	$4m$
i	1	2	3	4	5

From this table, we guess the invariant $I \equiv p = m * (i-1)$. Unfortunately, it is still too weak, as

$$I \wedge \neg b \equiv p = m * (i-1) \wedge i > n \not\rightarrow p = m * n.$$

Given $I \wedge \neg b$, we have that $p = m * (i-1)$ and that $i > n \equiv i \geq n+1$. The minimum extra information required to conclude that $p = m * n$ would be if we also had that $i \leq n+1$ (which is an invariant property, as the code stops running when $i \leq n$ is false.) Hence, we choose as our invariant $I \equiv p = m * (i-1) \wedge i \leq n+1$, and give the following proof:

6. $\{p = m * (i+1-1) \wedge (i+1) \leq n+1\}$ `i := i + 1` $\{p = m * (i-1) \wedge i \leq n+1\}$ (Assignment)
7. $\{p+m = m*(i+1-1) \wedge (i+1) \leq n+1\}$ `p := p + m` $\{p = m*(i-1+1) \wedge i+1 \leq n+1\}$ (Assignment)
8. $\{p+m = m*(i+1-1) \wedge i+1 \leq n+1\}$ `p := p + m; i := i + 1` $\{p = m*(i-1) \wedge i \leq n+1\}$ (Sequence, 6, 7)
9. $p = m*(i-1) \wedge i \leq n+1 \wedge i \leq n \rightarrow p+m = m*(i+1-1) \wedge i+1 \leq n+1$ (Basic Maths)
10. $\{p = m*(i-1) \wedge i \leq n+1 \wedge i \leq n\}$ `body` $\{p = m*(i-1) \wedge i \leq n+1\}$ (Precondition Strengthening, 8, 9)

3. We apply the while-rule and glue the pieces together with precondition strengthening and postcondition weakening. We write `program` for the entire program (in the last line of the proof)

11. $\{I\}$ `while (i <= n) do body` $\{I \wedge \neg(i \leq n)\}$ (While Loop, 10)
12. $M \rightarrow I$ (Basic Maths, M the mid-condition above)
13. $\{M\}$ `while (i <= n) do body` I $(i$ $n)$ (Precondition Strengthening, 11, 12)
14. $I \wedge \neg(i \leq n) \rightarrow p$ $n+1)$
15. $\{M\}$ `while (i <= n) do b` $4)$
16. $\{n \geq 1\}$ `progra`

Exercise 6 **Multiplication (Total Correctness)**

We consider the same program fragment as above, but now the goal is to establish total c

```
[n >=  1]
  p := 0;
  i := 1;
  while (i <= n) do
    p := p+m;
    i := i+1
[p = m * n]
```

1. Identify and state a *variant* E for the loop. Using the same invariant I as in the previous exercise, the variant should have the following two properties:

   - it should be ≥ 0 when the loop is entered, i.e. $I \wedge b \rightarrow E \geq 0$
   - it should decrease every time the loop body is executed, i.e. $[I \wedge b \wedge E = k]$ `body` $[I \wedge E < k]$

   where `body` stands for the body of the loop. You just need to state the variant, and do not need to prove the two bullet points above (yet).

   Solution. The variable i is counting up towards n, so we chose the variant $E \equiv n - i$.

2. For the variant E that you have identified, give a proof of the premise of the while-rule for total correctness, i.e. give a Hoare-logic proof of $[I \wedge b \wedge E = k]$ `body` $[I \wedge E < k]$, and argue that $I \wedge b \rightarrow E \geq 0$.

   Solution. Recall that the invariant is $I \equiv p = m * (i-1) \wedge i \leq n+1$ and the loop condition is $b \equiv i \leq n$. This means that $I \wedge b$ is equivalent to $p = m * (i-1) \wedge i \leq n+1 \wedge i \leq n$. In particular, this implies $i \leq n$, hence $n - i \geq 0$.

   We give the following Hoare-Logic proof:

1. $[p = m*(i+1-1) \wedge i+1 \leq n+1 \wedge n-(i+1) < k] i := i+1 [p = m*(i-1) \wedge i \leq n+1 \wedge n-i < k]$ (Assignment)
2. $[p+m = m*(i+1-1) \wedge i+1 \leq n+1 \wedge n-(i+1) < k] p := p+m [p = m*(i+1-1) \wedge i+1 \leq n+1 \wedge n-(i+1) < k]$ (Assignment)
3. $[p+m = m*(i+1-1) \wedge i+1 \leq n+1 \wedge n-(i+1) < k] p := p+m; i := i+1 [p = m*(i-1) \wedge i \leq n+1 \wedge n-i < k]$ (Sequence, 1, 2)
4. $p = m*(i-1) \wedge i \leq n+1 \wedge i \leq n \wedge n-i = k \rightarrow p+m = m*(i+1-1) \wedge i+1 \leq n+1 \wedge n-(i+1) < k$ (Basic Maths)
5. $[p = m*(i-1) \wedge i \leq n+1 \wedge i \leq n \wedge n-i = k] p := p+m; i := i+1 [p = m*(i-1) \wedge i \leq n+1 \wedge n-i < k]$ (Precondition Strengthening, 3, 4)

3. Hence, or otherwise, give a Hoare-logic proof of $[I]$ `while (i <= n) do body` $[I \wedge \neg b]$.

   Solution. We continue the above proof.

   6. $I \wedge b \rightarrow E \geq 0$ (Basic Maths, argued above)
   7. $[I]$ `while (i <= n) do body` $[I \wedge \neg b]$ (While Loop, 5, 6).

Exercise 7 **Fibonacci Numbers**

Recall the Fibonacci numbers defined by $f(0) = 0$, $f(1) = 1$ and $f(n) = f(n-2) + f(n-1)$ for $n \geq 2$ and consider the following code fragment that we have annotated with pre and postcondition.

```
{ x = 0 /\ y = 1 /\ z = 1 /\ n >= 1}
  while (z<n) do
    y := x+y;
    x := y-x;
    z := z+1
{ y = f(n) }
```


1. Identify an *invariant* I f
   - it should be *strong enough* to imply the postcondition if the loop c $(z < n) \rightarrow y = f(n)$.
   - it should be *weak enough* so that it is implied by the precondition, i $\wedge z = 1 \wedge n \geq 1 \rightarrow I$.
   - it should be an invariant, i.e $\{I \wedge (i \leq n)\}$ `body` $\{I\}$ shou is the body of the loop.

   State the invariant, and give a Hoare-Logic proof of the fact that I is indeed an invariant, i.e. show that $\{I \wedge (z < n)\}$ `y := x + y; x := y - x; z := z + 1` $\{I\}$ is provable in Hoare Logic.

   Hint. At what positions in the Fibonacci sequence do x and y occur?

2. Hence, or otherwise, establish the validity of the Hoare triple given by the annotated program above.

Solution.

1. We run the code, and observe the variables during execution. Recall that the fibonacci sequence is given by $0, 1, 1, 2, 3, 5, 8, 13, \ldots$

loop iteration	0	1	2	3	4	5	6
y	1	1	2	3	5	8	13
x	0	1	1	2	3	5	8
z	1	2	3	4	5	6	7

   From the table, we can see that $y = f(z)$, and that $x = f(z-1)$, so we guess $I \equiv y = f(z) \wedge x = f(z-1)$. This invariant is too weak however, as $I \wedge \neg b \equiv y = f(z) \wedge x = f(z-1) \wedge z \geq n \not\rightarrow y = f(n)$. The minimum extra information required is that $z \leq n$ (which is an invariant property, as the code only executes when $z < n$. Hence, we choose $I \equiv y = f(z) \wedge x = f(z-1) \wedge z \leq n$. We establish the premise of the while rule as follows:

   1. $\{y = f(z+1) \wedge x = f(z+1-1) \wedge (z+1) \leq n\}$ `z := z + 1` $\{y = f(z) \wedge x = f(z-1) \wedge z \leq n\}$ (Assignment)
   2. $\{y = f(z+1) \wedge y - x = f(z+1-1) \wedge z+1 \leq n\}$ `x := y - x` $\{y = f(z+1) \wedge x = f(z+1-1) \wedge (z+1) \leq n\}$ (Assignment)

3. $\{x + y = f(z + 1) \wedge x + y - x = f(z + 1 - 1) \wedge z + 1 \leq n\}$ `y := x + y` $\{y = f(z + 1) \wedge y - x = f(z + 1 - 1) \wedge z + 1 \leq n\}$ (Assignment)
4. $(y = f(z) \wedge x = f(z-1) \wedge z \leq n \wedge z < n) \rightarrow (x+y = f(z+1) \wedge x+y-x = f(z+1-1) \wedge z+1 \leq n)$
5. $\{I \wedge (z < n)\}$ `y := x + y` $\{y = f(z+1) \wedge y - x = f(z+1-1) \wedge z+1 \leq n\}$ (Precondition Strengthening, 4, 3)
6. $\{I \wedge (z < n)\}$ `y := x + y; x := y - x` $\{y = f(z + 1) \wedge x = f(z + 1 - 1) \wedge (z + 1) \leq n\}$ (Sequence, 5, 2)
7. $\{I \wedge (z < n)\}$ `y := x + y; x := y - x; z := z + 1` $\{I\}$ (Sequence, 6, 1)

2. We continue the linear proof, and apply the while rule. We abbreviate the loop body by `body`.

8. $\{I\}$ `while (z<n) do body` $\{I \wedge \neg(z < n)\}$ (While, 7)
9. $I \rightarrow (z = f(n))$ (Basic Maths)
10. $\{I\}$ `while (z<n) do body` $\{z = f(n)\}$ (Postcondition Weakening, 8, 9)
11. $(x = 0 \wedge y = 1 \wedge z = 1 \wedge n \geq 1) \rightarrow I$ (Basic Maths)
12. $\{x = 0 \wedge y = 1 \wedge z = 1 \wedge n \geq 1\}$ `while (z<n) do body` $\{z = f(n)\}$ (Precondition Strengthening, 10, 11)

Exercise 8 **Total Correctness**

Consider the program that computes Fibonacci numbers from the previous exercise, but from the perspective of *total* correctness. That is, we want to establish provability of the following Hoare triple:

```
[ x = 0 /\ y = 1 /\ z = 1 /\ n >= 1 ]
  while (z<n) do
    y := x+y;
    x := y-x;
    z := z+1
[ y = f(n) ]
```


1. We use the same invariant I as before. Find a suitable *variant* E for the loop. The variant should have the following two properties:

   - it should be ≥ 0 when the loop is entered, i.e. $I \wedge (z < n) \rightarrow E \geq 0$
   - it should decrease every time the loop body is executed, i.e. $[I \wedge (z < n) \wedge E = k]$ `body` $[I \wedge E < k]$

   where `body` stands for the body of the loop as above.

   Solution. The loop variant is $E \equiv n - z$. This is ≥ 0 if $z < n$ holds, and decreases (as z is increased in each iteration).

2. For the variant E you have identified above, give a proof of the premise of the while-rule for total correctness, i.e. give a Hoare-logic proof of $[I \wedge (z < n) \wedge E = k]$ `body` $[I \wedge E < k]$ and argue that $I \wedge b \rightarrow E \geq 0$.

   Solution.

   Recall that the loop invariant is $I \equiv y = f(z) \wedge x = f(z-1) \wedge z \leq n$ and the loop condition is $b \equiv z < n$. Assuming $I \wedge b$, we have in particular that $z < n$, hence $E \equiv n - z \geq 0$. The proof is as follows:

   1. $[y = f(z+1) \wedge x = f(z+1-1) \wedge (z+1) \leq n \wedge n-(z+1) < k]$ `z := z + 1` $[y = f(z) \wedge x = f(z-1) \wedge z \leq n \wedge n - z < k]$ (Assignment)
   2. $[y = f(z + 1) \wedge y - x = f(z + 1 - 1) \wedge z + 1 \leq n \wedge n - (z + 1) < k]$ `x := y - x` $[y = f(z + 1) \wedge x = f(z + 1 - 1) \wedge (z + 1) \leq n \wedge n - (z + 1) < k]$ (Assignment)
   3. $[x+y = f(z+1) \wedge x+y-x = f(z+1-1) \wedge z+1 \leq n \wedge n-(z+1) < k]$ `y := x + y` $[y = f(z+1) \wedge y-x = f(z + 1 - 1) \wedge z + 1 \leq n \wedge n - (z + 1) < k]$ (Assignment)
   4. $(y = f(z) \wedge x = f(z - 1) \wedge z \leq n \wedge z < n \wedge n - z = k) \rightarrow (x + y = f(z + 1) \wedge x + y - x = f(z + 1 - 1) \wedge z + 1 \leq n \wedge n - (z + 1) < k)$ (Basic Maths)
   5. $[I \wedge (z < n) \wedge n - z = k]$ `y := x + y` $[y = f(z+1) \wedge y - x = f(z+1-1) \wedge z+1 \leq n \wedge n - (z+1) < k]$ (Precondition Strengthening, 4, 3)
   6. $[I \wedge (z < n) \wedge n - z = k]$ `y := x + y; x := y - x` $[y = f(z+1) \wedge x = f(z+1-1) \wedge (z+1) \leq n \wedge n-(z+1) < k]$ (Sequence, 5, 2)

7. $[I \wedge (z < n) \wedge n - z = k]$ `y := x + y; x := y - x; z := z + 1` $[I \wedge n - z < k]$ (Sequence, 6, 1)

3. Hence, or otherwise, establish the provability of the Hoare-triple given by the annotated program at the beginning of the exercise.

   Solution. This is just a matter of copy-pasting what we had in the previous exercise, but using the total correctness rule for while. That is, we have the following proof:

   8. $(I \wedge b) \rightarrow (z - n) \geq 0$ (Logic)
   9. $[I]$ `while (z<n) do body` $[I \wedge \neg(z < n)]$ (While Loop, 7, 8)
   10. $I \rightarrow (z = f(n))$ (Basic Maths)
   11. $[I]$ `while (z<n) do body` $[z = f(n)]$ (Postcondition Weakening, 9, 10)
   12. $(x = 0 \wedge y = 1 \wedge z = 1 \wedge n \geq 1) \rightarrow I$ (Basic Maths)
   13. $[x = 0 \wedge y = 1 \wedge z = 1 \wedge n \geq 1]$ `while (z<n) do body` $[z = f(n)]$ (Precondition Strengthening, 11, 12)

Exercise 9 **Counting Modulo 7**

Consider the following code fragment that we refer to as `Count` below, and we refer to the body of the loop (i.e. the two assignments together with the if-statement) as `Body`.

```
while (y < n)
    y := y + 1;
    x := x + 1;
    if (x = 7) then x := 0 else x := x
```


The goal of the exercise is to show that

1. Given the desired postcondition $x < 7$, we need to find an invariant.

   Solution. If we run the code, we start with $x = 0$. Hence, we choose $I \equiv x < 7$ as invariant.

2. Give a Hoare Logic proof of the fact that your invariant above is indeed an invariant, i.e. $\{I\}$ `Body` $\{I\}$.

   Solution. We give the following Hoare-logic proof.

   1. $\{x + 1 \leq 7\}\ y := y + 1\ \{x + 1 \leq 7\}$(Assignment)
   2. $\{x + 1 \leq 7\}\ x := x + 1\ \{x \leq 7\}$ (Assignment)
   3. $\{x + 1 \leq 7\}\ y := y + 1; x := x + 1\ \{x \leq 7\}$ (Sequence, 1, 2)
   4. $x < 7 \rightarrow x + 1 \leq 7$ (Basic Maths)
   5. $\{x < 7\}\ y := y + 1; x := x + 1\ \{x \leq 7\}$ (Precondition Weakening, 3 4)
   6. $\{x < 7\}\ x := x\ \{x < 7\}$ (Assignment)
   7. $x \leq 7 \wedge \neg(x = 7) \rightarrow x < 7$ (Basic Maths)
   8. $\{x \leq 7 \wedge \neg(x = 7)\}\ x := x\ \{x < 7\}$ (Precondition Weakening, 6, 7)
   9. $\{0 < 7\}\ x := 0\ \{x < 7\}$ (Assignment)
   10. $x \leq 7 \wedge x = 7 \rightarrow 0 < 7$ (Basic Maths)
   11. $\{x \leq 7 \wedge x = 7\}\ x := 0\ \{x < 7\}$ (Precondition Weakening, 9, 10)
   12. $\{x \leq 7\}$ `if (x=7) then x := 0 else x := x` $\{x < 7\}$ (Conditional, 8, 11)
   13. $\{x < 7\}$ `Body` $\{x < 7\}$ (Sequence, 5, 12)

3. Hence, or otherwise, give a Hoare-logic proof of the triple $\{x < 7\}$ `Count` $\{x < 7\}$.

   Solution. We continue the above proof as follows.

   14. $x < 7 \wedge y < n \rightarrow x < 7$ (Basic Maths)
   15. $\{x < 7 \wedge y < n\}$ `Body` $\{x < 7\}$ (Precondition Strengthening, 13, 14)

16. $\{x < 7\}$ `Count` $\{x < 7\}$ (While Loop, 15)

4. Give an example of a precondition I so that the Hoare-triple $\{I\}$ `Count` $\{x < 7\}$ does *not* hold and justify your answer briefly.

   Solution. E.g. for $I \equiv y = 0 \wedge n = 0 \wedge x = 12$ we have that the while-loop would be executed zero times so that the value of x in the post-state would still be equal to 12, and hence violates the postcondition $x < 7$.

Appendix — Hoare Logic Rules

- Assignment:

$$\{Q(e)\}\ \texttt{x := e}\ \{Q(x)\}$$

- Precondition Strengthening:

$$\frac{P_s \ \rightarrow\ P_w \qquad \{P_w\}\ S\ \{Q\}}{\{P_s\}\ S\ \{Q\}}$$

  You can always replace predicates by equivalent predicates,
  i.e. if $P_s \leftrightarrow P_w$; just label your proof step with 'precondition equivalence'.

- Postcondition Weakening:

$$\frac{\{P\}\ S\ \{Q_s\} \qquad Q_s\ \rightarrow\ Q_w}{\{P\}\ S\ \{Q_w\}}$$

  You can always replace predicates by equivalent predicates,
  i.e. if $Q_s \leftrightarrow Q_w$; just label your proof step with 'postcondition equivalence'.

- Sequence:

$$\frac{\{P\}\ S_1\ \{Q\} \qquad \{Q\}\ S_2\ \{R\}}{\{P\}\ S_1; S_2\ \{R\}}$$

- Conditional:

$$\frac{\{P \wedge b\}\ S_1\ \{Q\} \qquad \{P \wedge \neg b\}\ S_2\ \{Q\}}{\{P\}\ \textbf{if}\ b\ \textbf{then}\ S_1\ \textbf{else}\ S_2\ \{Q\}}$$

- While Loop:

$$\frac{\{I \wedge b\}\ S\ \{I\}}{I \quad \textbf{while}\ b\ \textbf{do}\ S \quad I \qquad b}$$

- While Loop (Total Correctn

$$\frac{}{[I]\ \textbf{while}\ b\ \textbf{do}\ S\ [}$$

  where n is an auxiliary variable not appearing in I or b or S.