Exercise Sheet 4

General remarks:

- **Due date:** November 18[nd] 12:30 (before the exercise class).
- Please submit your solutions via MOODLE. Remember to provide your matriculation number. It is necessary to hand in your solutions in groups of **three**. You may use the MOODLE forum to form groups.
- Solutions must be written in English.
- While we will publish sketches of exercise solutions, we do *not* guarantee that these sketches contain all details that are necessary to properly solve an exercise. Hence, it is recommended to attend the exercise classes.
- If you have any questions regarding the lecture or the exercise, please use the forum in MOODLE.

Exercise 1 (Markov Chains and Fixed Points) $\boxed{\textbf{25P}}$

Consider a Markov chain $D = (\Sigma, \sigma_I, \mathbf{P})$ with goal set $G \subseteq \Sigma$. Let[1] $\Sigma_? = \Sigma \setminus G$. Further, define the matrix $\mathbf{A}$ via $\mathbf{A} = (\mathbf{P}(\sigma, \tau))_{\sigma,\tau \in \Sigma_?}$, and the vector $\mathbf{b} = (b_\sigma)_{\sigma \in \Sigma_?}$ where $b_\sigma = \sum_{\gamma \in G} \mathbf{P}(\sigma, \gamma)$.

In the rest of this exercise we identify vectors whose entries are indexed by the states in $\Sigma_?$ with functions of type $\Sigma_? \to [0,1]$. For instance, we have $\mathbf{b}\colon \Sigma_? \to [0,1], \mathbf{b}(\sigma) = b_\sigma$.

Now consider the function

$$f\colon (\Sigma_? \to [0,1]) \to (\Sigma_? \to [0,1]),\ \mathbf{x} \mapsto \mathbf{A}\mathbf{x} + \mathbf{b}\ .$$

(a) [15P] Define a partial order on $\Sigma_? \to [0,1]$ and prove that f has a least fixed point wrt. this order.

> **Solution:** We define the partial order $\sqsubseteq$ on $\Sigma_? \to [0,1]$ via $\mathbf{x} \sqsubseteq \mathbf{y}$ iff $\forall \sigma \in \Sigma_?\colon \mathbf{x}(\sigma) \leq \mathbf{y}(\sigma)$ (in words, we compare vectors element-wise).
> Note that $(\Sigma_? \to [0,1], \sqsubseteq)$ is a complete lattice.
> Now we show that f is monotonic. Let $\mathbf{x} \sqsubseteq \mathbf{y}$.
> $\implies$ $\mathbf{A}\mathbf{x} \sqsubseteq \mathbf{A}\mathbf{y}$ because $\mathbf{A}$ has non-negative entries.
> $\implies$ $\mathbf{A}\mathbf{x} + \mathbf{b} \sqsubseteq \mathbf{A}\mathbf{y} + \mathbf{b}$.
> $\iff$ $f(\mathbf{x}) \sqsubseteq f(\mathbf{y})$.
> By the Knaster-Tarski theorem, f has a least fixed point.
>
> *Hint:* lfp f is a vector that contains for each (initial) state in $\Sigma_?$ the probability to reach G. In contrast to the equation system approach from the lecture, we do not have to precompute the states that cannot reach G at all.

(b) [10P] Use the results of lecture #7 to formulate an iterative algorithm that approximates the least fixed point of f *from below*.

[1] Note that this definition of $\Sigma_?$ slightly differs from the one given in lecture #3

Solution: Kleene's fixed point theorem suggest to iterate f as follows:

$$\mathbf{0},\ f(\mathbf{0}),\ f(f(\mathbf{0})),\ \ldots$$

To ensure that this sequence actually converges to lfp f (monotonically; "from below") we have to check that f is continuous.
To this end let $(\mathbf{x}_i)_{i\geq 0}$ be an ascending chain. Note that we have $\bigsqcup_{i\geq 0} \mathbf{x}_i = \lim_{i\to\infty} \mathbf{x}_i$, where lim is meant in the usual sense.
Monotonicity $\implies$ $(f(\mathbf{x}_i))_{i\geq 0} = (\mathbf{A}\mathbf{x}_i + \mathbf{b})_{i\geq 0}$ is also an ascending chain.
By basic facts about limits, we have

$$\lim_{i\to\infty} f(\mathbf{x}_i) \;=\; \lim_{i\to\infty} (\mathbf{A}\mathbf{x}_i + \mathbf{b}) \;=\; \mathbf{A}(\lim_{i\to\infty} \mathbf{x}_i) + \mathbf{b} \;=\; f(\lim_{i\to\infty} \mathbf{x}_i)\ .$$

Exercise 2 (Program verification with weakest pre-conditions) $\boxed{\textbf{25P}}$

(a) [10P] Compute $wp(P, f)$ for the program P and post-condition P below:

P: $\texttt{if}\ (x > 0)\ \{\ x := y + 1\ \}\ \texttt{else}\ \{\ \texttt{if}\ (x = 0)\ \{\ \texttt{skip}\ \}\ \texttt{else}\ \{\ x := x + 1\ \}\ \}$
f: $x > y$

Solution:

$$\begin{aligned}
&wp(\texttt{if}\ (x > 0)\ \{\ x := y + 1\ \}\ \texttt{else}\ \{\ \texttt{if}\ (x = 0)\ \{\ \texttt{skip}\ \}\ \texttt{else}\ \{\ x := x + 1\ \}\ \}, f)\\
=&(x > 0 \wedge wp(x := y + 1, f)) \vee (x \leq 0 \wedge wp(\texttt{if}\ (x = 0)\ \{\ \texttt{skip}\ \}\ \texttt{else}\ \{\ x := x + 1\ \}, f))\\
=&(x > 0 \wedge y + 1 > y) \vee (x \leq 0 \wedge ((x = 0 \wedge wp(\texttt{skip}, f)) \vee (x \neq 0 \wedge wp(x := x + 1, f))))\\
=&(x > 0 \wedge y + 1 > y) \vee (x \leq 0 \wedge ((x = 0 \wedge x > y) \vee (x \neq 0 \wedge x + 1 > y)))\\
=&(x > 0) \vee (x = 0 \wedge x > y) \vee (x < 0 \wedge x + 1 > y)
\end{aligned}$$

From now on, we will also use the following *annotation style* to compute weakest pre-conditions (we write `&` and `|` for logical and/or, respectively, and we assume that `&` binds stronger than `|`). Can you figure out the connection between *wp* and the program annotations? *Hint:* We annotate the program from bottom to top.

```
// (x > 0)  |  (x = 0) & (x > y)  |  (x < 0) & (x + 1 > y)
// (x > 0)  |  (x <= 0) & ( (x = 0) & (x > y)  |  (x != 0) & (x + 1 > y) )
if(x > 0) {
    // true
    // y+1 > y
    x := y + 1
    // x > y
} else {
```

```
    // (x = 0) & (x > y)  |  (x != 0) & (x + 1 > y)
    if(x = 0) {
        // x > y
        skip
        // x > y
    } else {
        // x + 1 > y
        x := x + 1
        // x > y
    }
    // x > y
}
// x > y
```

(b) [15P] For the following loop P, compute the loop-characteristic functional $\Phi_f(f)$ with post-condition f and parameter f. What does the result tell you about lfp Φ_f and $wp(P, f)$? Hint: $(\mathbb{P}, \Rightarrow)$ is a complete lattice.

P: `while` $(x \geq 1)$ $\{\ a := a * x;\ \ x := x - 1\ \}$

f: $a = x!$

Solution:

$$\begin{aligned}
\Phi_f(f) &= ((x \geq 1) \wedge wp(a := a * x,\ wp(x := x - 1,\ f))) \vee ((x < 1) \wedge f) \\
&= ((x \geq 1) \wedge wp(a := a * x,\ wp(x := x - 1,\ a = x!))) \vee ((x < 1) \wedge a = n!) && \text{(def. } f\text{)} \\
&= ((x \geq 1) \wedge wp(a := a * x,\ a = (x - 1)!)) \vee ((x < 1) \wedge a = x!) && \text{(def. } wp\text{)} \\
&= ((x \geq 1) \wedge a * x = (x - 1)!) \vee ((x < 1) \wedge a = x!) && \text{(def. } wp\text{)}
\end{aligned}$$

Due to an unfortunate error in the program, this doesn't tell us anything about the least fixed point of Φ_f. Sorry for that. **The full 15 points will already be awarded for a correct calculation of $\Phi_f(f)$.**

Exercise 3 (Monotonicity of weakest pre-expectations) $\boxed{\textbf{25P}}$

The goal of this exercise is to prove that $wp(P)$ is a *monotonic* predicate transformer.

(a) [8P] Let $(D, \sqsubseteq)$ be a complete lattice and $f, g \colon D \to D$ be monotonic such that for all $d \in D$, $f(d) \sqsubseteq g(d)$. Show that lfp $f \sqsubseteq$ lfp g.
Hint: You may use without proof that lfp $f = \bigsqcap S$, where $S = \{d \in D \mid f(d) \sqsubseteq d\}$.

Solution: We have $f(\text{lfp } g) \sqsubseteq g(\text{lfp } g) = \text{lfp } g$, and thus lfp $g \in S$.
$\implies$ lfp $f = \bigsqcap S \sqsubseteq$ lfp g.

(b) [17P] Prove that for all `GCL` programs P, the function $wp(P)\colon \mathbb{P} \to \mathbb{P}$ is monotonic.

Solution: By induction on the structure of P. Let $F \subseteq G$ be predicates.

- $P = \texttt{skip}$.
  $wp(\texttt{skip}, F) = F \subseteq G = wp(\texttt{skip}, G)$

- $P = x := E$.
  Recall that $wp(x := E, F) = F[x := E]$, where

  $$F[x := E] \;=\; \{s \in \mathbb{S} \mid \exists s' \in F\colon s[x \mapsto E(s)] = s'\} \;.$$

  We have

  $$\begin{aligned} & F \subseteq G \\ \implies & (\exists s' \in F\colon s[x \mapsto E(s)] = s' \implies \exists s' \in G\colon s[x \mapsto E(s)] = s') \\ \implies & F[x := E] \subseteq G[x := E] \\ \implies & wp(x := E, F) \subseteq wp(x := E, G) \end{aligned}$$

- $P = P_1; P_2$.

  $$\begin{aligned} & F \subseteq G \\ \implies & wp(P_2, F) \subseteq wp(P_2, G) && \text{(as } P_2 \text{ is monotonic by the I.H.)} \\ \implies & wp(P_1, wp(P_2, F)) \subseteq wp(P_1, wp(P_2, G)) && \text{(as } P_1 \text{ is monotonic by the I.H.)} \\ \implies & wp(P_1; P_2, F) \subseteq wp(P_1; P_2, G) \;. \end{aligned}$$

- $P = \texttt{if } (\varphi) \ \{ \ P_1 \ \}\texttt{else} \ \{ \ P_2 \ \}$.

  $$\begin{aligned} & F \subseteq G \\ \implies & wp(P_1, F) \subseteq wp(P_1, G) \quad \text{and} \quad wp(P_2, F) \subseteq wp(P_2, G) && \text{(By I.H.)} \\ \implies & \varphi \wedge wp(P_1, F) \subseteq \varphi \wedge wp(P_1, G) \quad \text{and} \quad \neg\varphi \wedge wp(P_2, F) \subseteq \neg\varphi \wedge wp(P_2, G) \\ \implies & (\varphi \wedge wp(P_1, F)) \vee (\neg\varphi \wedge wp(P_2, F)) \;\subseteq\; (\varphi \wedge wp(P_1, G)) \vee (\neg\varphi \wedge wp(P_2, G)) \\ \implies & wp(\texttt{if } (\varphi) \ \{ \ P_1 \ \}\texttt{else} \ \{ \ P_2 \ \}, F) \subseteq wp(\texttt{if } (\varphi) \ \{ \ P_1 \ \}\texttt{else} \ \{ \ P_2 \ \}, G). \end{aligned}$$

- $P = \texttt{while } (\varphi) \ \{ \ P_1 \ \}$.
  $wp(\texttt{while } (\varphi) \ \{ \ P_1 \ \}, F) = \text{lfp } X.\Phi_F(X)$ where

  $$\Phi_F(X) \;=\; (\varphi \wedge wp(P_1, X)) \;\vee\; (\neg\varphi \wedge F)$$

  $F \subseteq G \implies \forall X,\ \Phi_F(X) \subseteq \Phi_G(X)$, and Φ_F, Φ_G are both monotonic.
  By part (a): lfp $X.\Phi_F(X) \subseteq$ lfp $X.\Phi_G(X)$.

Exercise 4 (Weakest liberal pre-conditions) $\boxed{\textbf{25P}}$

(a) [5P] Give a `GCL` program P along with a post-condition F such that $wp(P, F) \neq wlp(P, F)$. You may not use the `diverge` statement in your program!

> **Solution:** Any program that does not terminate for any input state, e.g.
>
> $$P\colon \texttt{while}\ (true)\ \{\ \texttt{skip}\ \}$$
>
> (In fact, this program simulates the `diverge` statement.) Then it holds for any post-condition F that $wlp(P, F) = true$ but $wp(P, F) = false$.

(b) [10P] Determine the inputs for which the following program P does *not* terminate by computing the *weakest liberal pre-condition* with respect to a suitable post-condition:

$$\texttt{while}\ (x \neq 0)\ \{\ x := x - 2\ \}$$

> **Solution:** We consider the post-expectation $F = false$. Then for all initial program states s with $s \models wlp(P, F)$ it holds that P does not terminate on input s, because otherwise it would terminate in a state $t \models false$ which is impossible.
> Recall that by Kleene's fixpoint theorem,
>
> $$wlp(\texttt{while}\ (x \neq 0)\ \{\ x := x - 2\ \}, false) = \inf_{n \in \mathbb{N}} \Psi^n(true)$$
>
> where $\Psi(X) = (x \neq 0 \wedge wlp(x := x-2, X)) \vee (x = 0 \wedge false) = (x \neq 0 \wedge wlp(x := x-2, X))$.
> We iterate Ψ a few times to deduce a pattern:
>
> $$\begin{aligned} \Psi(true) &= (x \neq 0) \\ \Psi^2(true) &= \Psi(x \neq 0) \\ &= x \neq 0 \wedge wlp(x := x - 2, x \neq 0) \\ &= x \neq 0 \wedge x \neq 2 \\ \Psi^3(true) &= \Psi(x \neq 0 \wedge x \neq 2) \\ &= (x \neq 0 \wedge x \neq 2) \wedge wlp(x := x - 2, x \neq 0 \wedge x \neq 2) \\ &= x \neq 0 \wedge x \neq 2 \wedge x \neq 4 \end{aligned}$$
>
> We see that $\Psi^n(true) = \bigwedge_{i=0}^{n-1} x \neq 2i$ (one could formally prove this by induction). The greatest lower bound of this infinite collection of predicates is $x < 0 \vee x \bmod 2 \neq 0$ (that is, x is negative or x is odd). Thus, P does not terminate for negative x and for non-negative odd x. For all other inputs (positive even x) the program terminates.

(c) [10P] Prove by induction on the program structure that for any `GCL` program P and post-condition F it holds that $wp(P, F) \sqsubseteq wlp(P, F)$.
Hint: Use Exercise 3 (b). You may use that $wlp(P)$ is monotonic without proof.

Solution: Let F be an arbitrary post-condition.
The proof is by induction over the structure of P.

- $P = \texttt{skip}$ or $P = (x := E)$: $wp(P, F) = wlp(P, F)$.
- $P = P_1; P_2$:

  $$wp(P_1; P_2, F) = wp(P_1, wp(P_2, F)) \overset{3(\text{a})+I.H.}{\sqsubseteq} wp(P_1, wlp(P_2, F)) \overset{I.H.}{\sqsubseteq} wlp(P_1, wlp(P_2, F))$$

- $P = \texttt{if}\ (G)\ \{\ P_1\ \}\ \texttt{else}\ \{\ P_2\ \}$:

  $$\begin{aligned} wp(\texttt{if}\ (G)\ \{\ P_1\ \}\ \texttt{else}\ \{\ P_2\ \}, F) &= (G \land wp(P_1, F)) \lor (\neg G \land wp(P_2, F)) \\ &\sqsubseteq (G \land wlp(P_1, F)) \lor (\neg G \land wlp(P_2, F)) \\ & wlp(\texttt{if}\ (G)\ \{\ P_1\ \}\ \texttt{else}\ \{\ P_2\ \}, F)\ . \end{aligned}$$

  Note that here we only need the I.H., no monotonicity.

- $P = \texttt{while}\ (G)\ \{\ P_1\ \}$: Recall that $wp(P, F) = \text{lfp}\ X.\Phi_G(X)$ and $wlp(P, F) = \text{gfp}\ X.\Psi_G(X)$.
  I.H. $\implies \Phi_G(X) \sqsubseteq \Psi_G(X)$ for all $X \in \mathbb{P}$.
  3 (a) $\implies$ lfp $\Phi_G \sqsubseteq$ lfp $\Psi_G \sqsubseteq$ gfp Ψ_G.