1.) Given an undirected graph \( G = (V, E) \), we call a set \( S \subseteq V \) self-defending in \( G \) if for each \( v \in S \) and \( u \in V \) with \((v, u) \in E\), there exists a \( w \in S \) with \((u, w) \in E\).

Consider the following decision problem:

**SELF-DEFENSE (SD)**

**INSTANCE:** A directed graph \( G = (V, E) \) and two vertices \( c, d \in V \).

**QUESTION:** Does there exist a set self-defending set \( S \subseteq V \) in \( G \) with \( d \in S \) and \( c \not\in S \).

(a) The following function \( f \) provides a polynomial-time many-one reduction from \( SD \) to \( SAT \): for an instance \( I = ((V, E), c, d) \) of \( SD \) let \( f(I) = \varphi \) over atoms \( x_v \ (v \in V) \):

\[
\varphi = \neg x_c \land x_d \land \bigwedge_{v \in V} \left( \neg x_v \lor \bigwedge_{(v, u) \in E} x_u \lor \bigvee_{(u, w) \in E} x_w \right)
\]

It holds that \( I \) is a yes-instance of \( SD \iff f(I) \) is a yes-instance of \( SAT \).

Show the \( \Rightarrow \) direction of the statement.

(9 points)

(b) In what follows assume the reduction from \( SD \) to \( SAT \) is correct, and recall that \( SAT \) is NP-complete.

Tick the correct statements (for ticking a correct statement a certain number of points is given; ticking an incorrect statement results in a substraction of the same amount; you cannot go below 0 points):

- \( SD \) is NP-complete
- \( SD \) is NP-hard
- \( SD \) is in NP
- \( SD \) is in NP and but not in P
- for any NP-complete problem, there exists a polynomial-time many-one reduction from \( SD \) to that problem
- for any NP-complete problem, there exists a polynomial-time many-one reduction from that problem to \( SD \)

(6 points)
2.) We consider the binary function $\delta$ which was introduced by Rózsa Péter in 1935. Function applications are written in the form $\delta_x(y)$ where $x, y$ are the arguments from $\mathbb{N}_0$ (the set of natural numbers including 0). The function definition is:

$$\delta_x(y) = \begin{cases} 
2y + 1 & \text{if } x = 0; \\
\delta_{x-1}(1) & \text{if } x \neq 0 \text{ and } y = 0; \\
\delta_{x-1}(\delta_x(y-1)) & \text{if } x \neq 0 \text{ and } y \neq 0. 
\end{cases} \quad (1, 2, 3)$$

(a) Use well-founded induction to show

$$\forall x \forall y \left( (x \in \mathbb{N}_0 \land y \in \mathbb{N}_0) \rightarrow \delta_x(y) > x + y \right).$$

**(12 points)**

(b) Suppose $\delta_C$ is an implementation of $\delta$ in the C programming language with $x$ and $y$ of type unsigned integers of size 64 bit (i.e., of type `uint64_t`). Is $\delta_x'(y') = \delta_C(x', y')$ true for all integers $x', y'$ satisfying $0 \leq x', y' \leq \text{UINT64_MAX}$, where $\text{UINT64_MAX}$ is the largest value for a variable of type `uint64_t`?

If so, then prove this fact. Otherwise provide a counterexample with an exact explanation of what is computed and what is happening. **(3 points)**
3.) (a) Let \( p \) be the following IMP program:

\[
x := 0; y := 0; \\
\textbf{while} \ x < n \ \textbf{do} \\
x := x + 1; \\
y := y - 10 \cdot x + 5;
\textbf{od}
\]

Give a loop invariant and variant for the \textbf{while} loop in \( p \) and prove the validity of the total correctness triple \([n = 10] \ p \ [y + 500 = 0]\).

(10 points)

(b) Let \( p \) be the following IMP program:

\[
\textbf{if} \ y - x < 0 \ \textbf{do} \\
a := x; \\
x := y; \\
y := a; \\
\textbf{od} \\
y := y - x; \\
z := z + x \cdot y
\]

Given the program \( p \) above, is it true, that the triple \( \{A\} \ p \ \{B\} \) is valid if and only if \( VC(p,B) \land (A \implies wlp(p,B)) \)? Briefly justify your answer.

(2 points)

(c) Let \( p \) be the following IMP program containing an integer-valued program variable \( x \):

\[
i := 0 \\
\textbf{while} \ x \leq 0 \ \textbf{do} \\
i := i + 1 \\
\textbf{od}
\]

Consider the invalid Hoare triple \( \{true\} \ p \ \{false\} \). Which of the following counterexamples is correct? Tick all the boxes of program states denoting valid counterexamples to the above triple. [Each correct box counts one point, that is you can lose points for incorrectly ticking or leaving the box empty with a minimum of 0 points. You will not lose points for other exercises.]

\[
\square \sigma(x) = 0 \quad \square \sigma(x) = 1 \quad \square \sigma(x) = -1
\]

(3 points)
4. (a) **Simulation**

Provide a non-empty simulation relation $H$ that witnesses $M_1 \leq M_2$, where $M_1$ and $M_2$ are shown below. The initial state of $M_1$ is $s_0$, the initial state of $M_2$ is $t_0$:

**Kripke structure $M_1$:**

**Kripke structure $M_2$:**

\[
\begin{align*}
\text{s0: } & \{a\} \\
\text{s1: } & \{b\} \\
\text{s2: } & \{b\} \\
\text{s3: } & \{a\} \\
\text{s4: } & \{c\}
\end{align*}
\]

\[
\begin{align*}
\text{t0: } & \{a\} \\
\text{t1: } & \{b\} \\
\text{t2: } & \{b\} \\
\text{t3: } & \{c\} \\
\text{t4: } & \{c\} \\
\text{t5: } & \{b\}
\end{align*}
\]

(4 points)
(b)CTL Marking Algorithm

Consider the following Kripke structure $M$:

```
s0: {x, y, z}  s3: {x, y, z}  s1: {y, z}
s2: {y}  s4: {y, z}
```

Execute the **CTL Marking Algorithm** to determine which states $s_i$ satisfy the formulae $\Phi$.

i. $\text{EXEX} x$, and

ii. $\text{AF}(\neg z)$.

In particular,

i. Transform $\Phi$ into an equivalent formula $\Phi'$ in the *existential fragment* of CTL.

ii. List the subformulae of $\Phi'$.

iii. For increasing nesting depth $i$, iteratively give the states $s_i$ marked by subformulae $\phi_0, \psi_0, \phi_1, \psi_1, \ldots$ of $\Phi'$.

iv. Finally, give the return value of the Marking Algorithm. That is, list the states $s_i$ that satisfy formula $\Phi$, i.e., for which states do we have that $M, s_i \models \Phi$?

*Hint:* Recall that the algorithm starts by marking propositional atoms $\phi_0$. It then iteratively marks boolean combinations $\psi_i$ of subformulas $\phi_i$, and temporal operator applications $\phi_{i+1} = \circ \psi_i$ where $\circ \in \{\text{EF}, \text{EU}, \text{EG}, \text{EX}\}$.

i) Answer template for $\Phi = \text{EXEX} x$

Subformulae of $\Phi$: _____________________________________________

Annotate the states of $M$ with the subformulae by which the Marking Algorithm marks them:

```
s0: {x, y, z}  s3: {x, y, z}  s1: {y, z}
s4: {y, z}  s2: {y}
```

States satisfying $\Phi$: _____________________________________________
ii) Answer template for $\Phi = \text{AF}(\neg z)$

Equivalent existential formula $\Phi' \equiv \Phi$: ____________________________

Subformulae of $\Phi'$: ______________________________________________________

Annotate the states of $M$ with the subformulae by which the Marking Algorithm marks them:

States satisfying $\Phi$: ______________________________________________________ (7 points)
(c) **LTL tautologies**

Prove that the following formulas are tautologies, i.e., they hold for every Kripke structure $M$ and every path $\pi$ in $M$, or find a Kripke structure $M$ and path $\pi$ in $M$, for which the formula does not hold and justify your answer.

i. $F(a \land b) \Rightarrow (Fa) \land (Fb)$

ii. $G(a \lor b) \Rightarrow (Ga) \lor (Gb)$

(4 points)