Name: **Room:** **Coordinates:**

0007 *sheet* 1 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0 1 2 3 4 5 6 7 8 9

Dinic's algorithm for maximum flow

Question 1
20 points

The figure below shows a flow network G. Perform **two** iterations of Dinic's algorithm, giving all the intermediate structures (for both iterations), namely:

- the residual network
- the level graph
- the blocking flow

For the *first* level graph also write down the maximum and minimum number of RETREAT steps executed by the algorithm for obtaining the blocking flow.

Name: **Room:** **Coordinates:**

0007 *sheet* 2 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0123456789

Name: **Room:** **Coordinates:**

0007 *sheet* 3 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0 1 2 3 4 5 6 7 8 9

Nagamochi-Ibaraki Algorithm for Global Minimum Cut

Question 2

20 points

The code below is a high-level description of a function used by the Nagamochi-Ibaraki algorithm, where $G = (V, E)$ is a graph and $c : E \to \mathbb{R}^+$ is a capacity function:

`FINDLEGALORDERING`(G,c)

---

a := some vertex of G;
A := $\{a\}$;
while $A \neq V$ **do**
  z := vertex most tightly connected to A;
  $A := A \cup \{z\}$;
return *vertices in order they were added to* A

---

a) Give a definition of a *vertex most tightly connected to* A (for $A \subseteq V$).

b) Write down the Nagamochi-Ibaraki algorithm, using the `FINDLEGALORDERING` procedure. (Your answer may be text or pseudocode – both are OK).

c) Analyse the complexity of the Nagamochi-Ibaraki algorithm.

d) What other algorithm also not based on flows, can be used to solve the Global Minimum Cut problem?

Name: **Room:** **Coordinates:**

0007 *sheet* 4 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0 1 2 3 4 5 6 7 8 9

Name: **Room:** **Coordinates:**

0007 ***sheet*** 5 ***učo*** ***score***

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0 1 2 3 4 5 6 7 8 9

Dynamic programming on trees

Question 3
20 points

A *dominating set* for a graph $G = (V, E)$ is a set $S \subseteq V$ such that each vertex of V is either in S, or is adjacent to a vertex in S. Now consider the following problem:

MIN-WEIGHT DOMINATING SET	
INPUT:	Graph G, weight function $w : V(G) \to \mathbb{R}^+$
TASK:	Find a dominating set $S \subseteq V(G)$ s.t. such that the value $\sum_{v \in S} w(v)$ is minimized.

(Note that the existence of a dominating set of size smaller than given k is one of the classic NP-complete problems.)
Your task is to show that the MIN-WEIGHT DOMINATING SET problem can be, using the dynamic programming approach, solved on **trees** in a *linear time*. If you are unable to do so, try to give at least an algorithm for the version without weights (i.e. you are looking for a dominating set of a minimum size).

Name: **Room:** **Coordinates:**

0007 *sheet* 6 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0123456789

Name: **Room:** **Coordinates:**

0007 *sheet* 7 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please. 0123456789

Edmond's blossom algorithm for maximum matchings **Question 4** **20 points**

Below is the pseudocode of Edmond's blossom algorithm for *bipartite graphs*:
PERFECTBIPARTITE(G)

```
M := ∅
T := ({r}, ∅)                                        // where r is M-exposed
while there exists vw ∈ E s.t. v ∈ B(T) and w ∉ V(T) do
    if w is M-exposed then
        M := AUGMENT(vw)
        if there is no M-exposed vertex in G then
            return M                                 // M is a perfect matching
        else
            T := ({r}, ∅)                            // where r is M-exposed
    else // w is M-covered
        T := EXTEND(vw)
error no perfect matching                            // T is frustrated
```

1. using pseudocode, modify the algorithm so it works for general graphs (Do not forget to give a pseudocode for all auxiliary functions you use, however you do not need to give the code for EXTEND and AUGMENT.)
2. carefully analyse the complexity of your algorithm
3. give an example of a graph G, matching M and an alternating tree T such that T is frustrated even though there exists a perfect matching for G.

Name: **Room:** **Coordinates:**

0007 *sheet* 8 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0123456789

Name: **Room:** **Coordinates:**

0007 *sheet* 9 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0123456789

Basic Minimum Spanning Tree (MST) algorithms

Question 5
10 points

a) For each of the following MST algorithms give

- a high-level description how they work
- their time complexity (with a short justification)

1. **Kruskal**

2. **Jarník (Prim)**

3. **Borůvka**

b) How do you prove the *correctness* of these algorithms?

Name: **Room:** **Coordinates:**

0007 *sheet* 10 *učo* *score*

Do not write anything here! Write down your personal identification number (učo) only. When doing so, follow the digit templates please.

0 1 2 3 4 5 6 7 8 9

Kernelization

Question 6
10 points

In the kernelization algorithm for VERTEX COVER we used the following two rules for an instance (G, k):

VC.1 If v is an isolated vertex, then remove it.
(producing a new instance $(G \setminus v, k)$)

VC.2 If v is a vertex of degree $> k$, then remove it and decrease k.
(producing a new instance $(G \setminus v, k-1)$)

What happens when none of these two rules is applicable?
(I.e. state what do we know at that moment and what should be done next.)