Technische Universität Wien
Institut für Computergraphik und Algorithmen
Algorithms and Complexity Group

186.813 Algorithmen und Datenstrukturen 1 VU 6.0

1.Übungstest SS 2017

27. April 2017

Machen Sie die folgenden Angaben bitte in deutlicher Blockschrift:

Nachname: ______ Vorname: ______

Matrikelnummer: ______ Unterschrift: ______

Legen Sie während der Prüfung Ihren Ausweis für Studierende vor sich auf das Pult.

Sie dürfen die Lösungen nur auf die Angabeblätter schreiben, die Sie von der Aufsicht erhalten. Es ist nicht zulässig, eventuell mitgebrachtes eigenes Papier zu verwenden. Benutzen Sie bitte dokumentenechte Schreibgeräte (keine Bleistifte!).

Die Verwendung von Taschenrechnern, Mobiltelefonen, Tablets, Digitalkameras, Skripten, Büchern, Mitschriften, Ausarbeitungen oder vergleichbaren Hilfsmitteln ist unzulässig.

	A1:	A2:	A3:	Summe:
Erreichbare Punkte:	18	12	20	50
Erreichte Punkte:				

Viel Erfolg!

Aufgabe A1: Algorithmenanalyse **(18 Punkte)**

a) (10 Punkte) Tragen Sie für das Codestück `FunktionA` die **Laufzeit** in Abhängigkeit von n in Θ-Notation in die nachfolgende Tabelle ein. Tragen Sie für das Codestück `FunktionB` den **Rückgabewert** (z) in Abhängigkeit von n in Θ-Notation in die nachfolgende Tabelle ein.

	`FunktionA`
Laufzeit	

	`FunktionB`
Rückgabewert (z)	

`FunktionA`(n):
$a \leftarrow 2$
$b \leftarrow 2n^2$
while $b > 1$
    $a \leftarrow a + 1$
    $c \leftarrow n$
    while $c \geq n$
        $b \leftarrow b - a$
        $a \leftarrow \left\lfloor \frac{a}{2} \right\rfloor$
        $c \leftarrow \left\lfloor \frac{2n}{c} \right\rfloor$

`FunktionB`(n):
$x \leftarrow 1$
for $j \leftarrow 1$ `bis` $\lfloor \log_5 n \rfloor$
    $x \leftarrow 4 \cdot x$
    $z \leftarrow j$
    $x \leftarrow \left\lfloor \frac{x}{2} \right\rfloor$
for $j \leftarrow 1$ `bis` $\lfloor \log_2 x \rfloor$
    $z \leftarrow z - 1$
return z

b) (8 Punkte) Gegeben sei die folgende Funktion $f\colon \mathbb{N}^+ \to \mathbb{R}$

$$f(n) = \begin{cases} \dfrac{1}{2\log(n)} - 2n + 2^n & \text{wenn } n > 42 \text{ und ungerade} \\ \left(\dfrac{7}{2}\right)^n + \dfrac{3}{4}n^2 - \log(n) & \text{wenn } n \leq 42 \\ n^{-3} \cdot 3^n - (n+1)^3 + \log(2^n) & \text{sonst} \end{cases}$$

Kreuzen Sie **in der folgenden Tabelle** die zutreffenden Felder an:

$f(n)$ ist	$O(.)$	$\Omega(.)$	$\Theta(.)$	keines
e^n				
n^2				
3^n				
5				

Jede Zeile wird nur dann gewertet, wenn sie vollständig richtig ist.

Aufgabe A2: Stable-Matching, Starker Zusammenhang (12 Punkte)

a) (8 Punkte)

	1.	2.	3.	4.	5.
V	E	B	A	C	D
W	A	D	C	B	E
X	A	C	D	B	E
Y	E	C	D	A	B
Z	C	B	A	D	E

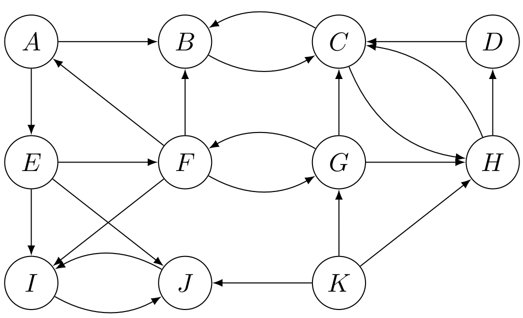
	1.	2.	3.	4.	5.
A	Z	X	Y	V	W
B	X	Y	V	Z	W
C	W	V	X	Z	Y
D	Z	X	Y	W	V
E	Y	X	V	W	Z

Jede Zeile der nachfolgenden Tabelle enthält ein Matching. Stellen Sie jeweils fest, ob dieses in Bezug auf obige Präferenzlisten stabil oder instabil ist. Wenn ein Matching instabil ist, dann zeigen sie das, indem Sie ein instabiles Paar angeben.

Matching	stabil	instabil	Instabiles Paar
V-B, W-C, X-D, Y-A, Z-E			
V-B, W-A, X-D, Y-E, Z-C			
V-B, W-D, X-C, Y-E, Z-A			

b) (4 Punkte) Partitionieren Sie den folgenden Graphen in maximale Teilgraphen, sodass jeder Teilgraph stark zusammenhängend ist. Tragen sie zusammengehörige Knoten jeweils in eine Zeile der nachfolgenden Tabelle ein.

(Wenn Sie nicht alle Zeilen benötigen, lassen sie die übrigen einfach frei.)



Teilgraph	Enthaltene Knoten
1	
2	
3	
4	
5	
6	

Aufgabe A3: Graphen **(20 Punkte)**

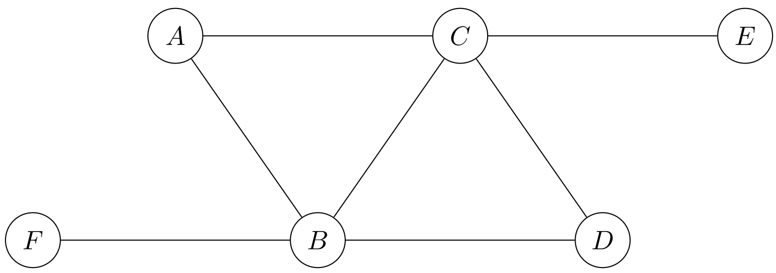
a) (12 Punkte) Gegeben ist der folgende Algorithmus für schlichte, ungerichtete, zusammenhängende Graphen $G = (V, E)$:

Hinweis: Das Array `Markiert` ist global.

```
WasBinIch(G):
X ← ∅
foreach Knoten v ∈ V
    Nachbar[u] ← false für alle Knoten u ∈ V
    Markiert[u] ← false für alle Knoten u ∈ V
    foreach Kante (v, u) inzident zu v
        Nachbar[u] ← true
        q ← u
    Hilfsfunktion(G − {v},q)
    foreach Knoten u ∈ V
        if  Nachbar[u] und !Markiert[u]
            X ← X ∪ {v}
return X
```

```
Hilfsfunktion(G,q):
Markiert[q] ← true
foreach Kante (q, b) inzident zu q
    if  !Markiert[b]
        Hilfsfunktion(G,b)
```

Weiters ist ein Graph $G = (V, E)$ gegeben:



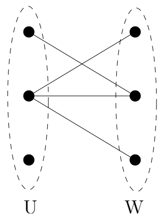
- (3 Punkte) Wenden Sie den Algorithmus *WasBinIch* auf den gegebenen Graphen G an. Geben Sie die vom Algorithmus retournierte Knotenmenge X an.

- (2 Punkte) Welchen, aus der Vorlesung bekannten, Algorithmus verwendet *WasBinIch*?

- (3 Punkte) Beschreiben Sie **kurz** die Funktion des Algorithmus und charakterisieren Sie die Knoten, die in der retournierten Menge X enthalten sind.

- (4 Punkte) Geben Sie die Laufzeit von *WasBinIch* in Θ-Notation in Abhängigkeit der Anzahl der Knoten n und der Kanten m des Eingabegraphen an. Begründen Sie **kurz**, wie sich diese Laufzeit ergibt.

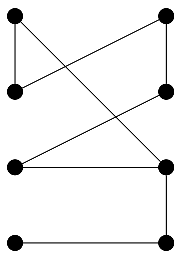
b) (8 Punkte) Ein ungerichteter Graph $G = (V, E)$ heißt *bipartit*, wenn man die Menge V in zwei disjunkte Teilmengen U und W so aufspalten kann, dass für alle Kanten $(u, w) \in E$ gilt: $u \in U$ und $w \in W$. Die untenstehende Abbildung stellt einen bipartiten Graphen dar. Die Teilmengen U und W sind gekennzeichnet.



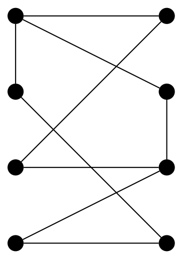
Schreiben Sie **unter** jeden der nachfolgenden Graphen „**Ja**“, wenn er bipartit ist und „**Nein**“ sonst. Beweisen Sie diese Angabe folgendermaßen:

- Ist ein Graph **nicht bipartit**, dann markieren Sie einen minimalen (bzgl. der Anzahl der enthaltenen Knoten und Kanten) Teilgraph der nicht bipartit ist.
- Ist ein Graph **bipartit**, dann teilen Sie die Knoten in zwei disjunkte Teilmengen (gemäß obiger Beschreibung, z.B. durch Färben oder Einkreisen).

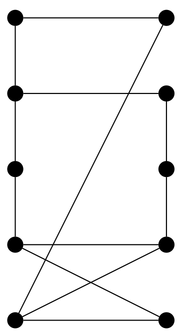
A



B



C



D