Technische Universität Wien
Institut für Computergraphik und Algorithmen
Arbeitsbereich für Algorithmen und Datenstrukturen

186.815 Algorithmen und Datenstrukturen 2 VU 3.0

Übungstest SS 2013

27. Juni 2013

Machen Sie die folgenden Angaben bitte in deutlicher Blockschrift:

Nachname: ______ Vorname: ______

Matrikelnummer: ______ Unterschrift: ______

Anzahl abgegebener Zusatzblätter: ______

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

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

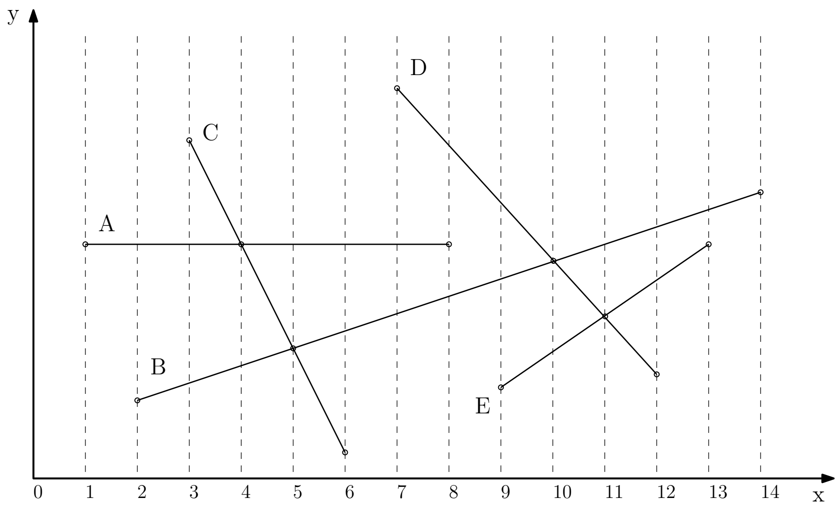
Die Verwendung von Taschenrechnern, Mobiltelefonen, PDAs, Digitalkameras, Skripten, Büchern, Mitschriften, Ausarbeitungen oder vergleichbaren Hilfsmitteln ist unzulässig.

	A1:	A2:	A3:	Summe:
Erreichbare Punkte:	17	17	16	50
Erreichte Punkte:				

Viel Erfolg!

Aufgabe 1.A: Schnitt von allgemeinen Liniensegmenten **(17 Punkte)**

Führen Sie den aus der Vorlesung bekannten *Scan-Line Algorithmus zum Schnitt von allgemeinen Liniensegmenten* aus.
(Die vertikalen Linien markieren die Zeitpunkte und dienen zur Orientierung.)



a) (4 Punkte)

Geben Sie dabei für die folgenden Zeitpunkte den Zustand der Scan-Line-Status Struktur an, nachdem das jeweilige Ereignis abgearbeitet wurde.

- Zeitpunkt 5:
- Zeitpunkt 12:

b) (8 Punkte)

Geben Sie die Zeitpunkte an, wann die Schnittpunkte in die Ereignisstruktur eingefügt werden.

- $A \cap C$:
- $B \cap C$:
- $D \cap E$:
- $B \cap D$:

c) (2 Punkte)

Geben Sie die Laufzeit für den *Scan-Line Algorithmus zum Schnitt von allgemeinen Liniensegmenten* möglichst genau in O-Notation an. Beschreiben Sie, wofür die verwendeten Variablen stehen.

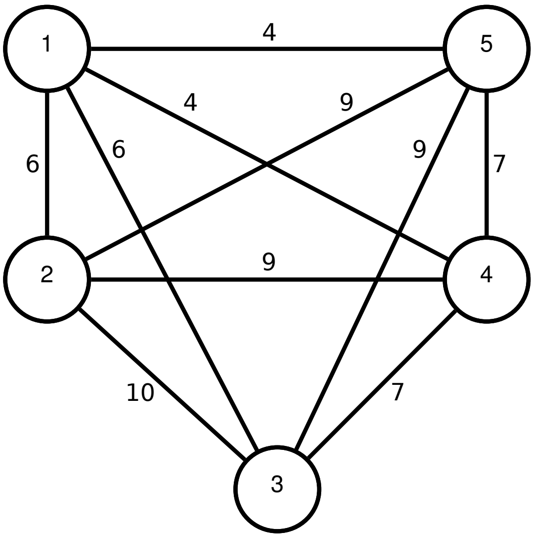
d) (3 Punkte)

Scan-Line-Status Struktur:

- Was wird darin wie gespeichert?
- Womit wird sie initialisiert?

Aufgabe 2.A: Approximationsalgorithmen **(17 Punkte)**

Gegeben ist der folgende, vollständige, ungerichtete Graph G von fünf Städten und den jeweiligen Distanzen. Wenden Sie die Christophides-Heuristik an, um eine approximative Lösung für das Traveling-Salesman-Problem in diesem Graphen zu finden.



a) (2 Punkte)

Markieren Sie in G jene Kanten, die den minimalen Spannbaum B bilden, und geben Sie die Menge W der Knoten mit ungeradem Grad in B an.

b) (3 Punkt)

Geben Sie das perfekte Matching M kleinsten Gewichts im von W induzierten Graphen an und zeichnen Sie den Graphen G' mit den Kanten $B \cup M$.

c) (2 Punkte)

Zeichnen Sie eine Euler-Tour F im Graphen G'. Beginnen Sie beim Knoten 1 und orientieren Sie F, indem Sie, falls mehrere Knoten in Frage kommen, immer zu dem Knoten mit dem kleinsten Index weitergehen.

d) (3 Punkte)

Zeichnen Sie die Ergebnistour der Christophides-Heuristik.

e) (2 Punkte)

Welche Gütegarantie besitzt die *Christophides-Heuristik* für das metrische Traveling-Salesman-Problem? Beschreiben Sie die Bedeutung dieses Faktors auch in Worten.

f) (5 Punkte)

Die *Christophides-Heuristik* kann auf nicht metrischen Graphen beliebig schlechte Ergebnisse liefern. Zeichnen Sie ein Beispiel mit 4 Knoten, das ein solches Worst-Case Szenario darstellt.

Aufgabe 3.A: Lokale Suche (16 Punkte)

Dem Arbeitsamt liegen n Stellenangebote und ebenso viele Arbeitssuchende vor. Sei B die Menge der unterschiedlichen Berufe, für die zumindest eine Stelle ausgeschrieben ist. Es gilt $B = \{1, 2, \ldots, m\},\ m \leq n$. Zusätzlich gibt $b_i,\ 1 \leq i \leq m$ an, wie viele Stellen von Beruf i zu besetzen sind.

Jeder Arbeitssuchende $j,\ 1 \leq j \leq n$ gibt seine Präferenz zu den zur Auswahl stehenden Berufen in Form einer sortierten Liste $A_j = \langle a_1, a_2, \ldots, a_m \rangle$ an. Das bedeutet, dass er Beruf a_1 gegenüber a_2 bevorzugt und den Beruf a_m am wenigsten gern ausüben kann und/oder möchte.

Das Ziel besteht darin, die Stellenangebote so zu vermitteln, dass jeder Arbeitssuchende eine Stelle bekommt und seine Präferenzen berücksichtigt werden, d.h., dass jeder mit seinem Job soweit wie möglich zufrieden ist.

a) (2 Punkte)

Geben Sie an, wie eine Kandidatenlösung in einer lokalen Suche sinnvoll repräsentiert bzw. gespeichert werden kann.

b) (2 Punkte)

Geben Sie eine Möglichkeit an, wie die Zielfunktion formuliert werden könnte.

c) (3 Punkte)

Definieren Sie eine sinnvolle Nachbarschaftsstruktur für eine lokale Suche durch die genaue Angabe erlaubter Züge. (Eine Beschreibung in einfachen Worten reicht.)

d) (2 Punkt)

Wie viele Nachbarlösungen besitzt eine Lösung in der von Ihnen definierten Nachbarschaftsstruktur?

e) (7 Punkte)

Der Pseudocode zur allgemeinen Lokalen Suche ist unten abgebildet. Entwerfen Sie für die Zeile 3: "Wähle $x' \in N(X)$" ein *konkretes* Stück Pseudocode, der die zuvor überlegte Nachbarschaftsstruktur umsetzt und der *best improvement* Strategie folgt.

Eingabe: eine Optimierungsaufgabe
Ausgabe: heuristische Lösung x

1: x = Ausgangslösung;
2: **wiederhole**
3:     Wähle $x' \in N(x)$; // leite eine Nachbarlösung ab
4:     **falls** x' besser als x **dann {**
5:         $x = x'$;
6:     **}**
7: **bis** Abbruchkriterium erfüllt

Technische Universität Wien
Institut für Computergraphik und Algorithmen
Arbeitsbereich für Algorithmen und Datenstrukturen

186.815 Algorithmen und Datenstrukturen 2 VU 3.0

Übungstest SS 2013

27. Juni 2013

Machen Sie die folgenden Angaben bitte in deutlicher Blockschrift:

Nachname: ____________ Vorname: ____________

Matrikelnummer: ____________ Unterschrift: ____________

Anzahl abgegebener Zusatzblätter: ____

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

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

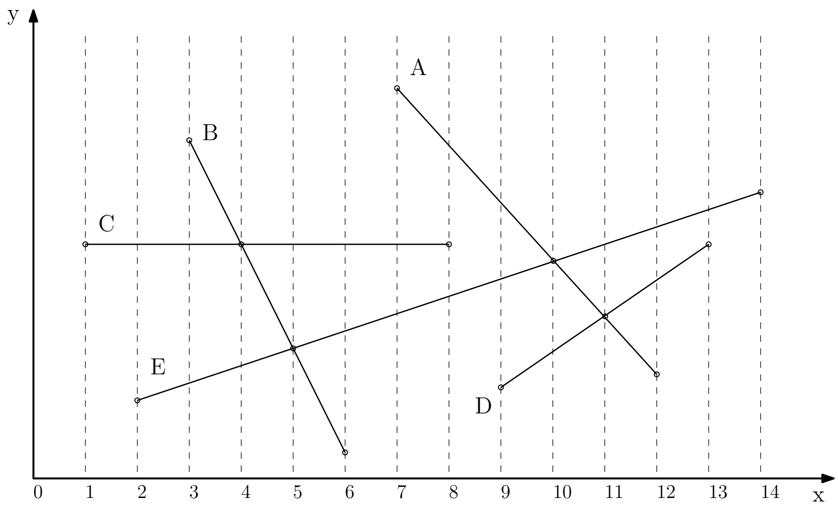
Die Verwendung von Taschenrechnern, Mobiltelefonen, PDAs, Digitalkameras, Skripten, Büchern, Mitschriften, Ausarbeitungen oder vergleichbaren Hilfsmitteln ist unzulässig.

	B1:	B2:	B3:	Summe:
Erreichbare Punkte:	17	17	16	50
Erreichte Punkte:				

Viel Glück!

Aufgabe 1.B: Schnitt von allgemeinen Liniensegmenten (17 Punkte)

Führen Sie den aus der Vorlesung bekannten *Scan-Line Algorithmus zum Schnitt von allgemeinen Liniensegmenten* aus.
(Die vertikalen Linien markieren die Zeitpunkte und dienen zur Orientierung.)



a) (4 Punkte)

Geben Sie dabei für die folgenden Zeitpunkte den Zustand der Scan-Line-Status Struktur an, nachdem das jeweilige Ereignis abgearbeitet wurde.

- Zeitpunkt 3:
- Zeitpunkt 10:

b) (8 Punkte)

Geben Sie die Zeitpunkte an, wann die Schnittpunkte in die Ereignisstruktur eingefügt werden.

- $B \cap E$:
- $A \cap D$:
- $A \cap E$:
- $B \cap C$:

c) (2 Punkte)

Geben Sie den Speicherplatzbedarf für den *Scan-Line Algorithmus zum Schnitt von allgemeinen Liniensegmenten* möglichst genau in O-Notation an. Beschreiben Sie, wofür die verwendeten Variablen stehen.

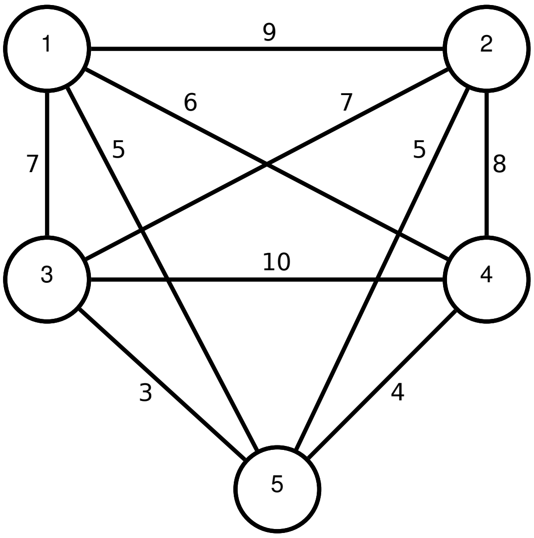
d) (3 Punkte)

Ereignisstruktur:

- Was wird darin wie gespeichert?
- Womit wird sie initialisiert?

Aufgabe 2.B: Approximationsalgorithmen **(17 Punkte)**

Gegeben ist der folgende, vollständige, ungerichtete Graph G von fünf Städten und den jeweiligen Distanzen. Wenden Sie die Christophides-Heuristik an, um eine approximative Lösung für das Traveling-Salesman-Problem in diesem Graphen zu finden.



a) (2 Punkte)

Markieren Sie in G jene Kanten, die den minimalen Spannbaum B bilden, und geben Sie die Menge W der Knoten mit ungeradem Grad in B an.

b) (3 Punkt)

Geben Sie das perfekte Matching M kleinsten Gewichts im von W induzierten Graphen an und zeichnen Sie den Graphen G' mit den Kanten $B \cup M$.

c) (2 Punkte)

Zeichnen Sie eine Euler-Tour F im Graphen G'. Beginnen Sie beim Knoten 1 und orientieren Sie F, indem Sie, falls mehrere Knoten in Frage kommen, immer zu dem Knoten mit dem kleinsten Index weitergehen.

d) (3 Punkte)

Zeichnen Sie die Ergebnistour der Christophides-Heuristik.

e) (2 Punkte)

Welche Gütegarantie besitzt die *Christophides-Heuristik* für das metrische Traveling-Salesman-Problem? Beschreiben Sie die Bedeutung dieses Faktors auch in Worten.

f) (5 Punkte)

Die *Christophides-Heuristik* kann auf nicht metrischen Graphen beliebig schlechte Ergebnisse liefern. Zeichnen Sie ein Beispiel mit 4 Knoten, das ein solches Worst-Case Szenario darstellt.

Aufgabe 3.B: Lokale Suche **(16 Punkte)**

Dem Arbeitsamt liegen n Stellenangebote und ebenso viele Arbeitssuchende vor. Sei B die Menge der unterschiedlichen Berufe, für die zumindest eine Stelle ausgeschrieben ist. Es gilt $B = \{1, 2, \ldots, m\},\ m \leq n$. Zusätzlich gibt $b_i,\ 1 \leq i \leq m$ an, wie viele Stellen von Beruf i zu besetzen sind.

Jeder Arbeitssuchende $j,\ 1 \leq j \leq n$ gibt seine Präferenz zu den zur Auswahl stehenden Berufen in Form einer sortierten Liste $A_j = \langle a_1, a_2, \ldots, a_m \rangle$ an. Das bedeutet, dass er Beruf a_1 gegenüber a_2 bevorzugt und den Beruf a_m am wenigsten gern ausüben kann und/oder möchte.

Das Ziel besteht darin, die Stellenangebote so zu vermitteln, dass jeder Arbeitssuchende eine Stelle bekommt und seine Präferenzen berücksichtigt werden, d.h., dass jeder mit seinem Job soweit wie möglich zufrieden ist.

a) (2 Punkte)

Geben Sie an, wie eine Kandidatenlösung in einer lokalen Suche sinnvoll repräsentiert bzw. gespeichert werden kann.

b) (2 Punkte)

Geben Sie eine Möglichkeit an, wie die Zielfunktion formuliert werden könnte.

c) (3 Punkte)

Definieren Sie eine sinnvolle Nachbarschaftsstruktur für eine lokale Suche durch die genaue Angabe erlaubter Züge. (Eine Beschreibung in einfachen Worten reicht.)

d) (2 Punkt)

Wie viele Nachbarlösungen besitzt eine Lösung in der von Ihnen definierten Nachbarschaftsstruktur?

e) (7 Punkte)

Der Pseudocode zur allgemeinen Lokalen Suche ist unten abgebildet. Entwerfen Sie für die Zeile 3: "Wähle $x' \in N(X)$" ein *konkretes* Stück Pseudocode, der die zuvor überlegte Nachbarschaftsstruktur umsetzt und der *best improvement* Strategie folgt.

Eingabe: eine Optimierungsaufgabe
Ausgabe: heuristische Lösung x

```
1: x = Ausgangslösung;
2: wiederhole
3:    Wähle x' ∈ N(x);  // leite eine Nachbarlösung ab
4:    falls x' besser als x dann {
5:       x = x';
6:    }
7: bis Abbruchkriterium erfüllt
```