

Klausur Softwaretechnik

16. März 2004

Hier das Namensschild aufkleben.

Aufg.1	Aufg.2	Aufg.3	Aufg.4	Aufg.5	Aufg.6	Σ	Note
/10	/6	/6	/10	/15	/13	/60	

Die Klausur besteht aus 20 Seiten und ist geheftet abzugeben.
Für die Vollständigkeit nicht gehefteter Klausuren wird keine
Verantwortung übernommen.

Viel Erfolg!
Bonne chance!
Buena suerte!
Good luck!
Buona fortuna!
Sok sikert!
Mult succes!
祝你成功
Удачи!
موفق باشيد!
حظ سعيد

Aufgabe 1 (Wissensfragen) – Für alle

(10 Punkte)

Hinweis: Falsche Kreuze geben negative Punkte, fehlende Kreuze, sowie fehlende oder falsche Freitext-Antworten bewirken nichts. Weniger als 0 Punkte können Sie mit dieser Aufgabe insgesamt nicht erreichen.

- a) Nennen Sie 3 Grundprinzipien zur Beherrschung der Komplexität bei der Entwicklung von Software.

- b) Kreuzen Sie nachfolgend alle Tätigkeiten an, die Sie während der Definitionsphase durchführen.

- Wirtschaftlichkeitsrechnung
- Pflichtenheft erstellen
- Durchführbarkeitsuntersuchung
- Anforderungen verabschieden

- c) Nachfolgend sind einige Basiskonzepte der Software-Entwicklung und verschiedene Sichten auf zu entwickelnde Software aufgeführt. Ordnen Sie Konzepte und Sichten einander korrekt zu.

- | | | |
|------------------------------|-------|-------------------------|
| 1. Funktionale Sicht | _____ | A. Klassendiagramm |
| 2. Datenorientierte Sicht | _____ | B. Entscheidungstabelle |
| 3. Objektorientierte Sicht | _____ | C. Pseudocode |
| 4. Algorithmische Sicht | _____ | D. Datenflußdiagramm |
| 5. Regelbasierte Sicht | _____ | E. Petrinetze |
| 6. Zustandsorientierte Sicht | _____ | F. Syntaxdiagramm |

- d) Ihre Aufgabe ist es, ein Sprachverarbeitungssystem mittels eines Datenflußdiagramms zu modellieren. Die erste Funktion ist das *Erkennen von Sprache*: *Erkannter Text* eines *Sprechers* wird in einer XML-Datenbank abgespeichert. Gleichzeitig wird die *Sprache aufgezeichnet* und in einer zweiten Datenbank für binäre *Audiodaten* abgespeichert. Neben dem Erkennen von Sprache bietet das System noch eine zweite Funktion, die *Sprachsynthese*: In der XML-Datenbank abgespeicherter Text kann synthetisiert und als Sprache einem *Hörer* ausgegeben werden.

Hinweis: Verwenden Sie für die einzelnen Teile des Datenflußdiagramms ähnliche Bezeichner wie im obigen Text kursiv hervorgehoben.

- e) Beantworten Sie die folgenden Fragen

Richtig/Falsch

- / Eine Transition in einem Petrinetz ist L3-lebendig für eine Markierung M wenn Sie für jede erreichbare Markierung L1-lebendig ist.
- / Eine Unterklasse muss alle abstrakten Methoden ihrer Oberklasse implementieren.
- / Als Regressionstest bezeichnet man das wiederholte Ausführen einer Testsuite nach Programmänderungen.
- / Eine inhaltlich vollständige Entscheidungstabelle ist immer auch formal vollständig.
- / Ein Cache heißt direkt-abgebildet, wenn er nur aus einem einzigen Satz besteht.

Aufgabe 2 (XML) – Für Informatiker

(6 Punkte)

Betrachten Sie das folgende XML-Instanzdokument. Vervollständigen Sie das gegebene XSD-Schema so, daß das XML-Instanzdokument für dieses Schema gültig ist.

```
<?xml version="1.0" encoding="utf-8" ?>
<swt:Brief xmlns:swt="http://www.ipd.uka.de/Tichy/SWT/"
  Absender="Alice" Betreff="Projektbesprechung">
  <swt:Absatz>Bob, wir wollen uns wegen des Projektes treffen.</swt:Absatz>
  <swt:Absatz>Kannst Du morgen mal vorbei kommen?</swt:Absatz>
  <swt:Absatz>Bis dann, Alice.</swt:Absatz>
</swt:Brief>
```

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema targetNamespace="http://www.ipd.uka.de/Tichy/SWT/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">
```

```
<xs:element name="Brief">
```

```
</xs:element>
```

```
</xs:schema>
```

Betrachten Sie das obige XML-Instanzdokument. Welche Werte haben die folgenden Informationseinheiten des XML Information Set?

a) Document_Information_Item::document_encoding_scheme _____

b) Document_Information_Item::standalone _____

c) Element_Information_Item::namespace_Name _____

d) Attribute_Information_Item::prefix _____

Aufgabe 3 (CVS, VSS) – Für Informatiker

(6 Punkte)

In dieser Aufgabe geht es um die gleichzeitige Verwendung des *Concurrent Versions Systems* durch mehrere Benutzer.

Ranjid ist Programmierer bei einer kleinen Outsourcing-Softwarefirma in Bangalore. Die folgende Funktion zur Fehlerbehandlung ist ein Ausschnitt aus einer größeren Quelltextdatei.

```
protected void Fehlerbehandlung(string strFehler)
{
}
```

In Ranjids Firma ist der CVS-Server `cvs.mydoomsoft.in` verfügbar. Die Dateien des aktuellen Projekts liegen in dem Repository `/projects/TBQ`. Ranjid verwendet das `pserver`-Protokoll und den Benutzernamen `ranjid`.

Wie muss Ranjid die Umgebungsvariable `CVSROOT` auf seinem Rechner setzen, um sich auf dem CVS-Server einloggen zu können?

```
d:\Ranjid\Prj>set CVSROOT=_____
_____
```

Ranjid erstellt nun wie folgt für die obige Quelltextdatei ein neues Modul.

```
d:\Ranjid\Prj>cvs import -m "Die 1. Version" Modul vtag rtag
```

Am nächsten Tag möchte Ranjid seine Arbeit fortsetzen. Geben Sie dazu das CVS-Kommando an, mit dem er seine Quelltextdatei auschecken kann.

```
d:\Ranjid\Prj>_____
```

In seinem lokalen Arbeitsbereich nimmt Ranjid nun die folgenden Ergänzungen an seinem Quelltext vor.

```
protected void Fehlerbehandlung(string strFehler)
{
    MessageBox.Show("Fehler: " + strFehler);
}
```

Kurz nach Ranjid hat auch sein Kollege **Yun** dieselbe Datei ausgecheckt und in seinem **lokalen Arbeitsbereich** die folgenden Änderungen vorgenommen:

```
protected void Fehlerbehandlung(string strFehler)
{
    Debug.WriteLine("Fehler: " + strFehler);
}
```

Mit welchem CVS-Kommando kann Ranjid Änderungen am Repository in seinem Arbeitsbereich sichtbar werden lassen?

d:\Ranjid\Prj>_____

Wie sieht die Quelltextdatei in Ranjids lokalem Arbeitsbereich nach dem Abgleich mit dem Repository aus?

```
protected void Fehlerbehandlung(string strFehler)
{
    _____
} _____
```

Im Laufe der Entwicklung hat Ranjids und Yuns Quelltextdatei die Version 4.2.1.3. Was sind dabei die Werte für...

- ... das Variantenlevel? _____
- ... das Releaselevel? _____
- ... die Variantennummer? _____
- ... die Releasenummer? _____

In Zukunft möchte Ranjid *Microsoft Visual Source Safe* verwenden. Er hat dazu aber noch zwei Fragen:

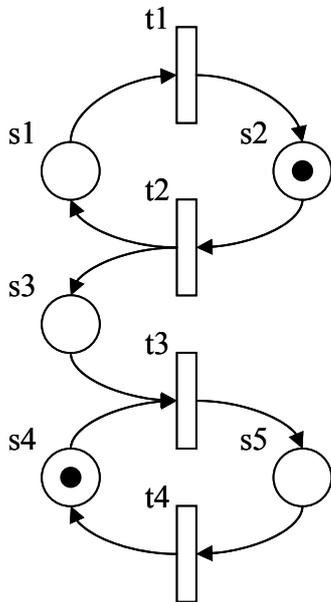
1. Warum kann man bei *Visual Source Safe* binäre Dateien nicht mehrfach auschecken bzw. warum würde das auch keinen Sinn machen?

2. Bei dem von *VSS* verwendeten Rückwärts-Delta Mechanismus zum Speichern von Änderungen an Dateien wird die _____ Version gespeichert und die Änderungen zu _____ gespeichert.

Aufgabe 2 (Petrietze) – Für Informationswirte

(6 Punkte)

Betrachten sie die folgenden Petrietze und beantworten Sie die zugehörigen Fragen.



Ist das Petrietz beschränkt (Ja/Nein)? _____

Geben Sie die Lebendigkeit (L0 – L4) des Petrietzes an:

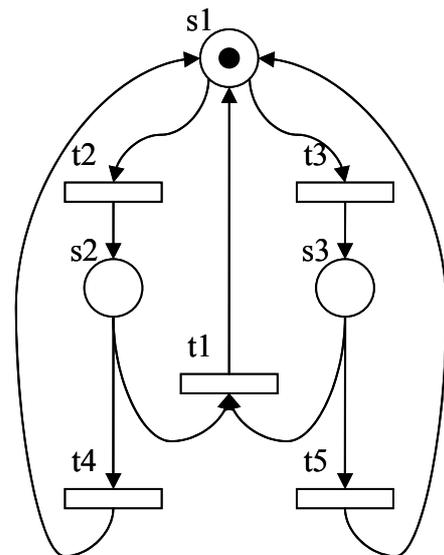
Das Netz ist _____

Geben Sie die Lebendigkeit (L0 – L4) der Transitionen t1, t2 und t4 an:

Die Transition t1 ist _____

Die Transition t2 ist _____

Die Transition t4 ist _____



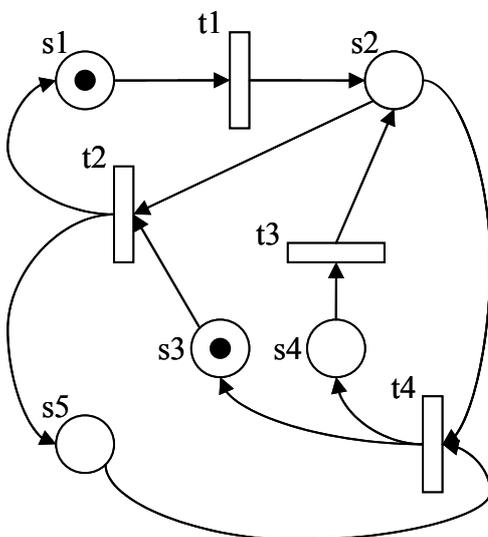
Geben Sie die Lebendigkeit (L0 – L4) der Transitionen t3 und t4 an:

Die Transition t3 ist _____

Die Transition t4 ist _____

Geben Sie die Lebendigkeit (L0 – L4) des Petrietzes an:

Das Netz ist _____



Aufgabe 3 (Optimierung) – Für Informationswirte (6 Punkte)

Die folgende Java-Klasse repräsentiere einen Ordner in einem Dateisystem.

```
public class FileFolder
{
    // Name des Ordners
    private String strFolderName;

    // Wenn dieser Ordner keinen Elternordner hat,
    // ist fileFolderParent gleich null
    private FileFolder fileFolderParent;

    // Für einen Ordner "/Verwaltung/Buchhaltung/"
    // gäbe diese Methode den lokalen Ordnernamen
    // "Buchhaltung" zurück
    public String getFolderName()
    {
        return strFolderName;
    }

    // Gibt eine Referenz auf den Eltern-Ordner zurück
    public FileFolder getParentFolder()
    {
        return fileFolderParent;
    }
}
```

Vervollständigen Sie die folgende **rechtsrekursive** Funktion `buildFilePathHelper` so, dass nach dem Ende aller rekursiven Aufrufe der komplette Pfad eines Ordners zurückgegeben wird. Für einen Ordner „Einkauf“ würde z.B. der Pfad „/Verwaltung/Buchhaltung/Einkauf“ zurückgegeben werden, wenn die Elternordner Buchhaltung und Verwaltung sind.

```
String buildFilePath(FileFolder f)
{
    String strPath = "";

    return buildFilePath(f, strPath);
}

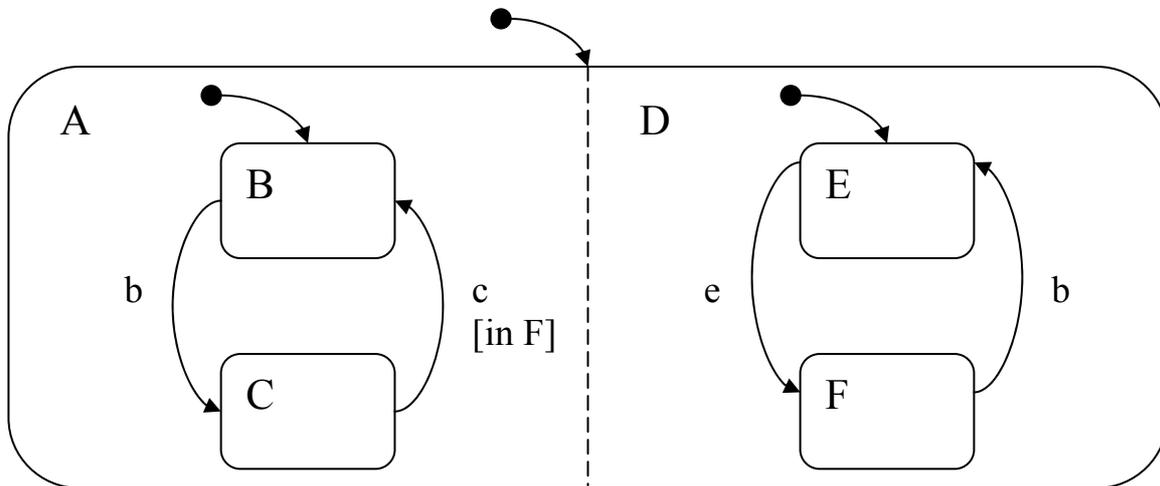
String buildFilePathHelper(FileFolder f, String strPath)
{
    if ( _____ )
    {
        _____
        _____
    }
    else
    {
        return _____;
    }
}
```

Wandeln Sie nun die obige rechtsrekursive Funktion `buildFilePathHelper` nach dem aus der Vorlesung bekannten Verfahren in eine iterative Form `buildFilePathHelper2` um.

```
string buildFilePathHelper2(FileFolder f, string strPath)
{
    while(_____)
    {
        _____
        _____
    }
    _____
}
```

Aufgabe 4 (Petrietze und Automaten) – Für alle (10 Punkte)

Betrachten Sie den folgenden Harelautomaten.



a) Wandeln Sie den obigen Harelautomaten in ein Petrinetz um. Modellieren Sie das Petrinetz so, daß wenn der Harelautomat sich in den Zuständen B, C, E und F befindet, sich jeweils genau eine Marke in den Stellen B, C, E und F des Petrinetzes befindet (z.B. entspräche der Zustand BE des Harelautomaten der Markierung $(1, 0, 1, 0)$ im Petrinetz).

Hinweis: Die Eingaben b, c und e sollen durch das Schalten von entsprechenden Transitionen im Petrinetz modelliert werden. Für eine Kante k in dem Harelautomaten können Sie wenn nötig mehrere Transitionen k_1, k_2 , usw. in dem Petrinetz einführen. Beachten Sie, daß in dem gegebenen Harelautomaten der Zustand BE nur einmal angenommen werden kann.



b) Zeigen Sie mittels eines Erreichbarkeitsgraphen, daß die für den Harelautomaten genannte Eigenschaft, daß der Zustand BE nur einmal angenommen werden kann, auch für Ihr Petrinetz gilt.

c) Wandeln Sie den Harelautomaten in einen äquivalenten Mealy-Automaten um.

Aufgabe 5 (Kontrollflußorientierter Strukturtest) – Für alle (15 Punkte)

Die Java Methode `sort` sortiert ein Feld von Ganzzahlen mit Hilfe des Bubblesort Algorithmus:

```
public int[] sort(int[] array) {
    int i, j, temp;

    i = array.size - 1;
    while (i >= 0) {
        j = 1;
        while (j <= i) {
            if (array[j-1] > array[j]) {
                temp = array[j-1];
                array[j-1] = array[j];
                array[j] = temp;
            }
            j++;
        }
        i--;
    }
    return array;
}
```

- Vervollständigen Sie die Schablone des Kontrollflussgraphen der Funktion `sort` auf der folgenden Seite. Verwenden Sie dabei für die `while`- und `if`-Anweisungen leere Knoten und numerieren Sie alle Knoten in der Reihenfolge, in der sie im Quelltext auftreten.
- Formulieren Sie für die gegebenen Testkriterien jeweils eine minimale Menge von Testdaten. Für den Boundary-Interior Test müssen Sie nur Testfälle für das Testen der inneren `while`-Schleife angeben. Geben Sie für alle Testfälle den im Kontrollflussgraphen durchlaufenen Pfad an.

Anweisungsüberdeckungstest:

Zweigüberdeckungstest:

Boundary-Interior Test (nur für die innere while-Schleife!):

1	<i>Start</i>
	Eingabe: <i>array</i>

10	<i>Stop</i>
	<i>return array</i>

- c) Schreiben Sie eine Methode in Java, bei der für den minimalen-mehrfachen Bedingungsüberdeckungstest mehr Testfälle erforderlich sind, als für den Zweigüberdeckungstest.

```
public _____ beispielC(_____ ) {  
  
    _____  
  
    _____  
  
    _____  
  
    _____  
  
}
```

Testfälle für minimalen-mehrfachen Bedingungsüberdeckungstest:

Testfälle für minimalen Zweigüberdeckungstest:

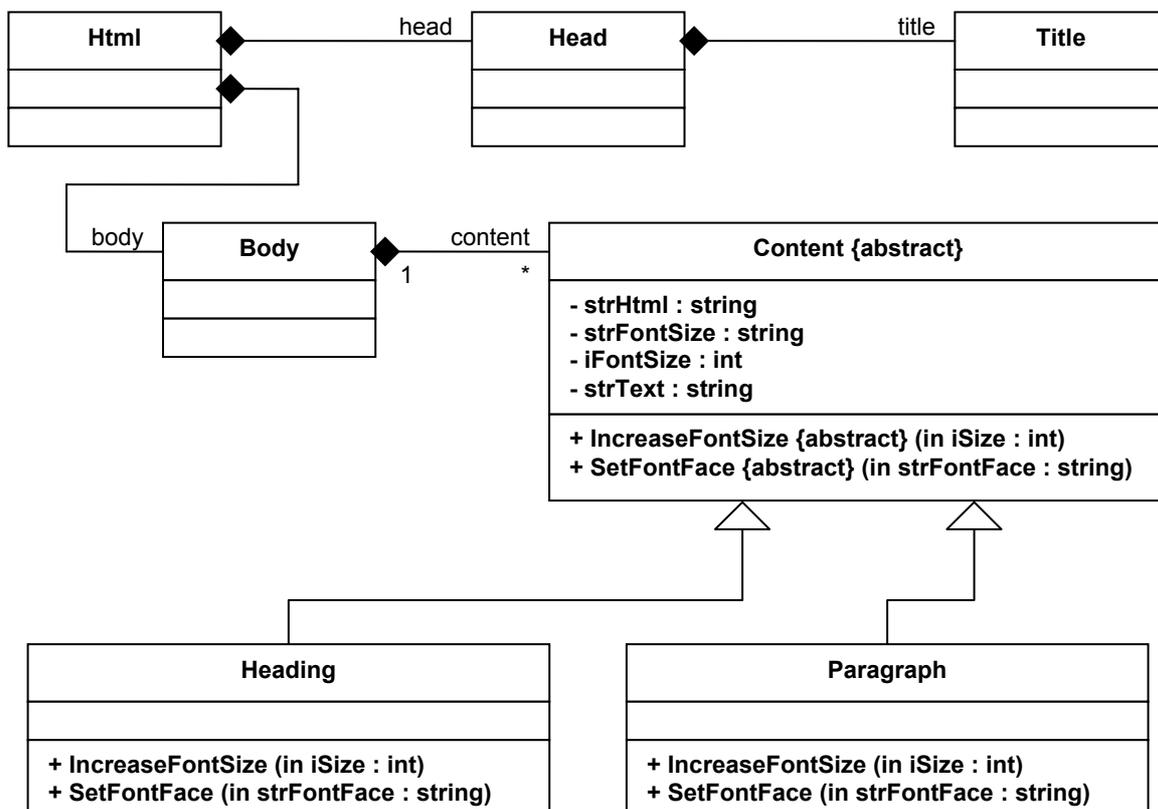
Aufgabe 6 (Entwurfsmuster)

(13 Punkte)

Das *W3C Document Object Model (DOM)* ist ein sprachunabhängiges Datenmodell und eine API für den programmatischen Zugriff auf z.B. (X)HTML. Solche Dokumente können mit dem DOM komplett als Baum von Objekten im Speicher dargestellt werden. Bestimmte im Dokument vorkommende Elemente werden dabei von einer Instanz der diesem Element entsprechenden Klasse repräsentiert.

Betrachten Sie das folgende HTML-Dokument. Für die Elemente `<html>`, `<head>`, `<title>`, `<body>`, `<p>`, `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` und `<h6>` gebe es in dem DOM jeweils die entsprechenden Klassen `Html`, `Head`, `Title`, `Body`, und `Paragraph`. Die Elemente `<h1>` bis `<h6>` seien alle durch die Klasse `Heading` repräsentiert.

```
<html>
  <head>
    <title>Softwaretechnik</title>
  </head>
  <body>
    <h1>
      <font face="Arial">Willkommen auf der SWT Seite</font>
    </h1>
    <p>
      <font face="Arial" size="4">Die Klausur ist am 16. März</font>
    </p>
  </body>
</html>
```



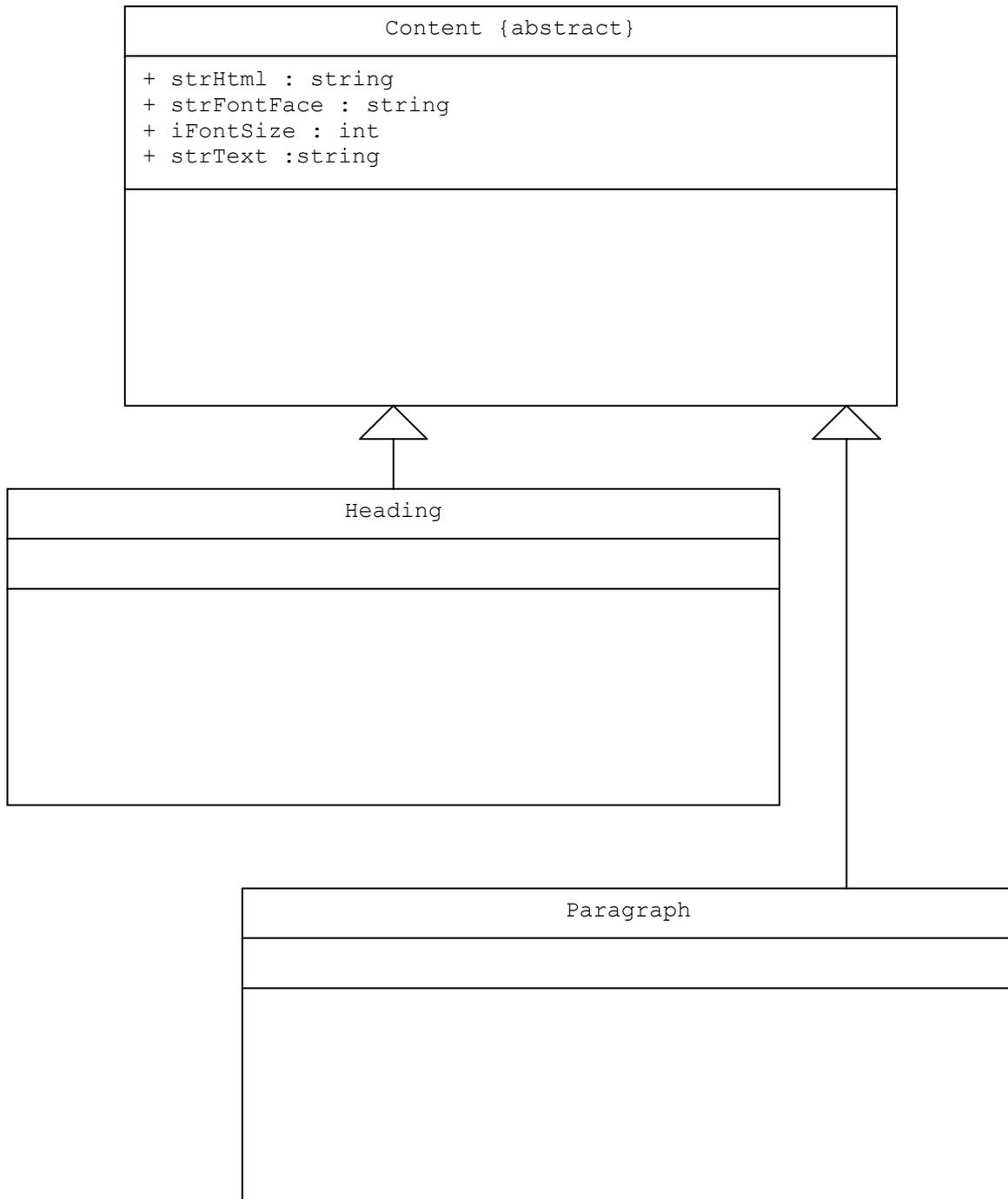
a) Erstellen Sie ein UML-Objektdiagramm, das das DOM für die oben gegebene HTML-Datei repräsentiert. Verwenden Sie dazu die in dem oben gegebenen UML-Klassendiagramm definierten Klassen.

Hinweise:

- Sie brauchen die Objekte nicht zu benennen sondern können einfach nur den entsprechenden Klassennamen als Bezeichner verwenden.
- Sie brauchen die Belegungen für die Instanzvariablen nicht anzugeben.

Die von der abstrakten Oberklasse `Content` abgeleiteten Klassen `Heading` und `Paragraph` weisen beide die Funktionen `IncreaseFontSize` und `SetFontFace` auf. In Zukunft werden jedoch noch weitere `Content`-Unterklassen hinzugefügt. Daher sollen gleiche Operationen in einer gemeinsamen Klasse gekapselt werden.

b) Verwenden Sie das Entwurfsmuster *Besucher*. Kapseln Sie die Operationen `IncreaseFontSize` und `SetFontFace` in zwei Besucherklassen. Vervollständigen Sie dazu auch das folgende UML-Klassendiagramm. Hinweis: Sie brauchen in die Klassen `Content`, `Heading` und `Paragraph` nur die neuen Funktionen einzutragen, die für das Besuchermuster relevant sind. Die Klassen `Html`, `Head`, `Title` und `Body` brauchen Sie nicht anzugeben. Zeichnen Sie das Klassendiagramm für die Besucherklassen auf die nächste Seite.



c) Vervollständigen Sie das UML-Sequenzdiagramm für folgende Interaktionen:

- Ein Klient `aClient` ändert die Schriftart einer Überschrift, indem er einen entsprechenden Besucher bei einer Instanz der Klasse `Head` vorbeischickt.
- Ein Klient `aClient` ändert die Schriftgröße eines Paragraphen, indem er einen entsprechenden Besucher bei einer Instanz der Klasse `Paragraph` vorbeischickt.

