

## Prüfung Softwareentwicklung I (IB)

---

Datum : 03.02.2014, 08:30 Uhr  
Bearbeitungszeit : 90 Minuten  
Prüfer : Prof. Dr. Oliver Braun  
Hilfsmittel : Keine  
Erreichbare Punkte : 90

---

Name: \_\_\_\_\_

Vorname: \_\_\_\_\_

Matrikelnummer: \_\_\_\_\_ Studiengruppe: \_\_\_\_\_

Hörsaal: \_\_\_\_\_ Platz Nr.: \_\_\_\_\_

Unterschrift: \_\_\_\_\_

Bitte kontrollieren Sie, ob Sie eine vollständige Angabe mit 5 Aufgaben auf 9 Seiten erhalten haben.

Aufgabe	1	2	3	4	5	Summe
max. Punkte	11	33	12	22	12	90

### Anmerkungen:

- Sie müssen als Antworten **keine** kompletten Programme schreiben, sondern nur den explizit verlangten Teil eines Programms.
- Schreiben Sie die Lösungen in die dafür vorgesehenen Kästchen. Sollte Ihnen der Platz dabei nicht reichen, benutzen Sie die Rückseite **und vermerken Sie das im dazugehörigen Kästchen!**

## Aufgabe 1 (11 Punkte)

Gegeben seien folgende Interfaces und Klassen:

```
interface X {
    int getX();
}
abstract class Z {
    abstract int getZ();
    int getY() {
        return getZ() + 5;
    }
}
class A extends Z implements X {
    int getZ() {
        return 13;
    }
    public int getX() {
        return getY();
    }
}
class B extends A implements X {
    public int getX() {
        return 100;
    }
    public int getY() {
        return super.getX();
    }
    int getZ() {
        return 7;
    }
}
class App {
    public static void main(String[] args) {
        A a = new A();
        System.out.println(a.getX()); //
        System.out.println(a.getZ()); //
        Z z = a;
        System.out.println(z.getZ()); //
        z = new B();
        System.out.println(z.getZ()); //
        System.out.println(a.getZ()); //
    }
}
```

Was wird beim Ausführen der Klasse App ausgegeben? Schreiben Sie Ihre Antworten direkt hinter die Zeile in der die Ausgabe angestossen wird (hinter die Kommentarzeichen).

## Aufgabe 2 (33 Punkte)

Gegeben sei das folgende Interface für eine Website:

```
interface Website {
    String baseUrl();
    String getPage(final String name);
    void addPage(final String name, final String content);
}
```

Implementieren Sie im Folgenden schrittweise eine Klasse `SimpleWebsite` die das Interface `Website` implementiert.

- (a) Geben Sie den Kopf der Klasse `SimpleWebsite` an:

(2)

- (b) Ein einfaches Website-Objekt speichert den Servernamen und das Protokoll als Zeichenketten in zwei Objektvariablen `servername` und `protocol`. Außerdem wird in zwei getrennten Arrays `pagenames` und `pagecontents` der Name bzw. der Inhalt einer Webpage gespeichert. Dabei findet sich der Name und der Inhalt einer Page unter dem gleichen Index. Beispiel: Der Inhalt der Seite deren Name *Impressum* den Index 5 im Array `pagenames` hat, findet sich im Array `pagecontents` beim Index 5. Ohne Einschränkung können Sie davon ausgehen, dass die Webpages paarweise verschiedene Namen haben.

(8)

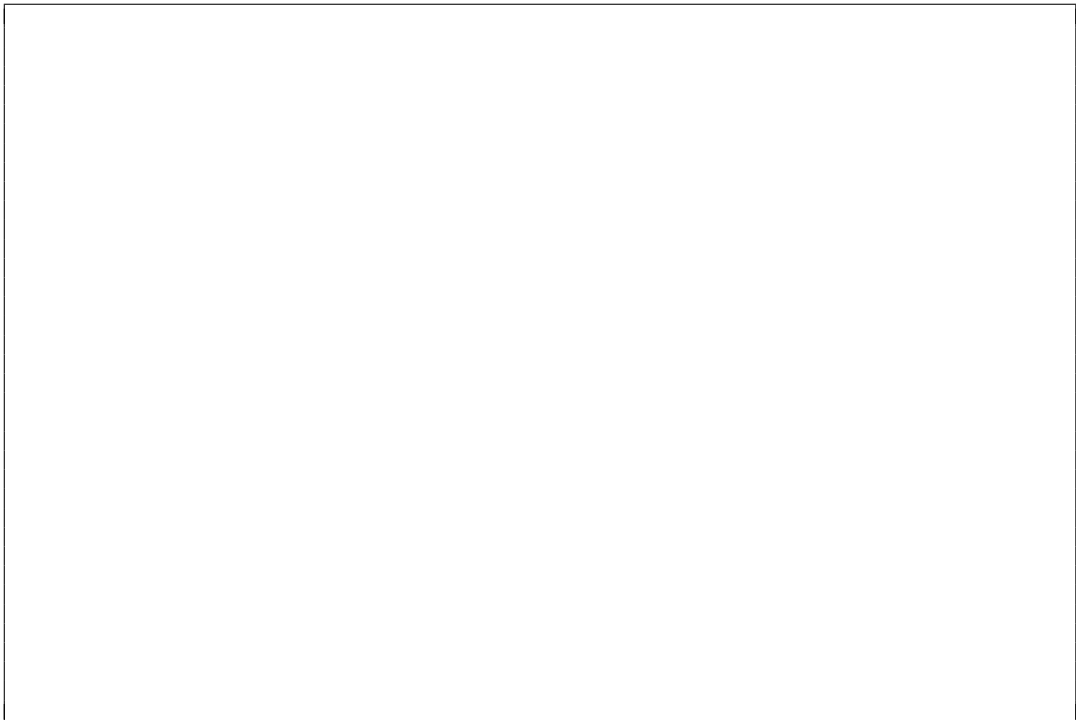
Darüber hinaus wird in einer Objektvariablen `nextpage` gespeichert, welcher Index als nächster befüllt werden kann, bzw. wieviele Seiten es bereits gibt.

Definieren Sie alle benötigten Objektvariablen.

- (c) Definieren Sie einen Custom-Konstruktor, mit drei Parametern: `servername`, `protocol` und `maxpages`, so dass beispielsweise mit `new SimpleWebsite("ob.cs.hm.edu", "http", 100)` ein Website-Objekt das maximal 100 Pages haben kann, erzeugt wird. (5)



- (d) Geben Sie eine Implementierung für die Methode `baseUrl` an, die die Basis-URL der einfachen Website zurück gibt, z.B. `http://ob.cs.hm.edu/`. (3)



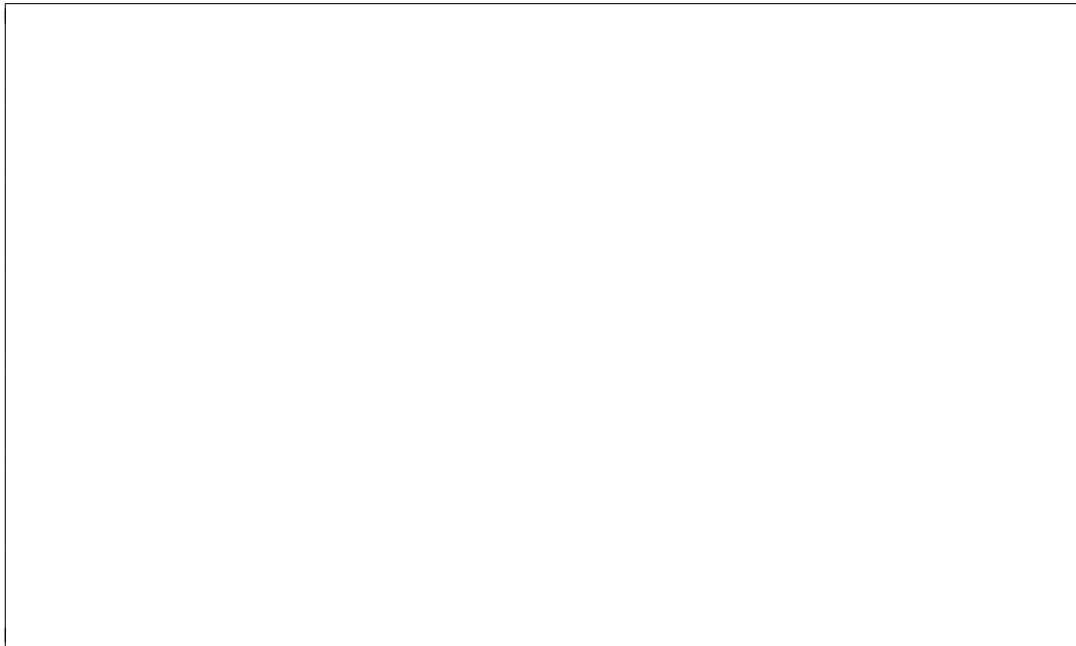
- (e) Geben Sie eine Implementierung für die Methode `getPage` an, die zu einem Seiten-Namen die URL gefolgt vom Inhalt zurück gibt, z.B. soll `getPage("about")` zurück geben: (9)

```
http://ob.cs.hm.edu/about
```

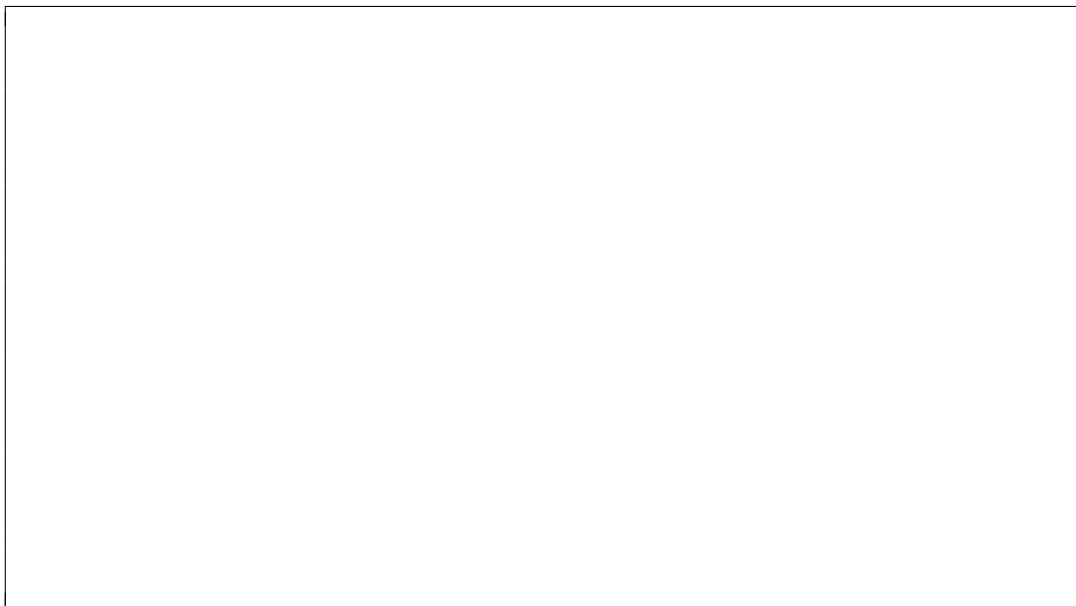
```
Blablabla...
```

Wobei „Blablabla...” der Inhalt sein soll.

Wird die Seite mit dem Namen nicht gefunden, soll statt dem **Inhalt** die Zeichenkette "404 - Page not found" zurück gegeben werden.



- (f) Geben Sie eine Implementierung für die Methode `addPage` an, die eine neue Seite in die beiden Arrays einfügt, aber nur dann, wenn noch Platz ist. (6)



### Aufgabe 3 (12 Punkte)

Implementieren Sie eine Enum `Protocol` mit den folgenden Eigenschaften:

- Die Enum `Protocol` soll folgende Werte enthalten: `HTTP`, `HTTPS`, `SMTP`, `POP3` und `IMAP`.
- Es gibt eine Objektvariable `portnumber`, in der die dazugehörige Portnummer steht.

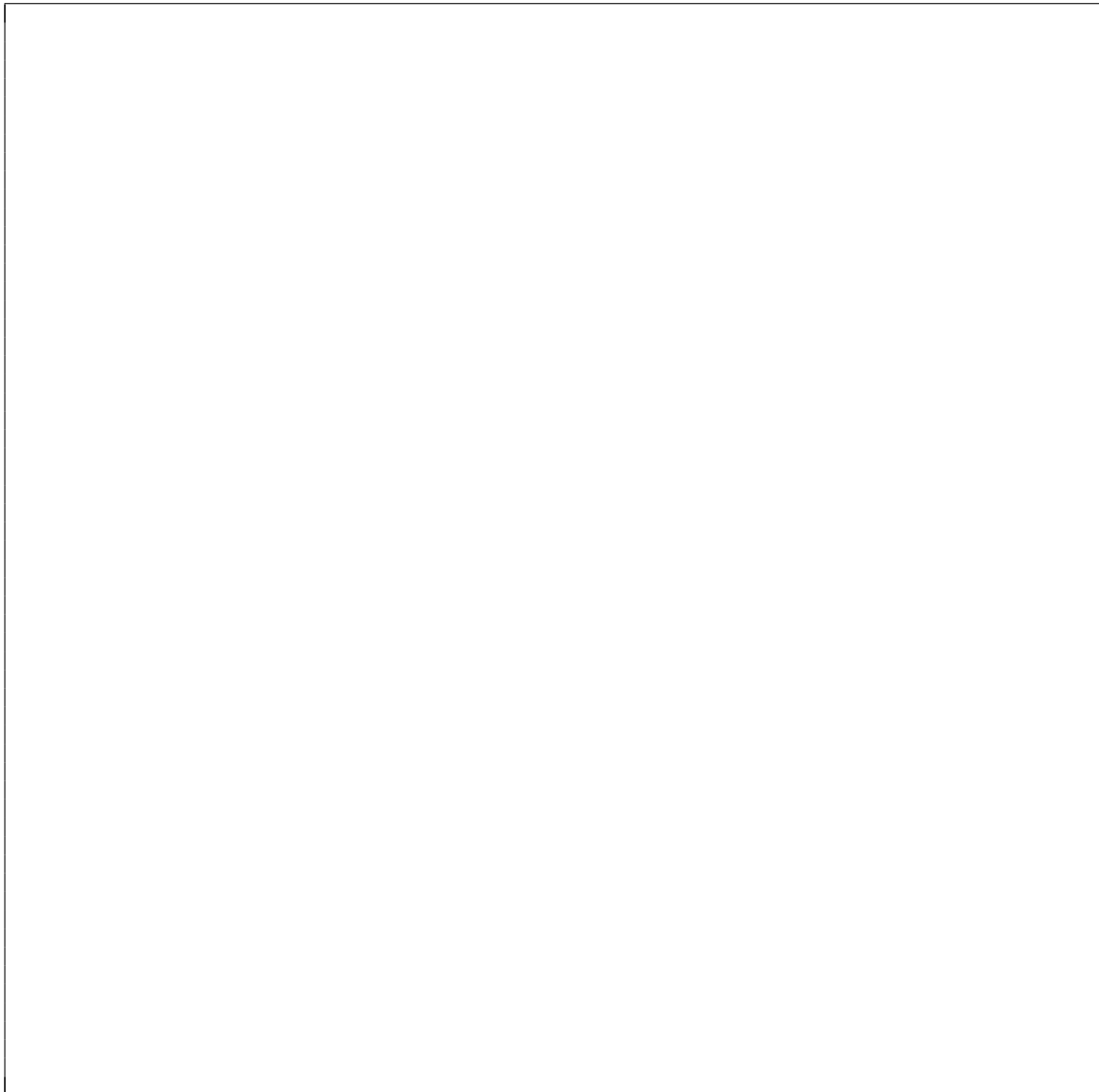
Anmerkung: Sie brauchen dann einen Konstruktor um es zu setzen. Die Portnummern lauten: 25: SMTP, 80: HTTP, 110: POP3, 143: IMAP, 443: HTTPS

- Außerdem gibt es die Methode

```
boolean isEmailProtocol()
```

die `true` genau dann zurück gibt, wenn das es sich bei dem Protokoll um eines handelt das zur Übertragung von E-Mails genutzt wird.

Anmerkung: Also POP3, IMAP und SMTP.



**Aufgabe 4 (22 Punkte)**

- (a) Definieren Sie eine abstracte Klasse **Vehicle** für Fahrzeuge. Die Klasse soll zwei abstrakte Methoden enthalten. Die Methode **hasEngine** soll zurück geben ob das Fahrzeug einen Motor hat und **numberOfWheels** die Anzahl der Räder der Fahrzeugs. (6)

- (b) Leiten Sie drei konkrete Klassen **Car**, **Motorcycle** und **Bike** für Autos, Motorräder und Fahrräder von **Vehicle** ab und implementieren Sie die beiden Methoden entsprechend unserer Realität. (8)

(c) Implementieren Sie eine **ausführbare** Klasse `VehicleApp`, die folgendes leistet: (8)

- Erzeugen von zwei Autos, einem Motorrad und zwei Fahrrädern.
- Einfügen der 5 Fahrzeuge in ein Array.  
Anmerkung: Sie können die ersten beiden Schritte (erzeugen und in Array einfügen) auch in einem Schritt implementieren.
- Durchlaufen des Arrays mit einer foreach-Schleife und Ausgabe von  
Das Fahrzeug hat `x` Räder und einen/keinen Motor.  
wobei für das `x` die Anzahl der Räder eingesetzt werden soll und statt  
einen/keinen je nach dem **entweder einen oder keinen** stehen soll.



### Aufgabe 5 (12 Punkte)

Der folgende Code ist fehlerhaft und wird nicht kompiliert:

```
1 interface X {
2     private int mu(int a);
3     X ma(Xyz);
4 }
5 class Y extends X {
6     private final int y;
7     private Y(final int a) {
8         a = 7;
9     }
10    Override
11    public int mu(int a) {
12        return a;
13    }
14    public X ma(X yz) {
15        return this;
16    }
17    X mkX() {
18        new Y(5);
19    }
20 }
```

Durch Änderung von 6 Zeilen in obigem Code, wird er kompilierbar. Geben Sie jeweils die Zeilennummer an und schreiben Sie dahinter wie die Zeile **korrekt** aussehen muss.

---

---

---

---

---

---

---