Technische Fachhochschule Berlin
Fachbereich Informatik
Medieninformatik
Luxemburgstr. 10
13353 Berlin

Prof. Dr. F.-M. Reisin

KLAUSUR

Medieninformatik PR1

12. Juli 2006

Tragen Sie bitte auf diesem Deckblatt und auf allen weiteren Blättern, die Sie während der Klausur verwenden, Ihren Namen und Ihre Matrikelnummer ein.
Verwenden Sie für Ihre Lösung jeweils das Blatt (einschließlich Rückseite), auf dem die Aufgabe steht. Sollte der Platz nicht reichen, verlangen Sie Zusatzblätter.
Insgesamt sind **155** Punkte erzielbar.
Für eine **1.0** genügen **140** Punkte.
Für eine **4.0** müssen **75** Punkte erzielt werden.
Die Bearbeitungszeit beträgt *105* Minuten.

Viel Erfolg!

Name, Vorname: _____

Matrikelnummer: _____

Wiederholungszahl: 0

Aufgabe	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	Σ
Maximal	10	10	10	10	15	20	20	20	30	10	155
Erreicht	9	8	6	8	15	20	20	16	23	7	132

1.3 Reisin

Aufgabe 1(10P)

Gegeben seien die folgenden Deklarationen:

```
int          iV      = 10;
final int    IK      = 1;
boolean      bV      = true;
boolean      BK      = false;
char         cV      = 'b';
char         CK      = '2';
char[]       cRei    = {'1','2','3','4'};
int[]        iRei    = {1,2,3,4};
```

Schreiben Sie für die folgenden Zuweisungen und Ausdrücke hinter jede Anweisung:

- **F**, wenn die Zuweisung inkorrekt ist, und
- den **Wert** der Variablen nach Ausführung der Zuweisung anderenfalls.

(10P)

Zuweisung	Wert
iRei[1] = 1234;	1234
bV = iRei[3] - iRei[2]- iRei[IK] == IK;	false
bV = !(CK == CK) && (bV != cV);	F
bV = ((CK <= CK) && true) ^ false;	true
bV = (bV && false) && (BK || true);	false
cV = cRei[0];	'1'
iRei[2] = cRei[3];	F
cV = (iV <= IK) ? '7' : '8';	'8'
IK = iRei[0] + iRei[3] * 3;	F
bV = (bV && false) && (BK || true);	false

✓ 4-3-2 == 1
✓ false &&
✓ (true && true) ^ false
✓ (true && false) && true
✓ (nil) '4' –
✓ 10 ≤ 1 → false
✓
✓ (true && false) && (false || true)
false && true

9

Aufgabe 2(10)

Deklarieren Sie bitte **vier Reihungsvariablen**

- **eine eindimensionale** **char-Reihung** **cR**
- **eine eindimensionale** **int-Reihung** **iR**
- **eine eindimensionale** **String-Reihung** **sR**
- **eine zweidimensionale** **int-Reihung** **iT**

cR und **iR** sollen jeweils mit einer **Länge 3 neu erzeugt werden.**

sR soll mit einer **initializer-list** erzeugt werden **und die Elemente "Alf", "BOB", "CIA" aufweisen.**

iT soll mit einer **initializer-list** Ihrer Wahl initialisiert werden.
Beide Dimensionen sollen die Länge **3 haben.** **(4)**

Geben Sie die Deklarationen an:

```
char[] cR = new char[3];
int[] iR = new int[3];
String[] sR = {"Alf", "BOB", "CIA"};
int[][] iT = {{1,2,3},{1,2,3},{1,2,3}};
```

4

Weisen Sie bitte den Komponenten von **iR** die folgenden Werte zu:

- die **Hälfte des Werts** einer beliebigen Komponente von iT in der **zweiten *Zeile***
- das **Dreifache des Werts** einer beliebigen Komponente von iT in der **zweiten *Spalte***
- den **mit 30 multiplizierten Wert** einer **bereits initialisierten iR-Komponente** **(3)**

Weisen Sie weiter

- der **dritten Komponente** von **cR** das **zweite Zeichen der ersten Komponente** von **sR** zu. **(1)**

Ersetzen Sie

- das **erste Zeichen** der **ersten Komponente** von **sR**
- **durch** das **erste Zeichen** der **letzten Komponente** von **sR.** **(2)**
-

```
iR[0] = iT[1][0] / 2;
iR[1] = iT[0][1] * 3;
iR[2] = iR[0] * 30;
cR[2] = sR[0].charAt(1);
sR[0].replace(sR[0].charAt(0), sR[2].charAt(0));
```

— — ✓ ✓ 2

4

8

Aufgabe 3 (10) Gegeben seien die folgenden Variablen und Zuweisungen

```
String str1 = "VIEL ERFOLG!";
String str2 = new String(str1.toLowerCase());
String str3 = new String ("VIEL ERFOLG!");
String str4 = "VIEL ERFOLG!";
String str5 = "SEMESTERBALL";
String str6 = str2;

boolean bV  =   false;
int iV      =   0;
char cV     =   ' ';
```

//Geben Sie bitte die Werte der einzelnen Variablen an

Zuweisungen	Werte
`iV = str1.indexOf('E');`	2 ✓
`iV = str2.lastIndexOf('l');`	3 9 –
`bV = str3 == "e";`	false ✓
`cV = str4.charAt(4);`	_ (Leerzeichen) ✓
`str5.replace('L', 'S');`	"SEMESTERBASS" ✓
`bV = str1 == str4;`	~~false~~ true ✓
`bV = (str1.length == str5.length);`	true ✓
`bV = (str1 != str3);`	~~true~~ ~~false~~ true –
`bV = (str1.equals(str3));`	false true –
`bV =  str1.equalsIgnoreCase(str2);`	true –

6

Aufgabe 4(10)
Gegeben seien die folgenden Deklarationen:
final int MIN = 15; final int MAX = 30;
int num = 20;

Unterstellen Sie, dass die Variable num, vor jeder Scheifenanweisung auf den int-Wert 20 wie oben gesetzt wird.

Geben Sie bitte fuer jede der Anweisungssequenzen wie jeweils gefordert entweder die Anzahl der Wiederholungen oder die Ausgaben an:

Anweisungssequenz	Wie oft? oder Ausgaben?

`while (num >= MIN)` 20 ≥ 15
`{`
`  pln (num);`
`  num = num - 1;`
`}`
	Wie oft? 6 ✓

`do`
`{`
`  num = num + 1;`
`  pln (num);`
`}while (num <= MIN);` 20 ≤ 15
	Wie oft? da 20 ≤ 15 : Schleife läuft bis zum maximalen int-Wert, Bereich springt ins Negative -> dann ist wieder ≤ Min —

`for (int iX=1; iX <= 5; iX++)` 1≤5
`  pln('*');`
`  for (int iY=1; iY <= 5; iY++)` 1≤5
`  pln('0');`
	Ausgaben
	*0000
	*0000
	*0000
	*0000
	*0000 ✓

`for (int iX=1; iX <= 5; iX++)` 1≤5
`{ pln('*');`
`  for (int iY=1; iY <= 5; iY++)` 1≤5
`  pln('+');`
`}`
	Wie oft '*' ? 5 ✓
	Wie oft '+' ? 25 ✓

*
+
+
+
+
+

8

Aufgabe 5 (15)

Der Aufruf der Funktion `F.f()`, führt im Rumpf von `testF()` zu unterschiedlichen Effekten.
Sie resultieren aus dem Unterschied zwischen **Referenz-Typ-** und **Primitiv-Typ-Prametern**.
Geben Sie bitte an den mit '?' markierten Tabellenzeilen präzise den Wert an, der ausgegeben wird.

`public class CallByVal_CallByRef {`		
`  public static void testF() {`		
`    char cV = 'e';`		
`    boolean bV = true;`		
`    Integer[] intR1 = {new Integer(2), new Integer (4)};`		
`    Integer[] intR2 = {new Integer(13)};`		
`    pln(intR1);`	@10b62c9	
`    pln(intR1[0].intValue());`	? 2	✓
`    pln(intR1[1].intValue());`	? 4	✓
`    pln(intR2);`	@82ba41	
`    pln(intR2[0].intValue());`	? 13	✓
`    pln(bV = cV != 'e');`	? false	✓
`    F.f(intR1, intR2, cV, bV);`		
`    pln(intR1);`	? @10b62c9	✓
`    pln(intR1[0].intValue());`	? 65	✓
`    pln(intR1[1].intValue());`	? 4	✓
`    pln(intR2);`	? @82ba41	✓
`    pln(intR2[0].intValue());`	? 13	✓
`    pln(bV = cV != 'e');`	? false	✓
`  }`		
`}`		

`class F{`		
`public static void f (Integer[] intR1, Integer[] intR2,`		
`                      char cP, boolean bP)`		
`  {`		
`    Integer[] locIntR = {new Integer(7),new Integer(9)};`		
`    intR2 = locIntR;`		
`    intR1[0] = new Integer(65);`		
`    cP = (char) intR1[0].intValue();`		
`    pln(locIntR);`	@923e30	
`    pln(intR1);`	? @10b62c9	✓
`    pln(intR1[0].intValue()*intR2[1].intValue());`	? 585	✓
`    pln(intR2);`	? @923e30	✓
`    pln(intR2[0].intValue());`	? 7	✓
`    pln(cP);`	? 65 'A'	✓
`    pln(bP=cP=='e');`	? false	✓
`  }`		
`}`		

15

Aufgabe 7(20P)

Spezifizieren Sie bitte die Funktion `int geradeZ(int[] iR)`

Die Funktion `geradeZ()`

liefere **die Anzahl** der Komponenten von **iR**, deren **int-Wert eine gerade Zahl** ist.

```
int geradeZ(int[] iR)
{ int count=0;
  if((iR!=null) return -1;        <-- if (iR.length!=0)
    for (int i=0; i<iR.length; i++)
      if(iR[i]%2==0) count++;

  return count;
}
```

20

Aufgabe 6(20P)

Spezifizieren Sie bitte die **boolean**-Funktion

`boolean gleichGbR(char[] cR1, char[] cR2)`

Die Funktion `boolean gleichGbR()` (zu lesen gleiche Großbuchstaben-Reihungen) liefert

`true`,

wenn beide Reihungen **gleich lang** sind und an **jeder Indexstelle i denselben Großbuchstaben** aufweisen und

`false`

2anderenfalls.

true: cR1.length == cR2.length
cR1[i] == cR2[i]
-> groß

false:

```
boolean gleichGbR(char[] cR1, char[] cR2)
{ if(cR1==null || cR2==null) return false;
  if(cR1.length!=cR2.length) return false;
  else for(int i=0; i<cR1.length; i++)
         { if((cR1[i]<'A'||cR1[i]>'Z') return false;
           if (cR1[i]!=cR2[i]) return false;
         }
  return true;
}
```

✓ ✓ ✓

Sehr gut!

20

Aufgabe 8 (20 Punkte)
Vereinbaren Sie bitte eine char[] -Funktion
char[] ohneRST(char[] cR)
Die **char[]** -Funktion **ohneRST()** wandelt die Zeichen 'R', 'S','T', 'r', 's', 't' der Zeichenreihe cR in ein 'X' um. Die übrigen Zeichen der Zeichenkette bleiben unverändert.

```
ohneRST ("Rotstift!")          liefert  "XoXXXifX!"
ohneRST ("RStuVw")                liefert  "XXXuVw"
ohneRST ("rSO8tt;= TAT?")         liefert  "XXO8XX;=XAX?"
```

neue Reihung zurückgeben?

```
char[] ohneRST(char[] cR)
{  if(cR==null) return null;
   char[] copy = new char[cR.length];
   for(int i=0; i<copy.length; i++)
      if((copy[i]>='R' && copy[i]<='T') || (copy[i]>='r' && copy[i]<='t'))
         copy[i]='X';
      else copy[i]=cR[i];
   return copy;
}
```

16

Aufgabe 9 (30 P)

Die Klasse `Gen` ist ein Bauplan für Gen-Objekte. Jedes Gen-Objekt habe drei Attribute:

`int[] iR, char cV und String str;` //iR für int-Reihung

9. 1 Vereinbaren Sie bitte: **drei Konstruktoren:**

- einen **parameterlosen Konstruktor,**
- einen mit **1 Parameter, vom Typ String**
- und einen mit **3 Parametern.**

Der parameterlose Konstruktor dient der Initialisierung von:

- `iR` mit einem **zwei-Elemente-Initializer**
  Beide `iR`-Elemente haben **denselben Wert im Bereich 0 .. 10**
- `cV` sei mit dem char-Wert des ersten Elements von iR und
- `str` sei mit `"HOT"` initialisiert.

Die Konstruktor-Rümpfe sind durch Aufruf bereits definierter Konstruktoren möglichst minimal zu programmieren. **(6)**

```
class Gen
{
    protected int[]  iR;
    protected char   cV;
    protected String str;

    Gen()
    {
      iR = {0,0};                    this("HOT", (char)iR[0], {0,0});
      cV = (char)iR[0];                                  T geht nicht
      str = "HOT";
    }

    Gen (String str)
    {
      this.str = str;

    }

    Gen (String str, char cV, int[] iR)
    {
     this.str = str;          (this(str),
     this.cV = cV;
     this.iR = iR;

    }
```

3

3

2

3

8

23

9.2 Vereinbaren Sie bitte außerdem die folgenden Objektfunktion:

```
public boolean istInIR(int zahl){//liefert true, falls zahl in iR vorkommt, false sonst
(8)  if(iR!=null)
{  for(int i=0;i<iR.length;i++)
     if(zahl==iR[i]) return true;

   return false;
}
```

6

9. 3 Vereinbaren Sie eine **Extensionklasse zu Gen** mit dem Namen **Spez**
Spez habe das Attribut:

- **char[] cR;**

Vereinbaren Sie bitte zwei Konstruktoren:

- einen mit **1 Parameter vom Typ char[]**
- einen mit **2 Parametern, einen vom Typ String und einen vom Typ char[]** **(6)**

Vereinbaren Sie weiter eine **Funktion:**

```
boolean strGlcR()
//liefert true, falls die Länge des char[]-Attributs cR
//und die Länge des String-Attributs str gleich sind.        ((10)
```

```
class Spez extends Gen                    //Klassenkopf(1) (bitte angeben)        1
{
 private char[] cR;                       //Objektattribut(1) (bitte angeben)     1
```

//Konstruktoren(4)

```
Spez(char[] cR)                                                    super();  1/2
{ this.cR = cR; }

Spez(String str1, char[] cR)
{ this(cR);   super(str);
  str = str1;                                                              2
}
```

//Funktion(4) ← jedoch bedarf auch private

```
public boolean strGlcR()
{ if(cR==null)return false;
  if(cR.length==str.length())return true;
  return false;
}
```

4

Aufgabe10 (10)

Gegeben seien die folgenden Klassen:


```
class A {};
class B extends A {};
class D extends B {};
class E extends B {};
```

Deklariert seien weiter die Variablen a, b, d, e wie folgt:

```
A a;
B b = new B();
D d = new D();
E e = new E();
```

Geben Sie bitte in den angegebenen Kommentarzeilen an, ob die Anweisung davor zulässig (Z) oder unzulässig (U) ist, indem Sie um ZUTREFFENDES einkreisen oder es unterstreichen. Begründen Sie Ihre Antwort knapp.

`b = d;`	Z / U	BEGRUENDUNG:	widening cast
`b = e;`	Z / U	BEGRUENDUNG:	widening cast
`d = e;`	Z / U	BEGRUENDUNG:	nicht möglich, da auf gleicher Hierarchieebene
`e = d;`	Z / U	BEGRUENDUNG:	— " —
`d = b;`	Z / U	BEGRUENDUNG:	narrowing cast → nur mit cast möglich → (d)
`a = e;`	Z / U	BEGRUENDUNG:	widening cast
`a = new B();`	Z / U	BEGRUENDUNG:	class A erbt nicht von B → hat dessen Attribute, Funktionen etc. nicht
`b = new D();`	Z / U	BEGRUENDUNG:	— " — nur mit class B und D
`e = new A();`	Z / U	BEGRUENDUNG:	class E erbt alle Elemente von A
`b = null;`	Z / U	BEGRUENDUNG:	b ist Referenztyp

[redacted]

7