Frühjahr 2023
252-0027 – Einführung in die Programmierung

Departement Informatik
ETH Zürich

25. August 2023 – Schriftliche Prüfung

ID: 3144

Nachname:_____ Vorname:_____

Legi-Nummer:_____–_____–_____

Sie dürfen diese Prüfung oder die Aufgaben für die Programmierprüfung erst öffnen nachdem die Aufsicht die Prüfung gestartet hat. **Wenn Sie diese Dokumente vorher öffnen gilt dies als Täuschungsversuch.**

Mit Ihrer Unterschrift bestätigen Sie, dass Sie die hier aufgeführte Person sind, Sie die Hinweise zur Kenntnis genommen haben, Sie die Aufgaben selbständig gelöst haben, Sie Ihre eigene Lösung abgeben, Sie keine Kopie der Prüfung mitnehmen, und dass Sie alle technischen Probleme, gesundheitlichen Probleme (die Ihre Leistungen in dieser Prüfung beeinträchtigten) und etwaige störende äussere Einflüsse gemeldet haben bzw. wissen, dass Sie diese melden sollen.

Unterschrift:_____

Hinweise

1. Öffnen Sie diese Prüfung und die Aufgabenstellung für die Programmierprüfung erst, wenn die Aufsicht den Beginn der Prüfung bekannt gibt.
2. Schreiben Sie zuerst Ihren Namen und Ihre Legi-Nummer auf das lose Kontrolblatt dieser Prüfung. Überprüfen Sie, dass die Zahl auf dem Kontrolblatt mit der Zahl auf der 1. Seite der schriftlichen Prüfung übereinstimmt.
3. Vergessen Sie nicht Ihre Unterschrift nach dem Ende der schriftlichen Prüfung.
4. Der schriftliche Teil der Prüfung dauert 40 Minuten. Falls Sie sich durch irgendjemanden oder irgendetwas gestört fühlen, melden Sie dies sofort einer Aufsichtsperson.
5. Die schriftliche Prüfung hat 12 Seiten (inkl. Kontrolblatt). Vergewissern Sie sich, dass Ihr Exemplar vollständig ist.
6. In dieser Prüfung gibt es 40 Punkte. Benutzen Sie die Anzahl der Punkte als *Hinweis*, wie Sie Ihre Zeit einteilen können. Fehler bei der Bearbeitung einer Aufgabe haben keinen Einfluss auf die Punkte, die Sie für andere Aufgaben erhalten. Sie können die Aufgaben in beliebiger Reihenfolge lösen.
7. Lesen Sie die Aufgabenstellungen genau durch.
8. Tragen Sie Ihre Antwort(en) direkt in die Prüfungsbögen ein. Falls Sie mehr Platz brauchen, ist Ihre Antwort wahrscheinlich zu lang.
9. *Benutzen Sie einen Stift (blau oder schwarz), der nicht ausradiert werden kann.* Bitte schreiben Sie deutlich und leserlich! Wenn Sie etwas durchstreichen wollen, so machen Sie dies bitte klar und eindeutig.
10. Trennen Sie nicht die zusammengeheftete Prüfung. Mit losen Blättern riskieren Sie, ein Blatt zu verlieren.
11. Es ist wichtig, dass Ihre Antworten die Aufgaben klar und *unzweideutig* behandeln. Die Klarheit der Antworten beeinflusst Ihre Note. Eine fehlende Antwort wird als falsche Antwort bewertet.

    Wenn Sie Annahmen (über die in den Aufgaben aufgeführten hinaus) treffen, so geben Sie diese bitte an.
12. Wenn Sie zur Toilette müssen, melden Sie dies einer Aufsichtsperson durch Handzeichen. Es darf zur gleichen Zeit immer nur eine Person zur Toilette.
13. Wir beantworten keine inhaltlichen Fragen während der Prüfung. Kommentare zur Aufgabe schreiben Sie bitte auf das Aufgabenblatt.
14. Wenn die Aufsicht diesen Teil der Prüfung beendet, schliessen Sie bitte die Prüfung und schreiben nicht mehr in die Prüfung. **Weiterarbeiten über die erlaubte Zeit gilt als Täuschungsversuch.** Bitte unterschreiben Sie die Prüfung auf der Vorderseite und legen Sie die Prüfung mit Ihrer Legitimationskarte ("Legi") gut sichtbar auf Ihren Arbeitstisch.
15. Wir sammeln die Prüfung ein. Wichtig: stellen Sie unbedingt selbst sicher, dass Ihre Prüfung von einer Aufsichtsperson eingezogen wird. Stecken Sie keine Prüfung (auch keine leere) ein.

ID: 3144

Aufgabe	Wert	Punkte		Aufgabe	Wert	Punkte	
1	4			5	11		
2	4			6	8		
3	4			7	4		
4	5			–			
Σ				Übertrag	––		
				Total	40		

Aufgabe 1 (4)

Gegeben sind Zuweisungen in einer Java Methode. `myVar` ist eine initialisierte `int` Variable in dieser Methode. `even` ist eine `boolean` Variable, die mit `false` initialisiert wurde. Welche dieser Anweisungen setzen (oder setzt) die Variable `even` zu `true` wenn – und nur dann wenn – `myVar` eine gerade Zahl ist. (Zur Erinnerung: eine Zahl ist gerade wenn sie ohne Rest durch 2 teilbar ist.) Markieren Sie diese Anweisung(en) durch ein "x" in der letzten Spalte.

```
even = myVar/2.0 == (double)(myVar/2)            [  ]
       ;

even = myVar % 2 == 0                            [  ]
       ;

even = myVar // 2 == 0                           [  ]
       ;

even = myVar % 2 == myVar / 2                    [  ]
       ;

even = (double) myVar % 2 == 0                   [  ]
       ;

even = (double) myVar / 2 == 0                   [  ]
       ;

even = (double) myVar / 2 == myVar / 2           [  ]
       ;

even = (double) myVar / 2 == myVar / 2.0         [  ]
       ;

even = (myVar/2) * 2 == myVar                    [  ]
       ;

Keine dieser Anweisungen setzt even to true      [  ]
```

Aufgabe 2 (4)

Gegeben sei die folgende Klasse, die die Referenzvariable `this` und den Konstruktur `this()` verwendet. Markieren Sie durch Ankreuzen (oder Umkreisen) die Verwendung von `this` als

- `richtig (R)` (wenn korrekt und zur Vermeidung von Nameskonflikten nötig, d. h., ein Weglassen ändert die Bedeutung des Ausdrucks/Programms),
- `extra (E)` (wenn korrekt aber nicht zwinged nötig, d. h., ein Weglassen ändert nicht die Bedeutung des Ausdrucks/Programms),
- `unerlaubt (U)` (wenn der Gebrauch von `this` oder `this()` an dieser Stelle nicht erlaubt ist), *oder als*
- `fehlerhaft (F)` (wenn der Gebrauch zwar syntaktisch korrekt ist, aber zur Laufzeit ein Fehler auftreten kann).

```
class R {

    static int v;

    int x;

    public R() {

        this.x = 1;                                R       E       U       F

        this.v = 1;                                R       E       U       F
    }

    public R(int x) {

        this();                                    R       E       U       F

        this.x = x;                                R       E       U       F
    }


    public R(int x, int y) {
        v = x * y;

        x = v / this.x;                            R       E       U       F
    }

    public R(int x, int y, int z) {
        x = y + 1;

        v = this.x * x;                            R       E       U       F
    }

    public static String getString() {

        return "v = " + this.v +                    R       E       U       F

            " x = " + this.x;                       R       E       U       F
    }

}
```

Aufgabe 3 (4)

Was druckt die Methode `main` der folgenden Klasse `Beispiel`?

```
class Beispiel {
    public static void main(String[] args) {

        int[] a = new int[6];

        a[0] = 5;
        a[5] = 5;

        int sum = 0;

        for (int k : a) {
           sum += a[k];
        }

        System.out.println("sum␣=␣" + sum);    // sum = _____________________

    }
}
```

Aufgabe 4 (5)

Gegeben ist die Methode `findLargestSmaller(int[] al, int value)`, die in einem *sortierten nicht-leeren* Array von `int` Werten den grössten Wert findet, der strikt kleiner als `value` ist. `value` muss strikt grösser als `al[0]` sein.

Was ist die Invariante für den Loop in der Methode `findLargestSmaller`? Wenn die Elemente `a[0]` .... `a[K]` des Arrays a sortiert sind, dann können Sie das mit `Sorted(a, 0, K)` abkürzen.

```
   static int findLargestSmaller(int[] al, int value) {

 // Precondition: Sorted(al, 0, al.length-1) && al.length >= 1 && value > al[0]

       int candidate  = al[0];
       int next = 1;

 // Loop Invariante:


       while (next < al.length && value > al[next] ) {
           candidate = al[next];
           next++;
       }

// Postcondition: Sorted(al, 0, al.length-1) && al.length >= 1 &&
((next < al.length && value <= al[next] && candidate == al[next-1]) ||
(next == al.length && candidate == al[al.length-1] && value > candidate))


       return candidate;
}
```

1. Geben Sie die Loop Invariante an. (Sie können, wenn Sie es für nötig halten, die Precondition oben erweitern.)

   Loop Invariante: _____

   _____

2. Durch Testen wurde demonstriert, dass diese Methode korrekt für Werte `value` ist, für die `al[0] < value <= al[al.length-1]` gilt. Können Sie diese Methode auch für Werte `value > al[al.length-1]` verwenden? Markieren Sie Ihre Antwort.

   (a) **Ja**
   Diese Methode kann auch für Werte `value > al[al.length-1]` verwendet werden.

   (b) **Nein**
   Diese Methode kann `nicht` für Werte `value > al[al.length-1]` verwendet werden.

Aufgabe 5 (11)

Gegeben seien diese Klassen und Interfaces, zusammen mit der Klasse A6. Alle Klassen und Interfaces sind im default Package.

```
interface I {
    public void g();
}

class M implements I {
    String s = "1";

    public void g() {
        A6.print("M␣g");
    }
    public void h(String s){
        A6.print("M␣" + s);
    }

}

class N extends M {
    String s = "2";

    public void h(String s) {
        super.h(s);
        A6.print("N␣h");
    }
}

class S {
    public void g() {
        A6.print("W");
    }

}
```

```
class P {
    public void g() {
        A6.print("P␣g");
    }

}

class Q extends P implements I {
    String s = "Q";

    public void g() {
        j(s);
        A6.print("Q␣" + s);
    }

    public void j(String s) {
        A6.print("Q␣" + s);
    }

}

class R extends Q {
    String s = "3";

    public void j(String s) {
        A6.print("R␣" + s);
    }

}
```

In der Klasse A6 befinden sich die Methoden main und print.

```
class A6 {
  public static void main (String[] args) {
     /* body  */
  }
  public static void print(String s) {
     System.out.println(s);
  }
}
```

Die folgenden Anweisungen sollen als "Body" (Rumpf) anstelle des Kommentars `/* body */` eingefügt werden. Geben Sie für jede Anweisungsfolge an, was für eine Ausgabe erzeugt wird – entweder was gedruckt wird, oder ob ein Laufzeitfehler auftritt (schreiben Sie "Exception"), oder ob der Compiler einen Fehler feststellt (schreiben Sie "Compile-Fehler"). Falls ein gedruckter String Leerzeichen enthält, dann ist die genaue Anzahl/Weite der Leerzeichen unwichtig.

1.
```
I x = new M();
x.g();
```
_____

2.
```
I x = new N();
x.g();
```
_____

3.
```
M x = new N();
x.g();
```
_____

4.
```
M x = new M();
((N)x).g();
```
_____

5.
```
Object z = new N();
M x = (N) z;
x.h("X");
```
_____

6.
```
I x = new S();
x.g(p);
```
_____

7.
```
Object x = new P();
Q y = (Q) x;
y.g();
```
_____

8.
```
P x = new P();
I y = (I)x;
y.g();
```
_____

9.
```
I x = new R();
x.g();
```
_____

10.
```
P x = new R();
x.g();
```
_____

Aufgabe 6 (8)

Gegeben sei in Abbildung 1 die EBNF-Beschreibung von *assignment_expression*. Für die EBNF-Beschreibung von *identifier* gelten die Regeln für Bezeichner in Java. Die EBNF Beschreibung von *assignment_expression* unterscheidet sich aber sonst von den in Java zulässigen Ausdrücken.

arithmetic_op ⇐ `+` | `*` | `-` | `/`

arithmetic_expression ⇐ `(` *arithmetic_expression* `)` |
identifier arithmetic_op identifier | *identifier*

expression ⇐ *assignment_expression* | *arithmetic_expression*

element_select ⇐ `[` *expression* `]`

assignment_operator ⇐ `=` | `+` `=`

scalar_or_element ⇐ *identifier* | *identifier* { *element_select* }

assignment_expression ⇐ *scalar_or_element assignment_operator expression*

Abbildung 1: **EBNF-Beschreibung** von *assignment_expression*

Geben Sie für jeden folgenden Ausdruck an, ob er nach der EBNF-Beschreibung von *assignment_expression* in Abbildung 1 gültig ist. (Tipp: alle Bezeichner und Werte in diesen Ausdrücken (d.h., *identifier* in Abbildung 1) sind korrekt.)

Ausdruck	Gültig	Ungültig	Ausdruck	Gültig	Ungültig
`x = a`			`y += (a + b)`		
`z[i] = y[j]`			`a[i][j] = b[j][i]*c`		
`a = b = c`			`a[i+1] += (i * 2)`		
`a = a[i] + 1 = b`			`c = (a==b)`		
`(p = q) = r`			`a+b = v[j]`		

Aufgabe 7 (4)

Bitte geben Sie für die folgenden Java Programmsegmente die schwächste Vorbedingung (weakest precondition) WP an. Bitte geben Sie die Precondition als Java Expression an. Alle Variablen sind vom Typ `int` und initialisiert; es gibt keinen Over/Underflow.

1. `WP: {` `}`

```
r = 2 * s;
if (r <= 4) {
  d = r * 4;
} else {
  d = r + 2;
}

Q: { d > 0 }
```

2. `WP: {` `}`

```
if (a > 0) {
  c = 2 * a;
  b = c + 1;
}

Q: { b > 5 }
```

Wir wünschen Ihnen alles Gute für den Rest der Prüfungssession und das nächste Semester. Ihr "Einführung in die Programmierung"-Team.

Aufgabe	Wert	Punkte		Aufgabe	Wert	Punkte	
1	4			5	11		
2	4			6	8		
3	4			7	4		
4	5			–			
Σ				Übertrag	––		
				Total	40		