Do., 19.12.2013 Informatik Q2 M (Kr)

**2. Klausur 13/I (Q2.1)**

Dauer: 180 Minuten [180 P.] (10:45 bis 13:45 Uhr) Name: ____www.r-krell.de____

Hilfsmittel: Taschenrechner

* *Achte auf sorgfältige Darstellung mit vollständigem, nachvollziehbarem Lösungsweg!* *
* *Kommentiere deine Programme!* *

Der Weihnachtsmann geht mit der Zeit: Er will sein Geschäft künftig mit dem Computer verwalten.

**1** Softwareengineering, UML-Klassendiagramm und Java [48 P.]

Der Weihnachtsmann verwaltet Kinder, Eltern(teile), Geschenke und Rentiere.
Kinder und Eltern sind Menschen, von denen Vor- und Zuname und die Adresse (hier nur Straße und HausNr) interessieren. Bei jedem Kind ist zusätzlich auf einer Skala von 0 (ganz böse) bis 10 (extrem lieb) vermerkt, wie brav es im letzten Jahr war. Bei jedem Kind wird außerdem eine Reihung oder Liste für alle (0 bis höchstens 3) Geschenke geführt, die das Kind erhalten soll. Kinder können sich bedanken.
Der Weihnachtsmann merkt sich immer nur einen Elternteil pro Kind. Dieser eine Elternteil ist ein Mensch, der ein oder mehrere Kinder hat. Den Weihnachtsmann interessiert der Gesamtbetrag, den der Elternteil bezahlen muss – ein Elternteil muss also zahlen können.
Jedes Geschenk hat eine Bezeichnung („Teddybär“, „Modellauto“ o.ä.), ein Gewicht (in kg) und einen Preis (in Euro).
Für seine Schlittenfahrten stehen dem Weihnachtmann bis zu 12 Rentiere zur Verfügung. Bei jedem Tier wird außer dem Namen („Dasher“, „Dancer“, .., „Rudolf“) eine Besonderheit („rote Nase“, „grüne Augen“.. ) sowie die (Ganz-)Zahl der geleisteten Arbeitsstunden vermerkt. Die Tiere können ziehen.

a) [22 P.] Zeichne ein UML-Diagramm mit geeigneten Klassen und einer gemeinsamen Verwaltungsklasse. Ist eine Klasse abstrakt, so soll dies mit dem Hinweis <*abstract*> beim Klassennamen verdeutlicht werden. Die im Text genannten Attribute und Methoden sollen mit Typen eingetragen werden; die Attribute brauchen nicht ‚*private*‘ zu sein (bitte keine *get*- oder *set*-Methoden angeben!), als Parameterliste jeder Methoden reicht ( ) oder (...) ohne Details. Hingegen sind Vererbungspfeile sowie Assoziationen, Aggregationen und ggf. Kompositionen mit Multiplizitäten (=Kardinalitäten) einzuzeichnen.
b) [ 5 P.] Schreibe nur den Anfang der Klasse *Kind* passend zu a) in Java (inkl. Signatur = Kopfzeile der Methode *bedanktSich()*). Evtl. benötigte andere Klassen wie *Mensch* usw. sollen einfach benutzt, aber hier nicht in Java geschrieben werden.
c) [ 3 P.] Rentiere ziehen den Schlitten immer nur für eine oder mehrere volle Stunden. Wird z.B. *tier.ziehen (14, 17);* aufgerufen, so muss das Rentier *tier* den Schlitten von 14 Uhr bis 17 Uhr ziehen – also 3 Stunden arbeiten. Schreibe nur die eine Javazeile innerhalb der Methode *public void ziehen (int startzeit, int endzeit)*, die das in der Klasse *Rentier* vorhandene Attribut *arbeitsstunden* entsprechend erhöht. Kein Tier muss über Mitternacht hinaus arbeiten.
d) [12 P.] Beschreibe allgemein Aggregation und Komposition und erkläre den Unterschied mit/an einem selbstgewählten Beispiel.
e) [ 6 P.] Erläutere kurz den Sinn und Zweck eines UML-Diagramms, z.B. wieso das UML-Diagramm a) nützlich ist, wenn man ein Java-Programm für den Weihnachtsmann schreiben soll.

**2** Sortieren [62 P.]

Das Ausliefern der Geschenke geschieht straßenweise. Deshalb will der Weihnachtsmann die Kinder lexikografisch nach Straßen sortieren. In der Verwaltungsklasse gibt es *Kind[] kinder = new Kind[200000]* und *int kinderZahl* (= Füllgrad der Reihung *kinder*) – in der Klasse *Kind* gibt es u.a. die Attribute *String straße* (sowie *int hausNr* ). Alle Attribute sind ohne *private*-Zusatz.

a) Der Osterhase hat dem Weihnachtsmann ein spezielles Sortierverfahren empfohlen: In OsterSort wird der optimierte Bubblesort durch MinSort-Anteile ergänzt, damit nicht nur pro Durchgang die im Lexikon am weitesten hinten stehende Straße ans Ende getauscht wird, sondern auch das Kind mit der Straße vom Lexikonanfang nach vorne kommt (siehe Kasten).

```
public void osterSort()
{
 boolean getauscht;
 int durchgang = 0;
 int m;   // m = minStelle
 do
 {
  getauscht = false;
  m = durchgang;
  for (int i=durchgang; i < kinderZahl-durchgang-1; i++)
  {
   if (kinder[i].straße.compareTo(kinder[i+1].straße) >0)
   {
    tausche (i, i+1);
    getauscht = true;
   }
   if (kinder[i].straße.compareTo(kinder[m].straße) <0)
   {                                  // ^ ***
    m = i;
   }
  } // end of for
  if (m > durchgang)
  {
   tausche (m, durchgang); // ‚kleine' Str. nach vorne
  }
  durchgang++;
 } while (getauscht == true);
}
```

a1) [ 6 P.] Schreibe die passende Methode *tausche* in Java

a2) [ 6 P.] An der mit *** markierten Stelle wird nur *kinder[i]*, nie *kinder[i+1]* mit *kinder[m]* verglichen. Erläutere, ob dies problematisch oder gefahrlos ist.

a3) [ 4 P.] Begründe kurz, warum bei *kinderZahl* = *n* höchstens *n/2* Durchgänge nötig sind.

a4) [20 P.] Erstelle nachvollziehbar für OsterSort jeweils eine Formel für die Zahl V der Vergleiche von Kindern bzw. für die Zahl U von Umspeicherungen von Kindern, jeweils für den besten und den schlechtesten Fall. Gib für jeden Fall außerdem den Gesamtaufwand in O-Notation an und notiere, bei welcher Vorsortierung der beste bzw. schlechteste Fall vorliegt.

b) Nach der Sortierung in a) stehen zwar alle Kinder mit gleicher Straße in der Reihung *kinder* direkt hintereinander, aber ihre Hausnummern gehen noch wild durcheinander. Eine zusätzliche Sortierung nach Hausnummern (ganzzahliges Attribut *hausNr*) ist nötig.

b1) [ 6 P.] Markiere oben im Kasten für *osterSort* die Stellen im Programmtext, die für das Sortieren nur nach Hausnummern (statt nach Straßen) abgeändert werden müssen, und schreibe die veränderte(n) Zeile(n) auf den Klausurbogen. Begründe kurz, ob/wie *tausche* verändert werden muss.

b2) [15 P.] Angenommen, es gäbe nur *kinderZahl* = 7 Kinder, die hier nur durch ihre Hausnummern 12 33 84 17 21 65 57 repräsentiert werden. Führe a) OsterSort und b) EinSort jeweils von Hand durch und notiere am Ende jeden Durchgangs die Durchgangsnummer und die aktuelle Reihenfolge der Hausnummern auf Papier. Zwischenschritte sollten angedeutet werden, müssen aber nicht.

b3) [ 5 P.] Nicht nur die Kinder einer Straße, sondern die gesamte Reihung *kinder* wird einmal komplett nach Hausnummern sortiert. Überlege begründet, ob vorher oder nachher alles nach Straßen sortiert werden sollte, wenn man am Ende der beiden Sortiervorgänge die Kinder mit Ellerstr. 10, Ellerstr. 13, Ellerstr. 84,.. hintereinander und vor denen aus der Friedrichstr. 23, Friedrichstr. 24, .. stehen haben will. Gib außerdem an, welche Eigenschaften das (zweite) Sortierverfahren haben muss, damit das gewünschte Ergebnis erscheint.

**3** Kryptologie [70 P. ohne e)]

Der Weihnachtsmann achtet auf den Datenschutz. Deshalb will er die persönliche Bewertung der Bravheit der Kinder (ausgedrückt durch eine ganze Zahl *brav* zwischen 0 und 10, vgl. Text am Anfang von Aufgabe 1) sicher verschlüsseln, damit Fremde nicht an diese sensiblen Informationen gelangen können. Er vertraut auf das RSA-Verfahren.

a) [ 8 P.] Erläutere in wenigen Sätzen die entscheidenden Unterschiede zwischen symmetrischen und asymmetrischen Chiffrierverfahren und ordne das RSA-Verfahren richtig zu.

b) [ 6 P.] Der Klapperstorch ist mit seiner traditionellen Aufgabe immer weniger ausgelastet und überlegt, ein Trustcenter mit Schlüsselagentur zu eröffnen. Als Probe seines neuen Könnens versorgt er den Weihnachtsmann mit p=7, q=17 und schlägt ihm als öffentlichen Schlüssel (n, e) = (96; 40) vor. Berechne φ und prüfe durch Rechnung, ob n richtig und das e geeignet ist!

c) Der Weihnachtsmann bevorzugt (wieder für p=7, q=17) die Zahl e=7 für den öffentlichen Schlüssel.

c1) [12 P.] Bestimme mit dem Erweiterten Euklid-Algorithmus ein passendes d und gib den öffentlichen und den privaten Schlüssel vollständig an.

c2) [10 P.] Das d kann auch durch „Probieren“ herausgefunden werden, für welches Vielfache v = 1, 2, 3, 4,... der Quotient $d = (1 + v \cdot \varphi)/e$ natürlich ist. Ermittle d auf diese Weise und vergleiche mit der Lösung aus c1). Überlege auch, ob grundsätzlich mehrere verschiedene Lösungen für d möglich oder sinnvoll wären.

c3) [ 5 P.] Der Weihnachtsmann hat (n, e) veröffentlicht und sich (n, d) gemerkt (andere kommen an (n, d) nicht heran). Jetzt gerät er ins Grübeln: Weil er die Bravheitszahlen ja nicht verschicken, sondern nur für sich selbst verschlüsseln will, ist er sich nicht sicher, ob er zum Verschlüsseln den öffentlichen oder den privaten Schlüssel nehmen soll. Hilf dem Weihnachtsmann begründet zur richtigen Wahl.

c4) [ 8 P.] Der Weihnachtsmann hat gehört, dass man große Zahlen p und q für das RSA-Verfahren nehmen soll. Angeblich gibt es neben der maximalen Größe der zu verschlüsselnden Zahl *brav* noch einen zweiten Grund. Nenne beide Gründe und entscheide, ob p=7 und q=17 für die Verschlüsselung der möglichen Bravheitszahlen ausreichen. Und: muss auch e größer als die größtmögliche Bravheitszahl 10 sein? Und wie werden immer *brav*=0 und *brav*=1 verschlüsselt?

d) Der Weihnachtsmann nimmt jetzt zwei Primzahlen mit jeweils über 100 Dezimalstellen; hier zum Rechnen nur p=17 und q=29. Weiterhin sind φ = 448, n = 493, e =11 und d=163. Außerdem addiert der Weihnachtsmann immer die Zahl 13 zum Bravheitswert, d.h. verschlüsselt bei *brav*=4 die Zahl k=*brav*+13=17.

d1) [12 P.] Berechne $g = k^e \bmod n = k^e \,\%\, n$ mit den gegebenen Zahlen von Hand nach dem square&multiply-Verfahren!

d2) [ 9 P.] Überlege, wie viele verschlüsselte Werte von Bravheitszahlen auftreten können und beurteile die Sicherheit des Verfahrens (wenn n mit mehreren Hundert Stellen zu groß für eine Faktorisierung ist). Wie könnte der Weihnachtsmann die Sicherheit erhöhen?

e) (*nur, falls noch Zeit* [0..+10 P.]): Zunehmend geben sich Betrüger als Weihnachtsmann aus, um von den Kindern ihre geheimsten Wünsche per Wunschzettel zu erfragen oder sich in böser Absicht Zugang zu deren Haus zu verschaffen. Erläutere kurz (aber prägnant), wie eine Digitale Signatur e-Mails des echten Weihnachtsmanns von Betrüger-Meldungen unterscheiden hilft.