Di., 25.11.2008 Informatik 11 M (Kr)

2. Klausur 11/I (A)

Dauer: 2 Schulstunden Name: ___***www.r-krell.de***___

Hilfsmittel: normaler Taschenrechner,

* *Achte auf sorgfältige Darstellung mit vollständigem, nachvollziebarem Lösungsweg!* *

* *Kommentiere deine Programme!* *

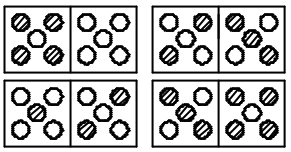
1 Bewegung

In einer Flüssigkeit steigt ein einziges kreisrundes Gasbläschen (mit Radius 5) von der Position (50,400) nach oben zu (50,100) (angegeben sind jeweils die Koordinaten des Mittelpunkts).

a) Schreibe nur die Methode *steigt()*, wobei es bereits einen Stift *stift* geben soll. Nach Ablauf der Methode sollte die Blase gelöscht sein (weil sie sich beim Erreichen der Oberfläche auflöst).

b) Markiere in deiner Methode *steigt* aus a) die Stelle, wo durch Einfügen der Zeile(n) *if (taste.wurdeGedrueckt() && taste.zeichen() == 's') { y = 105; }* erreicht werden kann, dass durch Drücken der Taste ‘s’ die Blase sofort bis fast ganz oben hoch geht. Unten bzw. an der bisherigen Position soll natürlich kein Rest der Blase mehr zurück bleiben.

2 Mehrere Klassen



Auf Dominosteinen für Anfänger sind nur die Zahlen 0 bis 4 möglich. Jeder Stein besteht aus zwei quadratischen Hälften, auf denen kein, ein, zwei, drei oder vier so genannte Augen dunkel gefärbt sind (siehe Beispiele). Ein Stein hat die Maße 160 Pixel x 80 Pixel; die Augen haben einen Radius von 10 Pixeln.

a) Zeichne die linke Hälfte eine Dominosteins groß in dein Heft und lege sinnvolle Positionen für die Mittelpunkte der 5 Augen fest, wenn die obere linke Ecke der Hälfte bei (*x;y*) liegt. Natürlich lassen sich keine reinen Zahlen angeben, sondern müssen Terme wie etwa *x–17* verwendet werden. Die ganze Hälfte soll 80 x 80 Pixel groß sein.

b) Schreibe eine Java-Klasse *Haelfte* für eine Dominosteinhälfte, die die gegebene Klasse *Auge* benutzt. Die Hälfte muss an eine bestimmte Position gesetzt werden können und soll eine der Zahlen 0 bis 4 anzeigen können (Es braucht aber nur der Text für das Anzeigen von 2 und 3 geschrieben werden. „...“ für 0,1 und 4).

c) Wird ein Stein z.B. mit den Zahlen 4 und 0 neu auf den Tisch bzw. mit der oberen linken Ecke an die Position (120, 70) auf den Bildschirm gelegt, so soll dies mit dem Befehl *Stein dominoA = new Stein (120,70,4,0);* geschehen können. Schreibe die Java-Klasse *Stein*.

d) Gib mit kurzer Begründung (aber ohne Java-Text) an, wo der Bildschirm erzeugt werden müsste.

e) Nenne die Vor- und Nachteile, die sich daraus ergeben sich, dass statt eines einzigen großen Dominoprogramms vier oder fünf eigene Klassen geschrieben und verwendet werden.

```
import stiftUndCo.*;
public class Auge
{
  BuntStift stift = new BuntStift();

  public Auge (int xM, int yM) // Konstruktor
  {
    stift.bewegeBis (xM, yM); // Mittelpunkt
    stift.setzeFuellMuster (Muster.GEFUELLT);
  }

  public void hell()  // hell = unsichtbar
  {
    stift.radiere();
    stift.zeichneKreis(10);  // löschen
  }

  public void dunkel()  // dunkel = sichtbar
  {
    stift.normal();
    stift.zeichneKreis(10);  // zeichnen
  }
```

3 Kontrollstrukturen

In folgenden Spielen soll immer eine Zufallszahl *x* zwischen 1 und 4 elektronisch 'gewürfelt' werden.

```
import stiftUndCo.*;
public class Aufgabe3
{
  Bildschirm f = new Bildschirm();
  BuntStift stift = new BuntStift();

  public void spielA ()
  {
    int x = Hilfe.zufall (1, 4);
    ...
    stift.schreibe ("x="+x);
  }
```

a) Beim Spiel A gilt: wird eine Zahl *x* kleiner als drei gewürfelt, soll diese Zahl *x* zu *2*x* verdoppelt werden. Wird drei gezogen, soll 3 addiert werden. Und wenn eine Zahl *x* größer als drei gezogen wird, soll von *x* eine 1 abgezogen werden. Das Ergebnis soll wieder in *x* gespeichert werden und wird vom Stift auf den Bildschirm geschrieben.

   a1) Schreibe den restlichen Javatext*) der bereits angefangenen Methode *spielA* unter Verwendung von 1- und/oder 2-seitigen Verzweigungen!

   a2) Schreibe nochmal *spielA*, jetzt aber nur mit *switch .. case* (4 Fälle!). *)

   a3) In der Zeile *stift.schreibe ("x="+x);* steht hinten in der Klammer zweimal *x*. Erläutere die Bedeutung von *x* mit und ohne Anführungszeichen sowie die Funktion des '+'-Zeichens. Und: Was geschieht durch diese Zeile?

   a4) Schreibe eine Startdatei, die ein Objekt nach dem Bauplan der Klasse Aufgabe3 erzeugt und dort das Spiel A startet.

b) Beim Spiel B soll eine 3 gewürfelt werden. Notfalls muss eben solange eine Zufallszahl *x* zwischen 1 und 4 gezogen werden, bis endlich das gewünschte Ergebnis erreicht ist/wird.

   b1) Entscheide jeweils mit kurzer Begründung, ob hierzu eine (1) *for*-, (2) *while*- oder (3) *do....while*-Schleife am besten geeignet ist, oder ob (4) ein *if* reicht.

   b2) Schreibe den kurzen Javatext von Spiel B mit der besten Kontrollstruktur (das letzte *x* soll wieder mit dem Stift geschrieben werden).

c) Auch beim Spiel C wird gewürfelt (siehe Kasten).

   c1) Notiere in einer Tabelle, wie oft der Schriftzug „SpielC" auf dem Bildschirm erscheint, wenn am Anfang *x* den Wert 1, 2, 3 oder 4 erhält.

   c2) Der gegebene Programmtext ist unnötig kompliziert. Notiere eine bessere Version für *spielC*!

```
public void spielC ()  // Aufg. 3c
{
  int x = Hilfe.zufall (1, 4);
  do {
    if ( x < 4)
    {
      x++;
      stift.schreibe ("SpielC ");
    }
  } while (x < 4);
}
```

---

*) statt Javatext ist auch ein beschriftetes Struktogramm erlaubt.