Maximilian Petri, 405602
Danje Petersen, 379748

Hausaufgabe 2

Programmierung, Gruppe 16
30. Oktober 2019

Aufgabe 2)

a)

2147483647 + '1' = -2147483600 (int)

Da das plus 2 Integer in dem Fall addieren möchte wird der char impliziet als int gecastet und hat dann den Wert '1' = 49. Nun wird auf die größt mögliche Zahl in einem int $2^{31} - 1$ die 49 draufaddiert was zu einem Überlauf führt, welcher diese "komischeünd mathematisch falsche Ergebnis hervorruft.

b)

(byte) 256 * 2147483648 → Fehler

Da der rechte Faktor (2147483648) größer als der größtmögliche int ist $(2^{31} - 1)$ und niergendwo angegeben ist, das es sich um einen long handelt, nimmt der Compiler an es solle sich um einen int handeln. Die Zahl ist jedoch zu groß um sie mit einem int darzustellen, weshalb man einen Fehler erhällt.

c)

"x"+ y + z = "x"+ 2 + 3 = "x23"(String)

Java interpretiert Additionen immer von links nach recht, weshalb zuerst "x"+ y berechnet wird. Da "xëin String ist, wird die String-Addition bzd. Konkatenation verwendet und der rechte Ausdruck (y bzw. 2) impliziet zu einem String umgewandelt und das Ergebnis ist nun "x2äls String. Nun wiederholt sich der Prozess mit "x2"+ 3, da nun wieder der rechte Teil impliziet als String umgewandelt wird und mit "x2"konkatiniert wird.

d)

$x < y$ & & true = 120 < 2 & & true → false (boolean)

In diesem Ausdruck werden zwei Wahrheitswerte miteinenader "ver-undet". Der rechte ist true, also immer wahr, weshalb es nur auf den linken Ausdruck $x < y$ ankommt. Dieser ist für die vorgegebenen Variablenwerte false, weil 120 nicht kleiner als 2 ist. Deshalb wird false und true "ver-undet", was false ergibt, da bei einer und-Verknüpfung nur true heraus kommt wenn beide Werte true sind.

e)

9f / 3d = 3.0 (double)

Hier wird ein float durch einen double geteilt, weshalb Java den float impliziet in einen double umwandelt. Es wird also 9d / 3d gerechnet, was als Ergebnis 3 vom Typ double hat.

f)

x + "y z = 120 + "y 3 → Fehler

Der Ausdruck wird von Java von links nach rechts bearbeitet. Deshalb wird zuerst x + "2", bzw. 120 + "2"gerechnet. Da "yëin String ist, wandelt Java x (120) auch in einen String "120üm, und konkatiniert die beiden dann zu "120y". Nun wird versucht z (3) von diesem String abzuziehen. In Java gibt es keine Definition für die Subtraktion von einem String, weshalb dies syntaktisch nicht korrekt ist und einen Fehler erzeugt.

g)

1 + 3f = 4.0 (float)

Es wird ein int und ein float miteinander addiert, weshalb Java den int impliziet zu einem float konvertiert. Nun wird 1f + 3f berechnet, was als Ergebnis 4 als float hat.

h)

x != 'x' ? 2f : 1 → 120 != 'x' ? 2f : 1 → 120 != 120 ? 2f : 1 → 1.0 (float)

Hier handelt es sich um einen Ternary-Ausdrück, welche immer zuerst einen Wahrheitswert entgegennimmt und je nachdem ob dieser true oder false ist, wird der Ausdruck links oder rechts vom ":"das Ergebnis. In diesem Fall wird sich angeguckt ob 120 != 'x' ist. Der char 'x' wird hierzu impliziet zu einem int gecastet um die beiden Ausdrücke miteinander vergleichen zu können, also 120 != 120. Das ist false da sie ja gleich sind, weshalb der hintere Teil, die 1 als float zurückgegeben wird. Diese wurde nun von Java impliziet in einen float gewandelt, da das Ergebnis eines Ternary-Ausdrucks immer vom Typ des abstraktesten Datentyps der beiden Rückgabemöglichkeiten ist.

Aufgabe 4)

```
public class Zinsrechner {

  public static void main(String[] args) {

    double startbetrag = SimpleIO.getDouble("Was ist ihr Startbetrag?");
    double zinssatz = SimpleIO.getDouble(
                        "Was ist ihr Zinssatz in Prozent?") / 100 + 1;

    String modus = SimpleIO.getString(
                        "Geben Sie 'Ziel' fuer den Zielmodus"
                        + "oder 'Zeit' fuer den Zeitmodus ein!");

    if (modus.equals("Ziel")) {
      double zielbetrag = SimpleIO.getDouble("Was ist ihr Zielbetrag?");
      double betrag = startbetrag;
      int jahre = 0;

      while (betrag < zielbetrag) {
        betrag *= zinssatz;
        jahre++;
      }

      SimpleIO.output(
        "Es dauert " + jahre + " Jahre bei einem Zinssatz von "
            + Math.round(zinssatz * 10E5) / 10E5
            + "%, um von " + startbetrag + " auf den Betrag "
            + zielbetrag + " zu sparen. Nach dieser Zeit hat man "
            + Math.round(betrag * 100.0) / 100.0 + ".",
        "Ergebnis");
    }
    else if (modus.equals("Zeit")) {
      double betrag = startbetrag;
      int jahre = SimpleIO.getInt(
                  "Wie viele Jahre sollen berechnet werden?");

      for (int i = 0; i < jahre; i++) {
        betrag *= zinssatz;
      }

      SimpleIO.output(
        "Bei einem Zinssatz von " + Math.round(zinssatz * 10E5) / 10E5 +
            "% und einem Startbetrag von "  + startbetrag + " hat man nach "
            + jahre + " Jahren " + betrag + " gespart.",
        "Ergebnis");
    }
    else {
      SimpleIO.output("Ihre Eingabe war Fehlerhaft!", "FEHLER");
    }

  }
}
```

Aufgabe 6)

a)

Partielle Korrektheit:
Die Schleifeninvariante ist $res = (-1)^i \cdot i \land sign = (-1)^i \land i \leq n$.

$\langle 0 \leq n \rangle$
$\langle 0 \leq n \land 1 = 1 \rangle$
sign = 1;
$\langle 0 \leq n \land sign = 1 \land 0 = 0 \rangle$
res = 0;
$\langle 0 \leq n \land sign = 1 \land res = 0 \land 0 = 0 \rangle$
i = 0;
$\langle 0 \leq n \land sign = 1 \land res = 0 \land i = 0 \rangle$
$\langle res = (-1)^i \cdot i \land sign = (-1)^i \land i \leq n \rangle$
while(i < n) {
$\langle res = (-1)^i \cdot i \land sign = (-1)^i \land i \leq n \land i < n \rangle$
$\langle -(res + sign) = (-1)^{i+1} \cdot (i+1) \land -sign = (-1)^{i+1} \land i + 1 \leq n \rangle$
res = -(res + sign);
$\langle res = (-1)^{i+1} \cdot (i+1) \land -sign = (-1)^{i+1} \land i + 1 \leq n \rangle$
sign = -sign;
$\langle res = (-1)^{i+1} \cdot (i+1) \land sign = (-1)^{i+1} \land i + 1 \leq n \rangle$
i = i + 1;
$\langle res = (-1)^i \cdot i \land sign = (-1)^i \land i \leq n \rangle$
}
$\langle res = (-1)^i \cdot i \land sign = (-1)^i \land i \leq n \land \neg(i < n) \rangle$
$\langle res = (-1)^n \cdot n \rangle$

Damit ist unter der Vorbedingung die partielle Korrektheit des Programms gezeigt.

b)

Terminierung:
Die Variante der Schleife ist $n - i$.

$\langle n - i = m \land i < n \rangle$
$\langle n - (i+1) < m \rangle$
res = -(res + sign);
$\langle n - (i+1) < m \rangle$
sign = -sign;
$\langle n - (i+1) < m \rangle$
i = i + 1;
$\langle n - i < m \rangle$

Damit ist gezeigt, dass das Programm unter der Vorbedingung terminiert.

Aufgabe 8)

Partielle Korrektheit:
Die Schleifeninvariante ist $res = \lceil \frac{i}{2} \rceil \cdot n - \lfloor \frac{i}{2} \rfloor \wedge i \leq n$.

$\langle 0 \leq n \rangle$
$\langle 0 \leq n \wedge 0 = 0 \rangle$
  i = 0;
$\langle 0 \leq n \wedge i = 0 \wedge 0 = 0 \rangle$
  res = 0;
$\langle 0 \leq n \wedge i = 0 \wedge res = 0 \rangle$
$\langle res = \lceil \frac{i}{2} \rceil \cdot n - \lfloor \frac{i}{2} \rfloor \wedge i \leq n \rangle$
  while (i < n) {
    $\langle res = \lceil \frac{i}{2} \rceil \cdot n - \lfloor \frac{i}{2} \rfloor \wedge i \leq n \wedge i < n \rangle$
      if (i % 2 == 0) {
        $\langle res = \lceil \frac{i}{2} \rceil \cdot n - \lfloor \frac{i}{2} \rfloor \wedge i \leq n \wedge i \mod 2 = 0 \rangle$
        $\langle res + n = \lceil \frac{i+1}{2} \rceil \cdot n - \lfloor \frac{i+1}{2} \rfloor \wedge i + 1 \leq n \rangle$
          res = res + n;
        $\langle res = \lceil \frac{i+1}{2} \rceil \cdot n - \lfloor \frac{i+1}{2} \rfloor \wedge i + 1 \leq n \rangle$
      } else {
        $\langle res = \lceil \frac{i}{2} \rceil \cdot n - \lfloor \frac{i}{2} \rfloor \wedge i \leq n \wedge \neg(i \mod 2 = 0) \rangle$
        $\langle res - 1 = \lceil \frac{i+1}{2} \rceil \cdot n - \lfloor \frac{i+1}{2} \rfloor \wedge i + 1 \leq \rangle$
          res = res - 1;
        $\langle res = \lceil \frac{i+1}{2} \rceil \cdot n - \lfloor \frac{i+1}{2} \rfloor \wedge i + 1 \leq n \rangle$
      }
    $\langle res = \lceil \frac{i+1}{2} \rceil \cdot n - \lfloor \frac{i+1}{2} \rfloor \wedge i + 1 \leq n \rangle$
      i = i + 1;
    $\langle res = \lceil \frac{i}{2} \rceil \cdot n - \lfloor \frac{i}{2} \rfloor \wedge i \leq n \rangle$
  }
$\langle res = \lceil \frac{i}{2} \rceil \cdot n - \lfloor \frac{i}{2} \rfloor \wedge i \leq n \wedge \neg(i < n) \rangle$
$\langle res = \lceil \frac{n}{2} \rceil \cdot n - \lfloor \frac{n}{2} \rfloor \rangle$

Damit ist unter der Vorbedingung die partielle Korrektheit des Programms gezeigt.

Terminierung:
Die Variante der Schleife ist $n - i$.

$\langle n - i = m \wedge i < n \rangle$
$\langle n - (i + 1) < m \rangle$
  if (i % 2 == 0) {
    $\langle n - (i + 1) < m \rangle$
      res = res + n;
  } else {
    $\langle n - (i + 1) < m \rangle$
      res = res - 1;
  }
$\langle n - (i + 1) < m \rangle$
  i = i + 1;
$\langle n - i < m \rangle$

Damit ist gezeigt, dass das Programm unter der Vorbedingung terminiert.

Da das Programm partiell Korrekt ist und immer termniniert ist es total Korrekt.