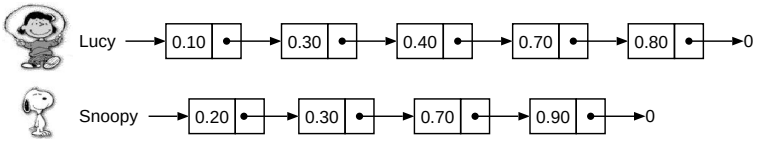
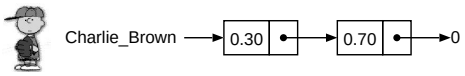
Übung 9

Aufgabe 1: Snoopy und Lucy's Geburtstagsüberraschung

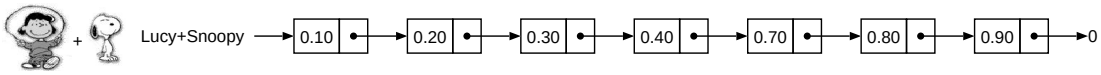
Snoopy und Lucy vergleichen ihre Briefmarkensammlungen. Sie haben nur amerikanische Briefmarken und keine einzige doppelt. Deshalb haben beide ihre Briefmarken der Reihe nach aufsteigend sortiert:



Die beiden haben bemerkt, dass sie zusammen ein paar Briefmarken doppelt haben. Weil Charlie Brown bald Geburtstag hat, legen die beiden ihre Briefmarkensammlungen zusammen und geben ihm die doppelten Exemplare. Charlie Brown bekommt also folgende Briefmarken:



Lucy und Snoopy behalten zusammen folgenden Rest:



Natürlich war das nur ein kleines Beispiel. Die beiden haben eine viel grössere Briefmarkensammlung und brauchen zur Durchführung dieser Idee daher Ihre Hilfe als Programmierer.
Die beiden Briefmarkensammlungen werden in Form von *sortierten Listen* verwaltet. Ihre Datenstruktur entspricht der herkömmlichen Deklaration:

```
1 struct Sammlung {
2   double marke;
3   Sammlung *next;
4 };
5
6 Sammlung *briefmarken;
```

Sortierte Listen sind Listen, deren Elemente in einer sortierten Reihenfolge geordnet sind (z.B. nach aufsteigenden Werten).

Die folgende Prozedur `einfuegen` fügt den Markenwert `marke` in der aufsteigend sortierten Liste `briefmarken` am vorgesehenen Platz ein:

```
 1 void einfuegen(Sammlung *&briefmarken, double marke)
 2 {
 3   Sammlung *akt;       // eine Briefmarke in der Sammlung
 4   Sammlung *vor;       // ihr Vorgaenger
 5   Sammlung *neu;       // die neue Briefmarke
 6
 7   akt = briefmarken;   // die erste Briefmarke in der Sammlung
 8   vor = 0;             // hat keinen Vorgaenger
 9
10   // Wir suchen den richtigen Platz fuer die Marke:
11   while ((akt!=0) && (akt->marke<marke)) { // solange nicht zu ende und
12                                           // neue Marke > akt. Marke
13     vor = akt;                            // wird die akt. Marke zum
14                                           // Vorgaenger
15     akt = akt->next;                      // wir gehen eine Marke weiter
16   }
17   // Wir sind am richtigen Platz: akt->marke>=marke
18
19   neu = new Sammlung;  // wir erzeugen die neue Briefmarke
20   neu->marke = marke;  // und geben ihr den Wert, den sie verdient
21   neu->next = akt;     // die neue Briefmarke kommt vor die aktuelle
22
23   if (vor==0) {        // wenn kein Vorgaenger (Liste leer oder ganz vorne)
24     briefmarken = neu; // aendert sich der Listenkopf
25   } else {             // sonst:
26     vor->next = neu;   // bekommt der Vorgaenger einen neuen Nachfolger
27   }
28 }
```

Aufgaben:

a) Schreiben Sie eine Prozedur

```
void geburtstag
  (Sammlung *&chbr, Sammlung *&lusn, Sammlung *lu, Sammlung *sn)
```

welche die Briefmarkensammlungen von Lucy, `lu`, und Snoopy, `sn`, so verarbeitet, dass auf die beschriebene Art eine Briefmarkensammlung für Charlie Brown, `chbr`, und die Sammlung von Lucy und Snoopy, `lusn`, entsteht. Dabei soll die gegebene Einfügeprozedur *nicht* verwendet werden. Es sollen vielmehr die bestehenden Elemente der gegebenen Listen umgehängt werden.

b) Vervollständigen Sie die in Aufgabe a) geschriebene Prozedur zu einem kleinen Testprogramm, das die Funktionsfähigkeit der Prozedur zeigt. Zum Aufbauen der Listen verwenden Sie die angegebene Einfügeprozedur.