Aufgabe 3 — Addition großer Zahlen — 35 Punkte

C++ verwendet bei ganzen Zahlen 16 Bit für `short int`, 32 Bit für `int` und 64 Bit für `long int`. Der damit darstellbare Zahlenbereich ist für die meisten Anwendungen ausreichend. Will man längere Zahlen verarbeiten, so muss man die einzelnen Stellen der Zahl in einer Liste darstellen und diese dann stellenweise verarbeiten – wir werden in dieser Aufgabe stets 3 Dezimalstellen zusammenfassen und große Zahlen als Liste von solchen 3er-Blöcken darstellen.

Beispiele:

`2098067145` wird dargestellt als `(2,98,67,145)`
`5102952000` wird dargestellt als `(5,102,952,0)`

Die Addition geht nun stellenweise vor wie beim schriftlichen Addieren:

```
  2 098 067 145
+ 5 102 952 000
===============
  7 201 019 145
```

Das Ergebnis ist also die Liste `(7,201,19,145)`. Beachten Sie den Übertrag, der hierbei vorkommt (`67+952=1019` ⇒ Ergebnis für diesen 3er-Block ist `19` und der Übertrag muss bei dem links davon stehenden 3er-Block berücksichtigt werden).

a) Erstellen Sie ein Datenhierarchiediagramm (DH) für die hier verwendete Liste und geben Sie die Datenstruktur in C++ an (die maximale Anzahl der 3er-Blöcke soll durch eine Konstante `Max3Block` definiert sein).

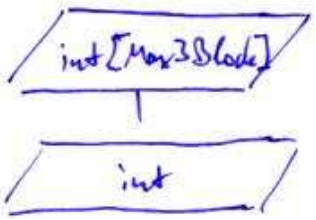
b) Entwerfen Sie einen Algorithmus zur Addition großer Zahlen und geben Sie den zugehörigen Programmablaufplan (PA) und das Programmnetz (PN) an. Ist das Ergebnis zu groß, so soll eine Fehlermeldung ausgegeben werden.

c) Setzen Sie Ihren Entwurf in eine C++-Funktion um.

Aufgabe 3 — Lösungshinweise — 35 Punkte

```
const int Max3Block = 4;
typedef int myint[Max3Block];
```

DH:

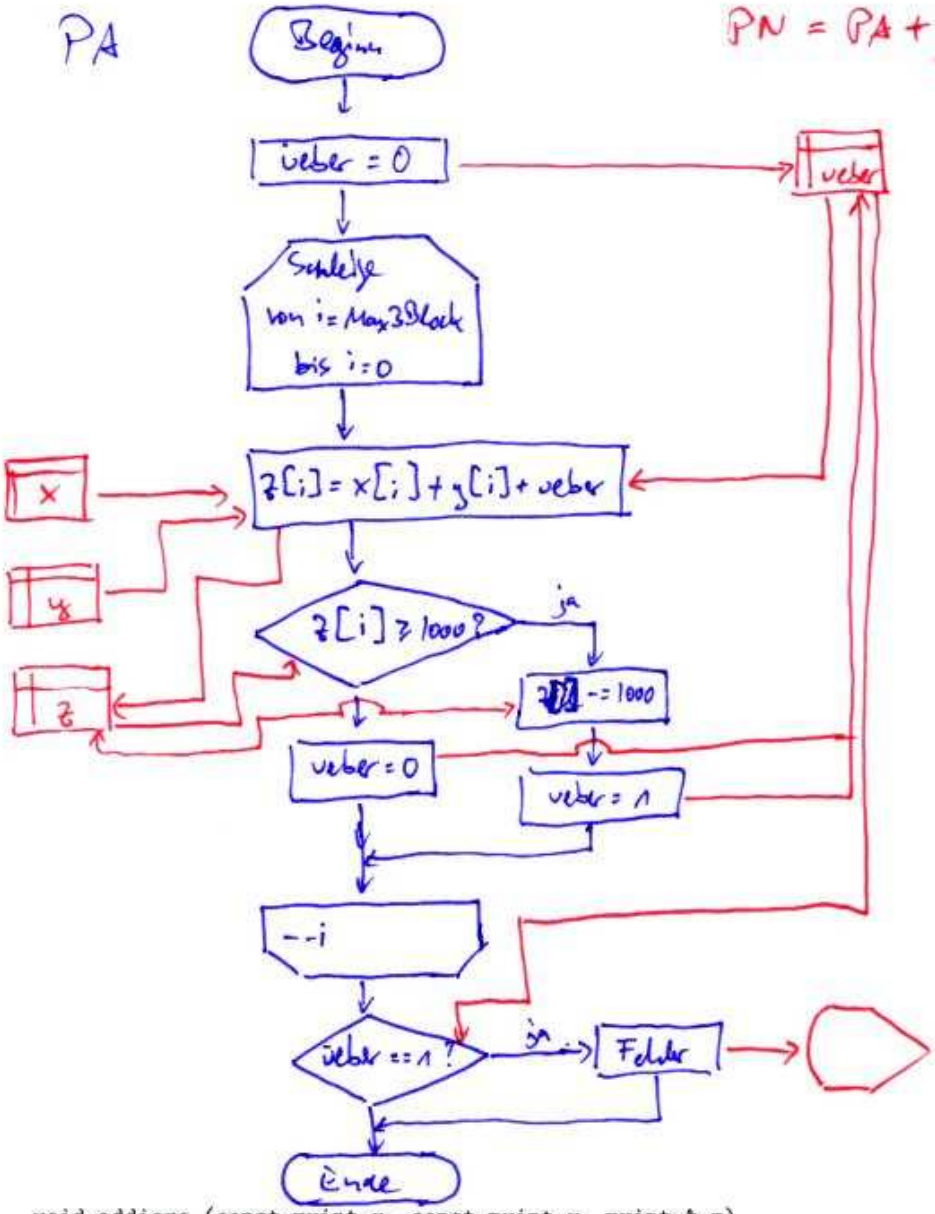


(andere Varianten möglich)

PA & PN müssen getrennt angegeben werden (zumindest kenntlich machen, was in welchem Diagramm dazugehört).

PA

PN = PA + „ "




```
void addiere (const myint x, const myint y, myint & z)
{
  int ueber=0;
  for(int i=Max3Block;i>=0;--i) {
    z[i]=x[i]+y[i]+ueber;
    if (z[i]>=1000) {z[i]=z[i]-1000; ueber=1;} else {ueber=0;}
  }
  if (ueber==1) {cout << "Fehler" << endl; }
}
```