Vorlesung 10

- Rekursion und Iteration
- Ein- und Ausgabe durch Datei
- Nicht elementare Datentypen
- Neben den elementaren Datentypen und Arrays gibt es in C++ auch feste und variable heterogene Datentypen.
- Weiter hin existieren Bitfelder für Speicherplatzsparende Strukturen.
- In C++ können Datentypen definiert und unter eigenem Namen verwendet werden.
- Das C++ Dateikonzept

Iteration

- Die Iteration ist der Begriff für ein Verfahren, das wiederholt wird, um ein bestimmtes Ergebnis zu erreichen oder sich anzunähern.

  Eine Iteration in Informatik besteht deshalb wesentlich aus einer Schleife.

Beispiel: Fibonacci-Zahlen `f(n) = f(n-1) + f(n-2)`

1, 1, 2, 3, 5, 8, 13, 21, 35,...

werden durch eine Iteration (Schleife) berechnet.

Rekursion

- Als Rekursion versteht man ein Verfahren, das durch sich selbst definiert ist.

  In Informatik bedeutet für eine Funktion dies, dass die Funktion sich selbst aufruft.

n!=n*(n-1)! und n! = 1, wenn n=1

```
fakultaet (n)
{
  if (n == 1)
  return 1;
  else
   return n*fakultaet(n-1);
}
```

Iteration und Rekursion

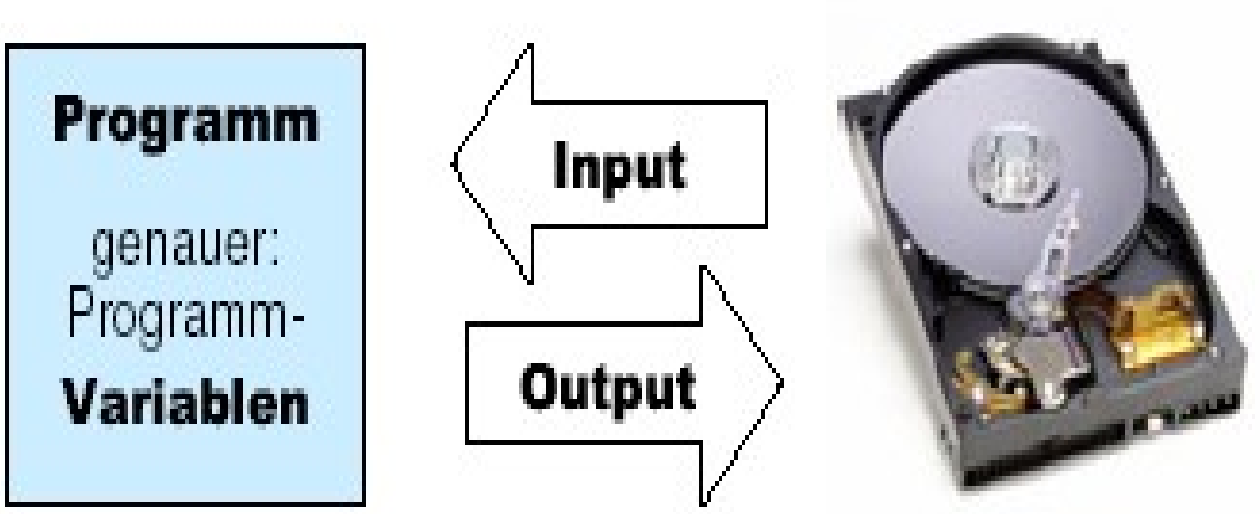
- Generell gilt:
  - Alle rekursive Verfahren können als eine Iteration ausgedrückt werden.
  - Ein iteratives Algorithmus läuft in der Regel schneller als ein rekursives. Rekursiv ist aber einfacher zu programmieren und eleganter.
- Die Fakultät als rekursive Funktion

```
fakultaet (n)
  {int i, f=1;

    for (i=2; i <=n; i++)
    f = f * i;

    return f;

  }
```

Datei-Ein- und Ausgabe:

Ziel:

Wir wollen Programme schreiben, die selber Verbindungen zu Dateien aufbauen, aus denen sie Daten einlesen bzw. in die hinein sie Daten schreiben können!



Dipl-Ing. J.L. Muluem
Dipl-Math J. Purcalla Arrufi

Dateien verarbeiten!

Das C++ Dateikonzept

Der Zugriff auf Dateien besteht aus drei Schritten:

- 1. Öffnen bzw. Anlegen der Datei
- 2. Zugriff auf die Datei
- 3. Schließen der Datei

Ströme von Daten: STREAMS

Die "**Streams**" sind ein Konzept für die **Datenübertragung** im Rechner.

In **C++** sind die Stream-Objekte definiert, sodass sie eine Folge von Bytes liefern beziehungsweise einlesen und ausgeben können.

- **Standard:** `cin, cout;`
- **Dateien:** `ifstream, ofstream, fstream`

Funktionsweise der Dateiverwaltung

1. Deklaration der Strom-Variable.

   Durch die Typen `ifstream ofstream`

2. Öffnen des Stromes.

   Durch die Funktion (Methode) `open`

3. Datenübertragung.

   Durch die Operatoren `<<` `>>`

4. Schließen des Stromes.

   Durch die Funktion (Methode) `close`

1. Deklaration der Strom-Variable

- Durch die Objekte
  - ofstream (Ausgabe = Eingeben in die Datei)
  - ifstream (Eingabe = Einlesen aus Datei)

```
ofstream NAME_DES_DATEISTROMES
```

Mit dieser Variable wird ALLES gesteuert.

ErstesOfstream.cpp

2. Öffnen des Stromes

Die Stromvariable wird mit einer Datei verbindet.

Durch die Funktion (Methode) `open`

Hier ist wo der Name der Datei benutzt wird.

```
NAME_DES_DATEISTROMES.open (DATEINAME)
```

3. Datenübertragung

- Entweder zum Einlesen aus Datei >>
- oder zum Eingeben in Datei <<

```
NAME_DES_DATEISTROMES << AUSGABE
```

Schließen des Stromes

- Die Verbindung mit der Datei wird unterbrochen durch die Funktion `close()`

```
NAME_DES_DATEISTROMES.close ()
```

ErstesIfstream.cpp

Dateienströme: *fstream*

Die C++-Standardbibliothek stellt dem Programmierer eine Reihe von Strom-Klassen zur verfügung.

Die Kopfdatei `<iostream>` enthält Deklarationen von 8 Standardströme.
Will man selbst Strom-Objekte vereinbaren und mit Dateien verbinden, braucht man die Kopfdatei `<fstream>`.

Strukturen: Definition

Viele Programmieraufgaben lassen sich mit den so genannten Strukturen,einem speziellen Datenkonstrukt, leichter lösen.

Def1:Eine Struktur ist ein von Ihnen definierter Datentyp.

Strukturen: Definition

Def2:Eine Struktur ist eine Sammlung von einer oder mehreren Variablen,die zum Zweck der leichteren Manipulation unter einem Namen zusammengefasst werden.

Die Variablen in einer Struktur können im Gegensatz zu denen in einem Array unterschiedlichen Datentypen angehören. Eine Struktur kann jeden beliebigen C++-Datentyp enthalten.

Strukturen: die Grundbegriffe

-Struktur **definieren** und **deklarieren**.

-**Variablen** mittels Stuktur definieren.

-**Zugriff** auf Daten (**Komponenten**) in Strukturen.

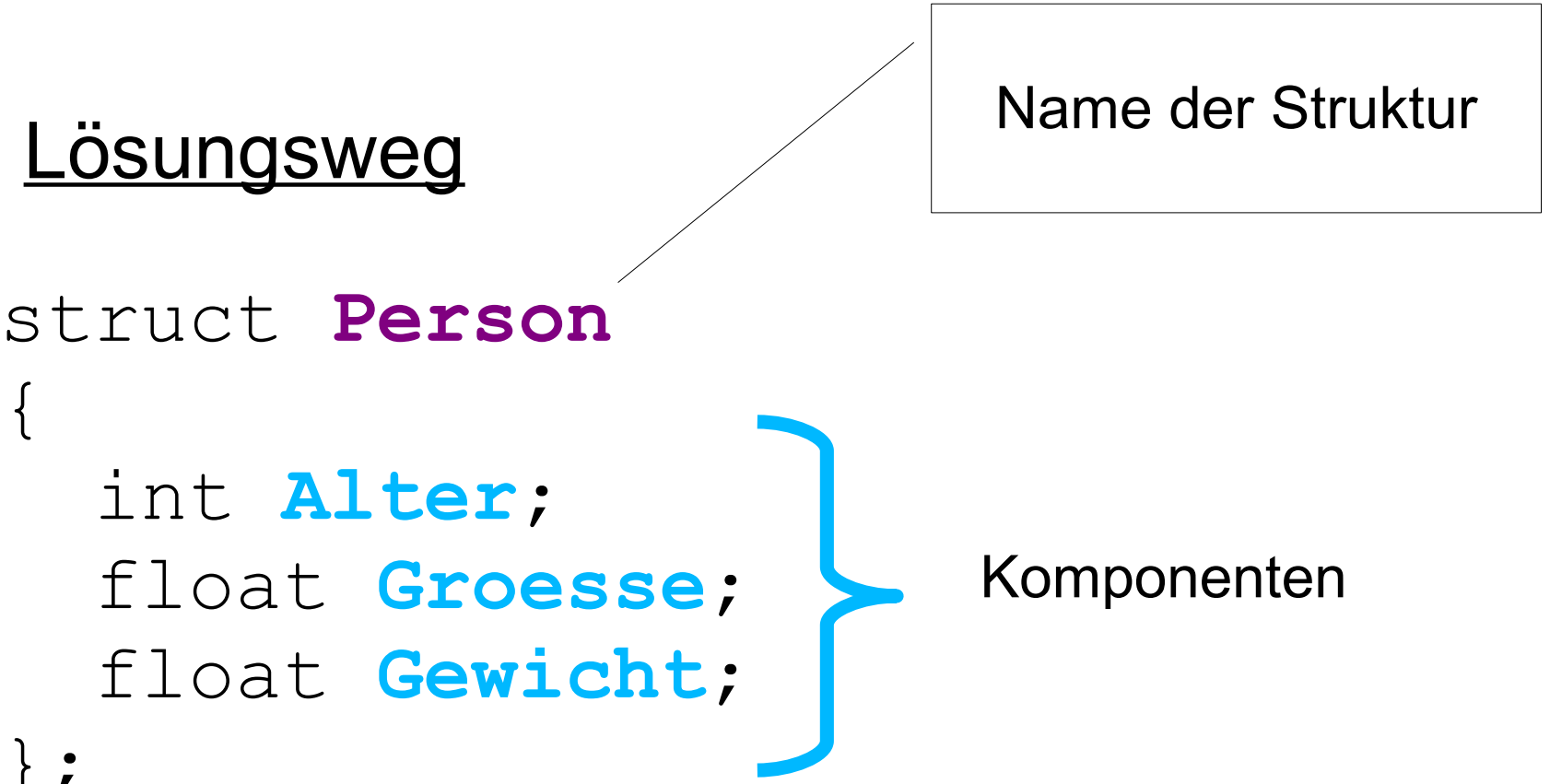
Struktur: Aufgabe

Aufgabe:
Es soll ein Programm geschrieben werden. Das Programm soll eine Struktur Name *Person* mit drei Elementen *Alter* ,*Größe* und *Gewicht* deklarieren.

Struktur deklarieren und definieren

Selbst definierte Datentypen
-Struct

Lösungsweg



Struktur deklarieren und definieren

Um eine Struktur zu deklarieren, braucht man:

-das Schlüsselwort **`struct`**.
-Den **Name** der Struktur.
-Die geschweiften Klammern.
-Die **Komponenten** der Struktur.
-Das Semikolon.

Struktur benutzen um Variablen zu vereinbaren

1.Selbstdefinierte Typen - struct

Schablone definieren → Schablone nutzen um Variablen zu definieren

```
struct person
{
 int ...;
 float ...;
 float ...;

};
```

```
person student, dozent,
   Arzt;
```

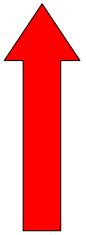
oder

```
struct person student, dozent,
          Arzt;
```

Auf Strukturen-Komponenten zugreifen:

Um Strukturen sinnvoll nutzen zu können,sollte man ihren Komponenten Werte zuweisen.

Zu diesem Zweck kennt C/C++
den **Punkt-Operator: .**



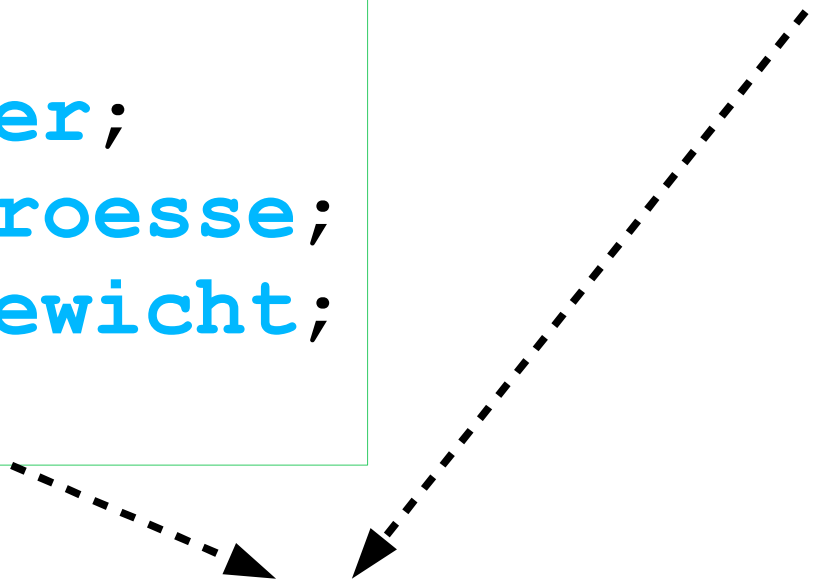
Auf Strukturen-Komponenten zugreifen:

DEFINITION

```
struct Person
{
   int Alter;
   float Groesse;
   float Gewicht;
};
```

DEKLARATION

```
person student;
```



```
student.Alter=18;
student.Gewicht=50.70;
student.Groesse=1.80;
```

personStruktur.cpp