I / O Streams (Eingabe- / Ausgabe-Datenströme)

- **Eingaben** in ein Programm können von der Tastatur oder aus einer sich auf einem externen Datenträger (Diskette, Festplatte, Wechseldatenträger) befindenden Datei erfolgen.
- **Ausgaben** können auf den Bildschirm oder in eine Datei erfolgen.
- Diese Eingaben / Ausgaben (I / O) erfolgen bei C++ über **Streams**. Streams sind ein erstes Beispiel für Objekte (Variablen eines speziellen Datentyps mit eigenen Funktionen).
- ***cin***, ***cout*** und ***cerr*** sind solche Streams, die in der Include-Datei ***iostream.h*** definiert sind.
- << und >> sind die Operatoren für die Ein- und Ausgabe von Daten in einen Stream.
- Man kann solche Datenströme auch selbst deklarieren.

1. Deklaration von Datenströmen

Die für das Arbeiten mit Datenströmen benötigten Funktionen befinden sich in der Include-Datei **fstream.h.** Die Standard-Datentypen (Klassen) sind **ifstream** und **ofstream.**

Beispiel: ***ofstream ausgabe;***
ifstream eingabe;

2. Dateien öffnen und schließen

Um aus einer Datei lesen oder in eine Datei schreiben zu können, muss man diese mittels der Funktion **open** zunächst öffnen, d. h. mit den Datenströmen verbinden.

Beispiel: ***ausgabe.open("a:\\filebsp.txt");***
eingabe.open("a:\\filebsp.txt");

Es geht aber auch so: ***ifstream eingabe("a:\\filebsp.txt");***

Nach Beendigung des Programms sollten die geöffneten Dateien mittels der Funktion **close** wieder geschlossen, d. h. die Verbindung mit den Streams gelöst werden.

Beispiel: ***eingabe.close();***
ausgabe.close();

Falls die Datei *filebsp.txt* nicht existiert, wird sie durch *open* erstellt. Falls die Ausgabe-Datei *filebsp.txt* schon existiert, wird sie überschrieben (Achtung, Fehlerquelle!).

3. Ausgabe von Daten in eine und Eingabe von Daten aus einer Datei

Die Ausgabe von Daten in eine Datei erfolgt mit dem Einfügeoperator <<, die Eingabe aus einer Datei mit dem Extraktionsoperator >>.

Beispiel:
```
int  i = 5, zahl;
eingabe  >>  zahl;
ausgabe  <<  i * i;
```

Beachte: Wenn ein Programm mit einer externen Datei arbeitet, dann hat diese Datei zwei Namen: den **externen** Dateinamen (z. B. *filebsp.txt*) und den **internen** Namen für die Arbeit innerhalb des Programms (z. B. *eingabe*).

4. Überprüfung des Öffnens von Dateien

Das Anlegen oder Öffnen einer Datei kann aus verschiedenen Gründen erfolglos sein, z. B. wenn der angegebene Pfad nicht existiert. Man erhält aber keine Fehlermeldung, und das Programm macht etwas Unvorhersehbares. Deswegen sollte man einen Test z. B. folgender Art einbauen:

```
if (! eingabe)
        {       cerr <<  "Oeffnen der Datei nicht moeglich\n";
                exit(1);        }
```

Die Funktion **exit** aus **stdlib.h** führt zum sofortigen Abbruch des Programms, als Parameter verlangt *exit* eine ganze Zahl. Gewöhnlich wird 1 verwendet, wenn im Programm ein Fehler auftritt und 0 sonst.

Aufgabe 1:

a) Interpretieren Sie folgendes Programm, erfassen Sie den Quelltext und testen Sie das Programm.

```
//streams1.cpp
#include <fstream.h>          //zum Arbeiten mit Datenstroemen
int main()
{
        int i;
        ofstream ausgabe;
        ausgabe.open("a:\\filebsp.txt");
        for (i = 1;  i <= 5;  i++)
                ausgabe <<  i*i << endl;
        ausgabe.close();
        return 0;
}
```

b) Lassen Sie sich mit Hilfe eines beliebigen Editors die Datei *filebsp.txt* anzeigen.

Aufgabe 2:

a) Interpretieren Sie folgendes Programm, erfassen Sie den Quelltext und testen Sie das Programm.

```
//streams2.cpp
#include <fstream.h>
int main()
{
        int i, s, zahl;
        ifstream eingabe("a:\\filebsp.txt");
        s = 0;
        for (i=1; i<=4; i++)
                {
                eingabe >> zahl;
                cout << zahl << endl;
                s = s + zahl;
                }
        cout << s;
        eingabe.close();
        return 0;
}
```

b) Ändern Sie mit Hilfe eines Editors die Datei *filebsp.txt* und starten Sie anschließend das Programm *streams2.cpp* erneut.

c) Arbeiten Sie den Fehlertest in das Programm ein.

Aufgabe 3:
Folgendes Programm zeigt, wie man den Namen einer externen Datei über Tastatur eingeben und Zeichen für Zeichen aus der Datei lesen kann. Erstellen und testen Sie das Programm (z. B. mit *k:\koehler\beispiel.txt*).

```
//textdat.cpp
#include <fstream.h>
#include <stdlib.h>          //fuer exit
int main()
{
        char ExtDat[20], zeichen;
        cout << "Name der Eingabedatei: ";
        cin >> ExtDat;
        cout << endl << endl;
        ifstream eingabe(ExtDat);
        if (! eingabe)
                {cerr << "Die Datei existiert nicht";
                 exit(1);}
        else    while (eingabe.get(zeichen)) cout << zeichen;
        eingabe.close();
        return 0;
}
```

Aufgabe 4:
a) Erstellen Sie mit einem einfachen Texteditor (z. B. WordPad) folgende Datei *matrix.txt*:

```
3
1       2       3
4       5       6
7       8       9
```

(In der ersten Zeile steht die Dimension der Matrix, in den nächsten Zeilen die Matrixelemente).

b) Fertigen Sie vom Programm *matrix.cpp* (Aufgabe 2 Seite 17) eine Kopie an und speichern Sie diese unter *matrixausdatei.cpp* ab. Wandeln Sie nun die Funktion *Eingabe* so ab, dass die Dimension der Matrix und die Matrixelemente nicht über Tastatur eingegeben sondern aus der Textdatei eingelesen werden.

c) Ändern Sie *matrix.txt* ab (andere Dimension und Werte) und testen Sie das Programm erneut.

Aufgabe 5:
a) Erstellen Sie analog zur Aufgabe 6 auf Seite 20 (Der strukturierte Datentyp *struct*) ein Programm, das eine Struktur *tMitglied* deklariert, die die Namen und Geburtsdaten von Familienmitgliedern enthält und diese auf einen externen Datenträger (z. B. Diskette) ausgibt. Hinweis: Benutzen Sie folgende Definition des Ausgabestroms:

```
ofstream ausgabe("a:\\GebDat.txt", ios::app);
/*hier wird ein Konstruktor des Objektes ausgabe aufgerufen mit 2 Parametern,
1. Ziel der Ausgabe, 2. Modus der Dateibearbeitung, hier ans Ende anhaengen*/
```

b) Erstellen Sie ein Programm, das die Geburtsdaten von dem externen Datenträger liest und auf dem Bildschirm ausgibt.

c) Schreiben Sie ein weiteres Programm, das nur die Namen der Familienmitglieder ausgibt, die in einem bestimmten Monat (z. B. September) Geburtstag haben.