1 Funktionszeiger

Jede Funktion wird beim Laden in den Speicher an eine bestimmte Adresse abgelegt. Im Programm wird diese Adresse durch den Funktionsnamen symbolisiert.

```
double Multipli(int wert, float zahl){
      return (wert * zahl);
}
```

```
int main(){
      const int zahl = 3;
      const float wert = 4.5;
      double rueckgabe;
      rueckgabe = Multipli(zahl, wert);
}
```

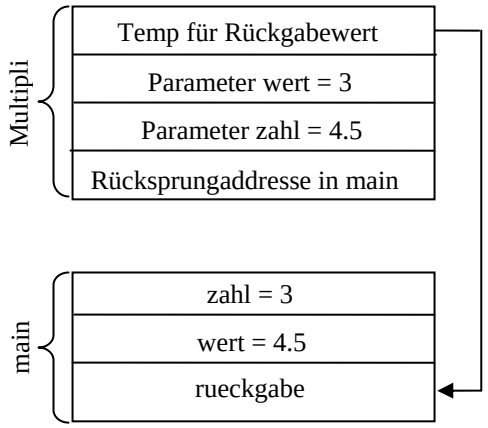

Abbildung 1: Speicherabbild

Mit Hilfe eines Zeigers kann ein Verweis auf die Adresse der Funktion erstellt werden.

```
double Multipli(int wert, float zahl){
      return (wert * zahl);
}
```

```
int main(){
      const int zahl = 3;
      const float wert = 4.5;
      double rueckgabe;
      double (*ptr)(int, float);
      ptr = Multipli;
      rueckgabe = (*ptr)(zahl, wert);
}
```

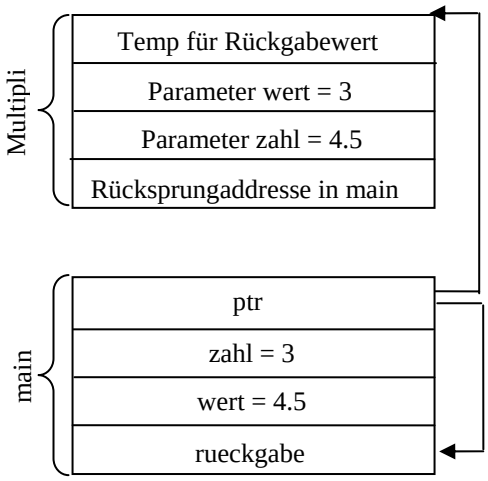

Abbildung 2: Nutzung von Funktionszeigern

Funktionszeiger werden zum Beispiel

- im Compilerbau,
- bei universellen Sortieralgorithmen und
- in der numerischen Mathematik

eingesetzt.

1.1 Funktionszeiger deklarieren

Die Deklaration eines Funktionszeigers gleicht der Deklaration einer Funktion

```
Datentyp (*pointerName)(Argument1, Argument2, … ArgumentN)
```

Als Datentyp kann jeder, in C++ definierter einfacher Datentyp genutzt werden.
Der Name des Zeigers ist frei wählbar. Das Sternchen definiert den variablen Namen als Zeiger. Ein Funktionszeiger benötigt Klammern um seinen Namen. Falls diese vergessen werden, wird eine Funktion vom `Datentyp*` definiert.
Der Datentyp und die Anzahl der Argumente sind abhängig von der Argumentliste der Funktion, auf die der Zeiger verweist.
Beispiele für die Deklaration von Zeigern:

`void (*funcPtr)()`	Der Funktionszeiger funcPtr zeigt auf eine Funktion, der keine Werte übergeben werden. Die Funktion selber besitzt keinen Rückgabewert.
`void (*funcPtr)(int, int)`	Der Funktionszeiger funcPtr kann auf eine Funktion zeigen, die als Argument zwei Variablen vom Datentyp Integer besitzt. Die Funktion selber gibt keinen Wert zurück.
`double (*funcPtr)(double, int)`	Der Funktionszeiger funcPtr enthält einen Verweis auf eine Funktion, der als Argumente eine Variable vom Datentyp double sowie eine Variable vom Datentyp int übergeben werden. Die Funktion selber gibt einen double-Wert zurück.
`bool (*funcPtr)()`	Der Funktionszeiger verweist auf eine Funktion, der keine Werte übergeben werden. Die Funktion selber gibt einen boolschen Wert zurück.

1.2 Bildung einer Funktionsadresse

Einem Funktionszeiger wird die Adresse einer Funktion mit Hilfe deren Namen zugewiesen:

```
pointerName = functionName.
```

Ein Funktionszeiger wird zuerst deklariert. Anschließend bekommt der Zeiger einen Funktionsnamen übergeben. Der Adressoperator & muss nicht dem Funktionsnamen vorangestellt werden. Der Funktionsname ist selber ein konstanter Zeiger auf eine Funktion.

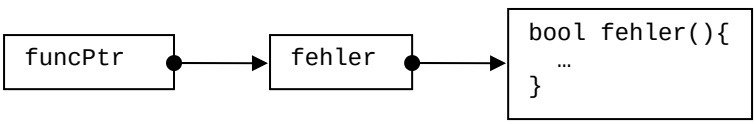

Abbildung 3 Konstanter Zeiger auf eine Funktion

1.3 Funktionsaufruf

Der Aufruf der Funktion erfolgt mit Zeigern ähnlich dem Aufruf über dem Funktionsnamen.

```
rueckgabeWert = (*funcPointer)(Arg1, Arg2, … ArgN)
```

Mit Hilfe des Dereferenzierungsoperators wird auf eine bestimmte Stelle im Speicher zugegriffen. Der dereferenzierte Funktionszeiger muss geklammert werden.