struct Konzept in C++

Daten bestehen häufig aus heterogenen Datentypen. Ein Ordnungskonzept, das zusammengehörige Elemente gruppiert, wäre beispielsweise bei einer Übergabe an Funktionen hilfreich. → Typ `struct` erlaubt einen nutzer-definierten, zusammengesetzten Datentyp.

- Syntax

```
struct StructName {
    Datentyp MyVar0 ;
    ……
    Datentyp MyVarN ;
} myStructVar ;
```

Name der Struktur

Angabe aller Datentypen und Variablen Namen

Variablen Name(n) der Struktur(en)

Ein fehlendes Semicolon erzeugt unverständliche Fehlermeldungen!

`struct` wird oft vor dem Hauptprogramm definiert

- Zugriff im Programm

```
StructName myStructVar;
myStructVar.MyVar0 = Val0;
…….
myStructVar.MyVarN = ValN;
```

- Die gesamte Struktur kann zugewiesen werden, keine Operatoren

```
StructName x,y,z;
……
z = x ;
```

struct Konzept in C++

- Zugriff im Programm auf `struct` Objekte

Operatoren für den Zugriff auf `struct` Elemente:

- Zugriff auf Mitglieder von Strukturen

`electron.mass;`

. (Punkt) Operator erlaubt den Zugriff auf Mitglieder von `struct`

- Zugriff auf Mitglieder über Zeiger auf Strukturen

`pelectron->charge;`

→ (Pfeil) Operator ist die Dereferenz auf Mitglieder für Zeiger auf `struct`

Ein äquivalenter Ausdruck ist

`(*pelectron).charge;`

Aber es ist nicht dasselbe wie

`*(pelectron.charge);`

(ein Zugriff über Zeiger auf Mitglieder von `struct`)

```
struct Particle{
    int charge;
    double mass;
    double energy;
};
…….
Particle electron ;
Particle *pelectron;
electron.mass ;
pelectron = &electron ;
pelectron->charge;
```

struct Konzept in C++

Es sind auch geschachtelte Strukturen erlaubt. Hier ist die Verwendung der Punkt und Pfeil Operatoren zu beachten.

- Zugriff im Programm auf geschachtelte `struct` Objekte

```
struct Wechselwirkung {
    char name[20];
    double masse;
} starke , elektroschwache ;
struct Teilchen{
    double ladung;
    double energy;
    Wechselwirkung austauschTeilchen;
} neutrino , proton ;
……
Teilchen *muonneutrino = &neutrino;
…….
neutrino.ladung;
proton.austauschTeilchen.masse;
muonneutrino→austauschTeilchen.name;
```

Struktur in einer Struktur!

Zugriff auf eine Struktur in einer Struktur!

struct Konzept in C++

Strukturen können direkt oder als Zeiger an Funktionen übergeben werden.

- Übergabe von `struct` Objekten an Funktionen

```
printTeilchen (struct Teilchen x){
cout<<“Charge=“<<x.ladung<<“ Energy =“<<x.energy<<endl;
}
printPointerTeilchen (struct Teilchen *p){
cout<<“Charge=“<<p->ladung<<“ Energy =“<<p->energy<<endl;
}
int main(){
  struct Teilchen{
     int ladung;
     double energy;
  } *pneutrino, proton;
  proton.ladung = 1;
  proton.energy = 20.;
  printTeilchen(proton);
  pneutrino = &proton;
  printPointerTeilchen(pneutrino);
  ……..
```

structFunc.cc

Das Verhalten von Programmen wird häufig durch externe Parameter gesteuert, die über Konfigurationsfiles eingelesen werden. Dabei wird jedem Parameter im Programm ein Schlüsselwort und ein Wert zugewiesen. Betrachten Sie folgendes Konfigurationsfile myConfigFile.txt

Arbeitsvorschlag:

Schreiben Sie ein Programm, das die Schlüsselwörter aus dem Konfigurationsfile liest und die Kommentare und leeren Zeilen unterdrückt. Die Schlüsselwörter und die zugehörigen Werte sollen in eine Struktur geschrieben werden.

1. Lesen Sie in einem Programm main() zunächst Zeile für Zeile des Konfigurations-files in einen string (siehe Beispiel FileIO.pdf, Seite 3).

2. Fügen Sie eine bool function hinzu, in der für die gelesenen strings, die aus Kommentar- und Leerzeilen bestehen, das flag false zurückgeben wird.

3. Definieren Sie ein globales string Array, das die Schlüsselwörter des Konfigurationsfiles enthält. Schreiben Sie nun eine integer function, die fuer jeden gelesenen string das Schlüsselwort durch Vergleich mit dem globalen string Array findet. Geben Sie die Position des gelesenen Schlüsselwortes im string array zurück. Im main() erlaubt das dann den zugehörigen Wert des Schlüssel-Wortes zu kopieren und in eine double Variable zu konvertieren.

4. Speichern Sie Schlüsselworte und Werte in einem Struktur Array und schreiben Sie eine print Funktion für die Werte der Struktur.

readConfig.cc