Fachhochschule Bingen

Programmieren

Union / Rekursion

Prof. Dr. Maximilian Mengel,
Professur Programmiermethodik,
Grundlagen der Informatik und Multimedia
Gebäude 1, Raum 212
Tel.: 06721-409 152
E-Mail: mengel@fh-bingen.de

Unionen

- Mit einer Union können unterschiedliche Datentypen alternativ in dem selben Speicherbereich abgelegt.
- Die Syntax ist analog zu einer Struktur, nur das statt des Schlüsselwortes **`struct`** das Schlüsselwort **`union`** benutzt wird
- Eine Union belegt im Speicher soviel Platz wie die größte Komponente Platz belegt
- Es sollte immer nur die genutzte Alternative angesprochen werden

Unionen

```
typedef union {
  long int lzahlen[10];
  short int szahlen[20];} t_zahlenfeld;

t_zahlenfeld meinZahlenfeld;

// entweder :
for (i=0;i<10;++i)
  meinZahlenfeld.lzahlen[i] = 0L;

// oder
for (i=0;i<20;++i)
  meinZahlenfeld.szahlen[i] = 0;
```


Induktive Definition mathematischer Funktionen

- In der Mathematik werden Funktionen oft induktiv definiert:

$$\text{Fak}(N) = \begin{cases} 1 & \text{wenn } N = 1 \\ N * \text{Fak}(N-1) & \text{wenn } N > 1 \end{cases}$$

$$\text{ggT}(A,B) = \begin{cases} A & \text{wenn } A = B \\ \text{ggT}(A, B-A) & \text{wenn } B > A \\ \text{ggT}(A-B, B) & \text{wenn } A > B \end{cases}$$

Induktive Definition

- Bei der induktiven Definition einer Funktion existieren mehrere zu unterscheidende Fälle
  - Fälle in denen eine direkte Lösung angegeben ist
  - Fälle in denen
    - Eine Vorschrift enthalten ist
    - Die Funktion selbst wieder benötigt wird
- Dieses Verfahren kann auch in der Programmierung zur Lösung von Problemen benutzt werden.

Rekursive Funktionen (direkte Rekursion)

- In einer rekursiven Funktion wird
  - In bestimmten Fällen das Ergebnis der Funktion direkt ausgerechnet bzw. bestimmt
  - In bestimmten Fällen die Funktion selbst wieder aufgerufen
  - Wird die Funktion selbst wieder aufgerufen, so muß dies so geschehen, daß sich der Aufwand zur Berechnung der Funktion verringert hat

Beispiel einer Rekursiven Funktion (in C)

- N!

```
int fak(int N)
{
   int erg;
   if (N < 2) erg = 1;
   else  erg = N * fak(N-1);
   return erg;
}

main()
{
   printf(" %d! = %d\n", 7, fak(7));
}
```


Beispiel einer Rekursiven Funktion (in C)

- ggT (A, B)

```
int ggt(int A, int B)
{
   int erg;
   if (A == B) erg = A;
   else
         if (A > B) erg = ggt(A-B, B);
         else erg = ggt(A, B-A);
   return erg;
}

main()
{
   printf("ggt(%d, %d) = %d\n", 140, 78, ggt(140,78));
}
```

Rekursive Prozeduren

- Neben rekursiven Funktionen zur Berechnung von mathematischen Funktionen existieren auch sonstige Probleme die elegant rekursiv lösbar sind
  - Ausgabe einer (Dezimal-)Zahl als Dualzahl
  - Die Türme von Hanoi

Horner Schema Rekursion

- Um eine Dezimalzahl als Dualzahl auszugeben kann folgende Prozedur benutzt werden

```
WriteDual(int A)
{
   if (A > 1)
   {
         WriteDual((int)(A / 2));
         WriteDual(A % 2);
   }
   else printf("%d",A);
}
```


Türme von Hanoi

```
void hanoi(int Anzahl, int *Start, int *Lager, int *Ziel)
{
   if (Anzahl == 1) move(Start,Ziel);
   else
   {
      hanoi(Anzahl-1, Start, Ziel, Lager);
      move(Start, Ziel);
      hanoi(Anzahl-1, Lager, Start, Ziel);
   }
}
```


Indirekte Rekursion

- Bei der indirekten Rekursion existieren mehrere Funktionen bzw. Prozeduren die sich wechselseitig aufrufen

```
int odd(int Zahl)
{
   if (Zahl = 1) return 1;
   else return even(Zahl-1);
}

int even(int Zahl)
{
   if (Zahl = 1) return 0;
   else return odd(Zahl-1);
}
```

Anwendung Backtracking

- Finden einer Lösung für ein gegebenes Problem der Größe N
  - A) Festlegen eines möglichen Lösungsschrittes
  - B) Rekursiver Aufruf mit der Problemgröße N-1
  - C) Überprüfen ob Erfolg
    - Ja: Lösung gefunden
    - Nein: Lösungsschritt revidieren und anderen Lösungsschritt ausprobieren => A)
  - Wenn kein Lösungsschritt zum Erfolg führt, dann ist das Problem nicht lösbar
  - Beispiele: Damenproblem, Labyrinth