

Programmieren in C 1: Wiederholung & Beispielklausur PR1

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 1

Programmieren in C 2

- In diesem Kapitel wollen wir:
 - ➔ eine Beispielklausur PR1 durchsprechen
 - ➔ dabei einige Grundlagen aus der Vorlesung PR1 wiederholen.
 - ➔ und auf eventuell aufgetretenen Fragen eingehen.

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 2

**Aufgabe 1: Berechnung von Ausdrücken 1**

Gegeben sind Anweisungen und Ausdrücke in der Programmiersprache C. Verwenden Sie dabei jeweils die folgenden vorgegebenen Werte der Variablen.

`int i = 8, j = 5; float x = 0.005, y = -0.01; char c = 'c', d = 'd';`
Berechnen Sie die Werte der folgenden Ausdrücke:

1.1. $(3 * j - 2 * i) \% (2 * d - c)$

$= (3 * 5 - 2 * 8) \% (2 * 100 - 99) = (-1) \% 101 = -1$

1.2. $2 * ((j / 6) + (4 * (i - 3)) \% (i - j + 2))$

$= 2 * ((5 / 6) + (4 * (8 - 3)) \% (8 - 5 + 2)) = 2 * (0 + 20 \% 5) = 0$

1.3. $-(i - j)$ $= -(8 - 5) = -3$

1.4. $i++$ $= i + 1 = 9$ (nach der Anweisung)

1.5. $--i$ $= i - 1 = 7$ (vor der Anweisung)

1.6. $++y$ $= -0.01 + 1 = 0.99$ (vor der Anweisung)

1.7. $i < j$ $= 0$ (false)

1.8. $x == 0$ $= 0$ (false)

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 3

**Aufgabe 1: Berechnung von Ausdrücken 2**

Gegeben sind Anweisungen und Ausdrücke in der Programmiersprache C. Verwenden Sie dabei jeweils die folgenden vorgegebenen Werte der Variablen.

`int i = 8, j = 5; float x = 0.005, y = -0.01; char c = 'c', d = 'd';`

Berechnen Sie die Werte der folgenden Ausdrücke:

1.8. $x == 0$ $= 0$ (false)

1.9. $(2 * x + y) >= 0$ $= 1$ (true)

1.10. $c \wedge d$ $= 0110\ 0011 \wedge 0110\ 0100 = 0111 = 0x7$

1.11. $c \parallel d$ $= 1$ (true)

1.12. $(x < y) \&\& (i > 7) \&\& (j <= 5)$ $= 0 \& 1 \& 1 = 0$ (false)

1.13. $!(c == 99)$ $= 0$ (false)

1.14. $2 * c \mid d$ $= 1100\ 0110 \mid 0110\ 0100 = 1110\ 0110 = 0xe6$

1.15. $(\text{short}) (y / x)$ $= (\text{short}) (-0.01 / 0.005) = -2$

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 4



Aufgabe 2: C-typische Schreibweise

Es gilt als guter C-Programmierstil, Ausdrücke möglichst kompakt zu schreiben.

Geben sie für folgende Ausdrücke jeweils eine C-typische Kurzschreibweise an:

- | | |
|---|---|
| 2.1 <code>if (Z != 0) break ;</code> | <code>if (Z) break ;</code> |
| 2.2 <code>if (a > b) Max = b ;
 else Max = a ;</code> | <code>Max = (a > b) ? b : a ;</code> |
| 2.3 <code>Y = (2*A / t) * Y - 3 * Y * Y ;</code> | <code>Y *= (2*A / t) - 3 * Y ;</code> |
| 2.4 <code>A = A >> 3 ;</code> | <code>A >>= 3 ;</code> |
| 2.5 <code>while (X != 0) X = X + 1;</code> | <code>while (X++) ... ;</code> |
| 2.6 <code>do (n = n + 1) while (s [n] == 0) ;</code> | <code>while (!s [n++]) ... ;</code> |

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 5



Aufgabe 3: Programmfehleranalyse 1

Das folgende C-Programm enthält Fehler!

Zur Vereinfachung der Fehleranalyse sind die Zeilen nummeriert.

Untersuchen Sie das Programm auf logische, syntaktische Fehler, Formatfehler und Ausdrücke, die offensichtlich nicht dem ANSI-C Standard entsprechen!

Erläutern Sie bei jedem Fehler die Fehlerursache!

Verwenden Sie als Bezug für die Fehlerangaben die Zeilennummer des Programmbefehls!

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 6

**Aufgabe 3: Programmfehleranalyse 2**

```

1.
2.
3. void main(void)
4. {
5.     int ok;
6.     unsigned short Summe = 0, x;
7.     long Anzahl = 0;
8.     clrscr ( );
9.     do
10.    {
11.        printf ( "Messwert im short-Format = ?: " ); ok = scanf ("%f",x);
12.        if (!ok) break;
13.        Anzahl--;
14.        Summe = Summe + x;
15.        printf ( "Die Summe der %u Messwerte ergibt: %u /n", Anzahl,
16.                Summe);
17.    }
18.    while (Anzahl < 4e6);
19.

```

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 7

**Aufgabe 3: Programmfehleranalyse 1**

```

1. #include <stdlib.h>
2. #include <stdio.h>
3. void main(void)
4. {
5.     int ok;
6.     unsigned short Summe = 0, x;
7.     long Anzahl = 0;
8.     fflush ( stdin ) ; system ( "cls" ) ;
9.     do
10.    {
11.        printf ( "Messwert im short-Format = ?: " ); ok = scanf ("%f",x);
12.        if (!ok) break;
13.        Anzahl--;
14.        Summe = Summe + x;
15.        printf ( "Die Summe der %u Messwerte ergibt: %u \n", Anzahl,
16.                Summe);
17.    }
18.    while (Anzahl < 4e6);
19.    return 0;
20. }

```

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 8



Aufgabe 3: Programmfehleranalyse 2

1. `clrscr ()` gibt es in MSVC nicht (vgl. 8.) Bibliothek `stdlib.h` nötig
2. Die Funktionen `printf ()`, `scanf ()` benötigen die Bibliothek `stdio.h`.
3. `void main(void)` in ANSI-C nicht erlaubt: `int main (void)`
6. `unsigned short Summe` : ungeeignetes Format, ruft u.U. Zahlbereichsüberlauf hervor, besonders im Hinblick auf: `long Anzahl`
8. `clrscr ()`; : siehe auch 1.: `fflush (stdin) ; system ("cls") ;`
11. `printf / scanf` : siehe auch 2.!
11. Meßwert im `unsigned short`-Format
11. Adreßoperator bei `scanf` fehlt: `&x`
13. `Anzahl--`; : falsch: `Anzahl++`;
15. `%u` ist falsches Format für `Anzahl`
15. `\n` statt `/n` als Zeilenvorschub
17. `while (Anzahl < 4e6)`; : obere Grenze zu hoch: Zahlbereichsüberlauf (siehe auch 6.!)
18. `return 0`; fehlt
19. `}` fehlt



Aufgabe 4: Felder und Schleifen

Untersuchen Sie den folgenden Programmausschnitt!

....

```
short i = 0;
float array [6] = { 0, 1, 2, 3, 4, 5 };
while ( i < 4 ) array [++i]--;
```

...

Geben Sie in der richtigen Reihenfolge den Wert oder die Werte an, welche die jeweiligen Variablen annehmen!

Aus Ihrer Lösung soll erkennbar sein, wie Sie zu den Werten kommen!





Aufgabe 4: Felder und Schleifen 1

Der Präinkrement-Operator `++i` erhöht *i* bevor *i* in der Anweisung verwendet wird.

Der Postdekrement-Operator `array [ ... ]--` erniedrigt das Feldelement nachdem es in der Anweisung verwendet worden ist.

Vorsicht beim Test von Programmen:

Verwenden Sie mit Funktionen keine Ausdrücke beispielsweise der Form:

```
printf("%d %d", ++n, power(2,n))
```

Ein solcher Ausdruck liefert je nach Compiler unter Umständen unterschiedliche Ergebnisse, weil die Reihenfolge der Auswertung in C nicht festgelegt ist!

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 11



Aufgabe 4: Felder und Schleifen 2

• Programmabschnitt

```
short i = 0 ;
float array [6] = { 0, 1, 2, 3, 4, 5 } ;

while ( i < 4 ) array [ ++i ] -- ;
```

• Schleifendurchläufe

1. <code>i = 0 < 4</code> <code>++i = 1</code> <code>array [1]-- = 0</code>	2. <code>i = 1 < 4</code> <code>++i = 2</code> <code>array [2]-- = 1</code>
3. <code>i = 2 < 4</code> <code>++i = 3</code> <code>array [3]-- = 2</code>	4. <code>i = 3 < 4</code> <code>++i = 4</code> <code>array [4]-- = 3</code>
5. <code>i = 4 >= 4</code> : Schleifenende	

⇒ `float array [6] = { 0, 0, 1, 2, 3, 5 } ;`

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 12



Aufgabe 5: rekursive und iterative Funktionen

Schreiben Sie eine C-Funktion zur Berechnung von

```
float a ;
long double y ;
unsigned char i ;
```

$$y = a \cdot (i!) \quad i > 0$$

Schreiben Sie das Unterprogramm als:

1. iterative Funktion
2. rekursive Funktion!

Eventuell mögliche Zahlbereichsüberläufe sollen nicht beachtet werden!

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 13



Aufgabe 5: iterative Funktion

$i > 0$ wird vorausgesetzt.

```
long double iterativ ( float a, unsigned char i )
{
    long double y = 1 ;
    short j ;
    for ( j=1; j<=i; j++) y *= j ;
    return a*y ;
}
```

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 14

✓ Aufgabe 5: *rekursive Funktion*

$i > 0$ wird vorausgesetzt.

```
long double rekursiv ( float a, unsigned char i )
{
    if ( i == 1) return a ;
    else
        return rekursiv (a, i-1) * i ;
}
```

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 15

**Viel Erfolg bei der
Praktikumsprüfung am 19.1.2005
und bei der
PR1-Klausur am 24.1.2005!**

**Weitere Informationen finden Sie in
den Aushängen des Fachbereichs
Elektrotechnik und Informatik und auf
meiner Homepage im Internet!**

02.12.2004

Hochschule für angewandte Wissenschaften Hamburg
Prof.Dr.Ing.U.Vogelsang, Prof.Dr.U.Sauvagerd



Programmieren in ANSI C:
K9_C-Klausur PR1.ppt, Seite 16