



Note / normierte Punkte

Klausur in Programmieren

Sommer 2012, 18. Juli 2012
Dauer: 1,5h
Hilfsmittel: Keine (Wörterbücher sind auf Nachfrage erlaubt)

Name:
Matrikelnr.:

Aufgabe	1	2	3	4	5	6	Summe
Punkte max	8	15	20	19	23	15	100
Punkte							

Alle Fragen beziehen sich auf den Stoff der Vorlesung. Somit sind sie z.B. bezogen auf die Programmiersprache C++. Auch sonst gelten die Konventionen wie in unserer Vorlesung.

1. Aufgabe: Grundlagen

a) Geben Sie an, welcher der C/C++ Ausdrücke eine Variablendefinition (F) und welche eine Variablendeklaration (D) ist. (4 P):

- 1) #define A 5 _____
- 2) double d = 9.81; _____
- 3) int d; _____
- 4) cout << "Definition: "; _____

b) Welche Werte enthalten die nachfolgenden Variablen: (4 P)

- 1) int i = 25.7 * 2.0; i == _____
- 2) double d = 3.5 * 3; d == _____
- 3) int i = 7;
double d = ((i++ * ((i + 1) % 3)) - 1) / 2.0; i == _____, d == _____

2. Aufgabe: Grundlagen

a) Schreiben Sie ein Hauptprogramm, das die potentielle Energie eines Körpers berechnet. Lesen Sie dazu von der Konsole die Masse m in [kg] sowie die Höhe der Position des Körpers h in [m] ein. Die Erdbeschleunigung definieren Sie als globale Konstante mit dem Wert 9.81 in [m / s²]. Berechnen Sie nun mit der Formel $E = m * g * h$ die potentielle Energie des Körpers, speichern das Ergebnis in einer passenden Variablen und geben anschließend diesen Wert auf die Konsole aus. (9 P)

b) Welche Schleifen gibt es in C/C++? Geben Sie kleine Beispiele: (6 P)

3. Aufgabe: Funktionen

a) Erklären Sie aus welchen Teilen eine Funktion besteht (strukturiert vorgehen!) (6 P):

b) Geben Sie ein Beispiel für eine Funktion, bei der genau ein Parameter mittels „call by value“ übergeben wird und ein double Ergebniswert mittels return zurückgegeben wird (nur ein Programmfragment, keine zusätzliche main-Routine, keine Ein-/Ausgabe) (4 P):

c) Geben Sie ein Beispiel, bei der eine Zeichenkette (Array mit Basistyp char) mittels „call by reference“ übergeben wird. Es soll kein Rückgabewert zurück gegeben werden (nur ein Programmfragment, keine zusätzliche main-Routine, keine Ein-/Ausgabe) (4 P).

d) Schreiben Sie eine vollständige Funktion, bei der die Summe von einem integer-Startwert bis zu einem integer-Endwert berechnet wird. Start- und End-Wert werden als Parameter übergeben, Ergebnis als Rückgabewert der Funktion (nur ein Programmfragment, keine zusätzliche main-Routine, keine Ein-/Ausgabe) (6 P).

4. Aufgabe: Array/Feld, Indizierung

a) Erklären Sie den Unterschied zwischen einem statischen und einem dynamischen Feld und was dabei generell zu beachten ist. Geben Sie jeweils ein einfaches Beispiel mit der Deklaration eines Feldes mit 10 int-Feldelementen: (10 P)

Statisches Feld:

Dynamisches Feld:

b) Gegeben ist folgender Funktionskopf: `int maximum(int* aInDaten, unsigned int uInAnzahl)`
Schreiben Sie den dazu passenden Funktionsrumpf, der als Rückgabewert den größten Wert des Feldes zurück gibt. Der Parameter `uInAnzahl` ist dabei immer größer 0 und enthält die Anzahl der Feldelemente. Sie können davon ausgehen, dass das Feld `aInDaten` auch genau so viele Feldelemente enthält, wie `uInAnzahl` angibt. (9 P)

5. Aufgabe: Zeichenketten

a) Schreiben Sie eine Funktion `strlen`, die die Länge einer mit 0 terminierten Zeichenkette bestimmt. Der Funktionswert soll die Länge zurückgeben (kein Hauptprogramm, keine Ein- oder Ausgabe!). (8 P)

b) Schreiben Sie eine Funktion `charCount`, die bestimmt, wie oft ein als Parameter übergebenes Zeichen in einer Zeichenkette vorkommt. Kommt das Zeichen gar nicht vor, soll 0 zurückgegeben werden (kein Hauptprogramm, keine Ein- oder Ausgabe!). (15 P)

6. Aufgabe: Algorithmus

Was berechnet die nachfolgende Funktionen unknown1? Und wozu dienen die Funktionen unknown2 und unknown3?

Bitte beschreiben Sie die Funktionsweise möglichst abstrakt – Romane geben Abzug! (15 P)

```
int unknown1(char* acInChars1, char* acInChars2)
{
    int iIndex = 0;
    while(acInChars1[iIndex] == acInChars2[iIndex] && acInChars1[iIndex] != 0)
    {
        iIndex++;
    }
    if(acInChars1[iIndex] == acInChars2[iIndex])
    {
        return 0;
    }
    else
    if(acInChars1[iIndex] < acInChars2[iIndex])
    {
        return -1;
    }
    return 1;
}

void unknown2(char* acInChars1, char* acInChars2, int iInValue)
{
    cout << "unknown1(' " << acInChars1 << "', '" << acInChars2 << "'): ";
    if(unknown1(acInChars1, acInChars2) == iInValue)
    {
        cout << "NOT " << endl;
    }
    cout << " FAILED!" << endl;
}

void unknown3()
{
    unknown2("Alfons", "Alfred", -1);
    unknown2("Alfons", "Alfons", 0);
    unknown2("Alfred", "Alfons", 1);
}
```