



Klausur in Programmieren

Sommer 2010, 19. Juli 2010

Dauer: 1,5h

Hilfsmittel: Keine (Wörterbücher sind auf Nachfrage erlaubt)

Name:

Matrikelnr.:

Aufgabe	1	2	3	4	5	6	Summe
Punkte max	12	9	15	19	30	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

Kreuzen Sie an, was in der Sprache C/C++ ohne zusätzliche Definitionen zutrifft. Falsche Aussagen geben einen Punkt Abzug. (0 ... 12P):

	korrekt	/ nicht korrekt
a) <code>int x := 4;</code>	<input type="radio"/>	<input type="radio"/>
b) <code>char* z = "Hello";</code>	<input type="radio"/>	<input type="radio"/>
c) <code>constant e = 2.1;</code>	<input type="radio"/>	<input type="radio"/>
d) <code>int i = 8 > 7 ? 6 % 3 : 1;</code>	<input type="radio"/>	<input type="radio"/>
e) <code>long int li = (int) 9.123;</code>	<input type="radio"/>	<input type="radio"/>
f) <code>real c = 2.1;</code>	<input type="radio"/>	<input type="radio"/>
g) <code>short int si = 417365876 >> 4;</code>	<input type="radio"/>	<input type="radio"/>
h) <code>char c = 044;</code>	<input type="radio"/>	<input type="radio"/>
i) <code>double d = 3,4;</code>	<input type="radio"/>	<input type="radio"/>
j) <code>int h = 0xABCdEF;</code>	<input type="radio"/>	<input type="radio"/>
k) <code>const int x = (8 * 8) * ((2 + 5) - 6 * 3 + (2+2));</code>	<input type="radio"/>	<input type="radio"/>
l) <code>int hex = 0#acbef;</code>	<input type="radio"/>	<input type="radio"/>

- Bitte beachten Sie auch die Rückseite -
- Lösen Sie die Aufgaben bitte auf dem Blatt -

2. Aufgabe: Grundlagen

Schreiben Sie ein Hauptprogramm, das die Kantenlänge a eines Quadrates und den Radius r eines Kreises beides als reellen Wert von der Konsole (über `cin`) einliest. Berechnen Sie anschließend beide Flächen (a^2 , πr^2) zumindest näherungsweise (definieren Sie π als globale Konstante) und geben Sie anschließend nur die eine Meldung aus, welche der beiden Flächen größer ist. (9P)

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: Ganzes Programm / Automatische Tests

Ergänzen Sie die Lücken im nachfolgenden Programmtext zu einem lauffähigen C++-Programm. Es hat die Aufgabe, eine Funktion zu testen – `numberOf` -, die die Anzahl eines Zeichens in einer Zeichenkette zählt. Die Zeichenkette ist durch eine abschließende 0 begrenzt. Falls alle Testfälle ohne Fehler durchlaufen werden, soll die Anzahl aller Kleinbuchstaben in einem dazu passenden Array gespeichert und anschließend ausgegeben werden. (30P)

```
#include <_____>

using _____;

// Anzahl des Buchstaben cInChar in der Zeichenkette acInString zählen
// und als Funktionswert zurückgeben

_____ numberOf(char* acInString, char cInChar)
{
    int iCount = _____;

    for(int li=___; _____ != 0; _____)
    {
        if(acInString[li] == _____)
        {
            _____;
        }
    }
    return _____;
}

// Funktion zur Anwendung eines Testfalls um die Korrektheit von numberOf
// zu prüfen. Gibt numberOf den erwarteten Wert zurück, wird von
// testCaseNumberOf true zurück gegeben, false im Fehlerfalle

_____ testCaseNumberOf(char* acInString, char cInChar,
                        int iInExpectedResult)
{
    _____ iResult = numberOf(_____, _____);

    cout << "Test: Anzahl " << cInChar << " in " << acInString << ": "
         << _____ << " ";

    bool bCheck = (iResult == iInExpectedResult);
    if(bCheck)
    {
        cout << "OK!" << endl;
    }
    else
    {
        cout << "NICHT OK! Erwartet: " << _____ << endl;
    }
    return bCheck;
}
```

- Bitte beachten Sie auch die Rückseite -
- Lösen Sie die Aufgaben bitte auf dem Blatt -

```

// Festlegung aller Testfälle für die Funktion numberOf
// Rückgabewert: Anzahl aller .fehlgeschlagenen Tests
int testNumberOf()
{
    int iErrorCount = 0;    // Anzahl aller aufgetretenen Fehlerfälle

    // Anzahl des Buchstaben 'l' mit testCaseNumberOf testen,
    // nur bei Rückgabe von false iErrorCount hochzählen

    if(_____("Hallole liebe Leut!", _____, _____))
    {
        _____;
    }

    // Anzahl des Buchstaben 'e' mit testCaseNumberOf testen,
    // nur bei Rückgabe von false iErrorCount hochzählen

    if(!_____("Hallole liebe Leut!", _____, _____))
    {
        _____;
    }
    cout << endl;
    cout << "Aufgetretene Fehler gesamt: " << _____ << endl;
    return iErrorCount;
}

// Zählen aller Kleinbuchstaben in der angegebenen Zeichenkette
void countCharacters()
{
    const int ciSize = 'z' - 'a' + 1;
    int aiCount[ciSize];
    for(char cFind='a'; cFind <= 'z'; ++cFind)
    {
        // Anzahl des Buchstaben cFind mit numberOf zählen

        aiCount[cFind - 'a'] = _____("Anzahl der Kleinbuchstaben.", _____);
    }
    for(int li=0; li<ciSize; ++li)
    {
        cout << " " << (char)('a'+li) << ": " << aiCount[li] << endl;
    }
}

int main()
{
    if(testNumberOf() == _____)
    {
        // alle Tests ok! rufe countCharacters auf

        _____;
    }

    // Programm erst nach Betätigen der Enter-Taste beenden

    _____;

    return 0;
}

```

- Bitte beachten Sie auch die Rückseite -
- Lösen Sie die Aufgaben bitte auf dem Blatt -

6. Aufgabe: Algorithmus

Was berechnen die nachfolgenden Funktionen `unknown1` und `unknown2` bzw. wann liefert `unknown2` den Wert `true` und wann den Wert `false`?

Bitte beschreiben Sie die Funktionsweise möglichst abstrakt. (15 P)

Hinweis: Testen Sie den Algorithmus anhand einer Zeichenkette, z.B. „Beatles“, „Abba“, „Scooter“ und beobachten Sie die Variablenwerte.

```
char unknown1(char cInChar)
{
    const int ciDiff = 'a' - 'A';
    if('a' <= cInChar && cInChar <= 'z')
    {
        return cInChar - ciDiff;
    }
    return cInChar;
}

bool unknown2(char* acInString)
{
    int iValue = 0;
    while(acInString[iValue] != 0)
    {
        ++iValue;
    }
    for(int li=0; li<iValue/2; ++li)
    {
        if(unknown1(acInString[li]) != unknown1(acInString[iValue - li - 1]))
        {
            return false;
        }
    }
    return true;
}
```