Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Studiengang Sensorik/Sensorsystemtechnik

Note / normierte Punkte

Klausur in Programmieren

Sommer 2009, 16. Juli 2009

Dauer: 1,5h

Hilfsmittel: Keine (Wörterbücher sind auf Nachfrage erlaubt)

Name:

Matrikelnr.:

Aufgabe	1	2	3	4	5	6	Summe
Punkte max	12	6	20	16	26	15	95
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

Welche Ergebnisse haben die folgenden Operationen auf den gegebenen Definitionen? *Die Operationen der einzelnen Teilaufgaben bauen nicht aufeinander auf!*

```
int     a = 5;
int     b = 18;
```

1. `int c = b++;` Variablenwerte: b = __19_, c = __18_ (2 Pkte)

2. `int c = ++a;` Variablenwerte: a = __6__, c = __6__ (2 Pkte)

3. `int c = --a / 5;` Variablenwerte: a = __4__, c = __0__ (2 Pkte)

4. `int c = b % a++;` Variablenwerte: a = __6__, b = __18_, c = __3__ (3 Pkte)

5. `int c = ++b % a;` Variablenwerte: a = __5__, b = __19_, c = __4__ (3 Pkte)

2. Aufgabe: Datentypen

Welche Ausgabe produzieren die folgenden Programmfragmente (6P):

a)

```
short int a = 32767;     // short int    2 Byte
a++;
if(a > 0)
{
     std::cout << "Wert groesser 0. " << endl;
}
else
{
     std::cout << "Wert kleiner oder gleich 0. " << endl;
}
```

Ausgabe: ***Wert*****.....*****kleiner oder gleich*****.....*****0.**

b)

```
short int b = -32768;    // short int    2 Byte
b++;
if(b > 0)
{
     std::cout << "Wert groesser 0. " << endl;
}
else
{
     std::cout << "Wert kleiner oder gleich 0. " << endl;
}
```

Ausgabe: ***Wert*****..... *****kleiner oder gleich*****.....*****0.**

3. Aufgabe: Array/Feld, Indizierung

a) Für Zeichenketten gibt es unterschiedliche Handhabungen. Die üblichsten Vorgehensweisen bauen auf Arrays mit dem Datentyp char auf. Meist füllen die Zeichenketten jedoch nicht das gesamte Array, sodass die Länge oder das Ende einer Zeichenkette gesondert betrachtet werden muss. Es sind dabei zwei Varianten üblich:

1. die Markierung des Endes einer Zeichenkette mit einer 0.

2. die Speicherung der Länge an der 1. Stelle des Arrays.

a) Nennen Sie je Methode einen Vor- und einen Nachteil bei der jeweiligen Vorgehensweise: (4P)

Vorteil bei 1:
Unbegrenzte Länge

Nachteil bei 1:
0 als Zeichen ist nicht verfügbar, die Länge muss durch Suchen der Endemarkierung festgestellt werden.

Vorteil bei 2:
Länge der Zeichenkette ist sofort verfügbar, alle Zeichen können verwendet werden.

Nachteil bei 2:
maximale Länge der Zeichenkette ist auf 255 begrenzt

b) Schreiben Sie eine Funktion, die eine Zeichenkette der Variante 2 (acInStringV2) in eine Zeichenkette der Variante 1 (acOutStringV1) umwandelt. Verwenden Sie dabei den nachfolgend vordefinierten Funktionskopf und ergänzen Sie den Funktionsrumpf: (16P)

```
void convertString(char* acOutStringV1, char* acInStringV2)
{
  int iLength = (int)acInStringV2[0];
  for(int li=0; li < iLength; ++li)
  {
    acOutStringV1[li] = acInStringV2[li+1];
  }

  acOutStringV1[iLength] = 0;
}
```

4. Aufgabe: Schleifen

a) Welche Schleifentypen gibt es in C/C++? Umreißen Sie in kurzen Stichworten die Unterschiede. (6P)

1. while-Schleife, kopfgetestete Schleife, Schleifenbedingung wird geprüft bevor der Schleifenrumpf ausgeführt wird.

2. do-while-Schleife, endegetestete Schleife, Schleifenbedingung wird nach der Ausführung des Schleifenrumpfes getestet

3. for-Schleife, Zählschleife, meist Verwendung, wenn die Anzahl der Durchläufe vor dem Start der Schleife bekannt ist.

b) Schreiben Sie nachfolgende while-Schleife als for-Schleife mit derselben Bedeutung (10P):

```
int li = 10;
int lSumme = 0;
while(li >= 0)
{
     std::cout << "Zwischensumme: " << lSumme << endl;
     lSumme += li;
     li--;
}
```

```
int lSumme = 0;
for(int li = 10; li >= 0; li--)
{
     std::cout << "Zwischensumme: " << lSumme << endl;
     lSumme += li;
}
```

5. Aufgabe: Ganzes Programm

(26 P)

Ergänzen Sie die Lücken im nachfolgenden Programmtext zu einem lauffähigen C++-Programm. Es hat die Aufgabe meteorologische Temperaturangaben einzulesen, daraus den Mittelwert und die Varianz zu bestimmen und auf die Konsole auszugeben. Der Mittelwert und die Varianz bestimmen sich durch

$$mw = \frac{1}{n}\sum_{i=0}^{n-1} x_i \,, \qquad varianz = \frac{1}{n}\sqrt{\sum_{i=0}^{n-1}\left(x_i - mw\right)^2}$$

```
#include <math.h>
#include <iostream>

using namespace std;


// Messwerte von der Konsole einlesen
void eingebenMesswerte(unsigned int& ruiOutMesswerteAnzahl, double*& rpdOutMesswerte)
{
   cout << “Anzahl Messwerte: “;
   cin >> ruiOutMesswerteAnzahl;
   rpdOutMesswerte = new double[ruiOutMesswerteAnzahl];

   for(unsigned int li=0; li<ruiOutMesswerteAnzahl; ++li                    )
   {
      cout << “Messwert: “;
      cin >> rpdOutMesswerte[                li       ];
   }
}

// Mittelwert bestimmen
double rechneMittelwert(unsigned int uiInMesswerteAnzahl, double* adInMesswerte)
{
   double dMittelwert = 0.0;
   for(unsigned int li=0; li<uiInMesswerteAnzahl; ++li)
   {
      dMittelwert += adInMesswerte[ li ];
   }

   return dMittelwert / uiInMesswerteAnzahl;


}
```

Kommentar [KM1]: 1 P

Kommentar [KM2]: 1 P

Kommentar [KM3]: 1 P

Kommentar [KM4]: 2 P

Kommentar [KM5]: 1 P

Kommentar [KM6]: 1 P

Kommentar [KM7]: 1 P

Kommentar [KM8]: 2 P

Kommentar [KM9]: 1 P

Kommentar [KM10]: 3 P

Kommentar [KM11]: 2 P

```
// Varianz bestimmen
double rechneVarianz(double dInMittelwert,
                     unsigned int uiInMesswerteAnzahl, double* adInMesswerte)
{
   double dSumme = 0.0;

   for(unsigned int li=0;  li < uiInMesswerteAnzahl;  ++li          )
   {
      double dDiff = dInMittelwert – adInMesswerte[   li      ];

      dSumme +=  dDiff *  dDiff                          ;
   }


   return  sqrt(dSumme)   /  uiInMesswerteAnzahl          ;
}



int main()
{
   double* adMesswerte;
   unsigned int uiMesswerteAnzahl;

   eingebenMesswerte(  uiMesswerteAnzahl, adMesswerte                          );

   double dMittelwert =   rechneMittelwert(  uiMesswerteAnzahl, adMesswerte )          ;

   double dVarianz = rechneVarianz(dMittelwert,  uiMesswerteAnzahl, adMesswerte          );

   cout << “Mittelwert: “ <<  dMittelwert << endl          ;

   cout << “Varianz: “ <<   dVarianz << endl          ;


   delete    []  adMesswerte                          ;   // Freigabe dynamisches Array


   return 0;
}
```

Kommentar [KM12]: 1 P
Kommentar [KM13]: 1 P
Kommentar [KM14]: 1 P
Kommentar [KM15]: 1 P
Kommentar [KM16]: 1 P
Kommentar [KM17]: 1 P
Kommentar [KM18]: 0,5 0,5
Kommentar [KM19]: 1 P
Kommentar [KM20]: 1 P
Kommentar [KM21]: 0,5
Kommentar [KM22]: 0,5
Kommentar [KM23]: 1 P

6. Aufgabe: Algorithmus

Was berechnet die nachfolgende Funktion bzw. was macht sie? Bitte beschreiben Sie die Funktionsweise möglichst abstrakt. (15 P)
Hinweis: Testen Sie den Algorithmus anhand eines kleinen Arrays und beobachten Sie die Variablenwerte.

```
void unknown(int* aiInOutArray, unsigned int uiInAnzahl)
{
   for(unsigned int uiIndex=0; uiIndex < uiInAnzahl; ++uiIndex)
   {
      bool bNoSwap = true;
      for(unsigned int uiIndex2=0; uiIndex2  < uiInAnzahl – uiIndex - 1; ++uiIndex2)
      {
         if(aiInOutArray[uiIndex2] > aiInOutArray[uiIndex2+1]
         {
            int iTmp                    = aiInOutArray[uiIndex2];
            aiInOutArray[uiIndex2]      = aiInOutArray[uiIndex2+1];
            aiInOutArray[uiIndex2+1] = iTmp;
            bNoSwap = false;
         }
      }
      if(bNoSwap)
      {
         return;
      }
   }
}
```

Die Zahlen im Array werden aufsteigend sortiert.