Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Studiengang ESTB / EEEB

Note / normierte Punkte

Klausur in Programmieren

Sommer 2015, 23. Juli 2015

Dauer: 1,5h

Hilfsmittel: Keine (Wörterbücher sind auf Nachfrage erlaubt)

Name:
MatrikelNr.:

Aufgabe	1	2	3	4	5	6	Summe
Punkte max	12	20	17	12	24	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

Welche Werte haben die angegebenen Variablen? (12 P)

a) int x = 25 / 4; x = __6__ (1 P)

b) double y = 19.0 / 2; y = __9.5__ (1 P)

c) int diff = 'D' – 'B' ; diff = __2__ (2 P)

d) int x = 5;
float f = ++x / 3.0 ; f = __2.0__, x = __6__ (2 P)

e) char c = 0xB0;
double d = (c++ % 176) * 3.141 ; d = __0.0__, c = __177__ (6 P)

2. Aufgabe: Grundlagen

Schreiben Sie ein funktionsfähiges Hauptprogramm, das eine Schleife enthält und so lange reelle Werte in die Variable x von der Konsole einliest, bis der Wert 0 eingegeben wird. Bestimmen Sie innerhalb der Schleife den jeweiligen Mittelwert aller bis zu diesem Zeitpunkt eingegebenen Werte und speichern Sie diesen Wert in der Variablen y mit einem dafür geeigneten Datentyp. Der Wert 0 für x darf nicht in die Mittelwertbildung einfließen. Danach geben Sie diesen reellen Wert von y auf die Konsole aus. Binden Sie alle dazu notwendigen Bibliotheken ein, aber nicht mehr. (20P)

```c
#include <stdio.h>

int main()
{
  float fValue = 0.0;
  float fSum = 0.0;
  int iCount = 0;

  do
  {
    float y;

    printf("Wert: ");
    scanf("%f", &fValue);

    if(fValue != 0.0)
    {
      fSum += fValue;

      y = fSum / ++iCount;
      printf("Mittelwert %f\n", y);
    }
  }
  while(fValue != 0.0);

  return 0;
}
```

3. Aufgabe: Variablen und Schleifen

a) Was versteht man unter einer Variablendeklaration im Unterschied zu einer Variablendefinition? (**wenige** und **kurze** Sätze, bei Romanen kann es Punktabzug geben!) (4 P)

Eine Variablendefinition ist eine Variablendeklaration, bei der die Variable einen Anfangswert erhält. Man spricht hier von einer Initialisierung der Variablen.
Eine Variablendeklaration legt nur den Namen und den Datentyp der Variablen fest. Der Wert ist undefiniert.

b) Was wird berechnet? (4P)

```
#define SQR_simple(x)    x*x
#define SQR(x)    ((x) * (x))
```

int x = SQR_simple(5+1); x == _____ 5+1*5+1 = 11 _____

int y = SQR(3+2); y == ___ ((3+2) * (3+2)) = 25 ____

c) Welche Schleifenarten gibt es in C? Geben Sie kleine Beispiele als Programm**fragmente** (keine Funktionen, kein Hauptprogramm, keine Ein- oder Ausgabe!). (9 P)

```
while:

double dValue = 1.0;
while(dValue < 100.0)
{
  dValue *= 10.0;
  if(dValue > 40.0)
  {
    dValue /= 2.0;
  }
}
```

```
do .. while:

int iValue = 20;
do
{
  iValue *= 4;
}
while(iValue < 50);
```

```
for:

for(int li=0; li<10;
++li)
{
  cout << li << endl;
}
```

- Lösen Sie die Aufgaben bitte auf dem Blatt -

4. Aufgabe: Array/Feld, Indizierung

a) Erklären Sie den Unterschied zwischen einem Array und einer Struktur (kein Roman – kann Punktabzug geben!). (4 P)

- ein Array besteht aus einer Anzahl von Variablen/Feldern desselben Datentyps. Der Zugriff auf einzelne Felder erfolgt mit dem Index-Operator [ ].
- eine Struktur besteht aus mehreren Variablen, die einen unterschiedlichen Datentyp besitzen können. Der Zugriff erfolgt mit dem Punkt-Operator .

b) Schreiben Sie eine kleine Funktion, bei der ein Array mit reellen Messwerten übergeben wird und das Maximum bestimmt wird. Das Maximum soll als reelle Zahl in Form eines Funktionswertes zurück gegeben werden. Geben Sie anschließend ein Programmfragment an, bei dem die von Ihnen definierte Funktion aufgerufen wird (d.h. weniger als 5 Anweisungen reichen und bitte kein Hauptprogramm oder irgendwelche Ein- oder Ausgaben schreiben!). Bibliotheken dürfen nicht verwendet werden! (8 P)

```
float maximum(float* afInValues, int iInCount)
{
  float fMaximum = afInValues[0];
  int li;

  for(li=1; li < iInCount; ++li)
  {
    if(fMaximum < afInValues[li])
    {
      fMaximum = afInValues[li];
    }
  }
  return fMaximum;
}

  float afValues[3];
  float fMaximum;
  afValues[0] = 50.5;
  fMaximum = maximum(afValues, 3);
```

- Lösen Sie die Aufgaben bitte auf dem Blatt -

5. Aufgabe: Zeichenketten

a) Schreiben Sie eine Funktion strlength, die die Länge einer mit 0 terminierten Zeichenkette bestimmt. Die Zeichenkette soll als Parameter übergeben werden. Der Funktionswert soll die Länge zurückgeben (kein Hauptprogramm, keine Ein- oder Ausgabe!). (8 P)

```
int strlength(char* acInString)
{
  int iIndex = 0;
  while(acInString[iIndex] != 0)
  {
    iIndex++;
  }
  return iIndex;
}
```

b) Schreiben Sie eine Funktion upper, die eine Zeichenkette acString mit Klein-/ und oder Großbuchstaben in Großbuchstaben ändert. Alle anderen Zeichen sollen unverändert bleiben. (kein Hauptprogramm, keine Ein- oder Ausgabe!). (16 P)

upper("This IS An ExamPle!") → "THIS IS AN EXAMPLE!"
Hinweis: Beim ASCII-Code sind 'A' … 'Z' und 'a' … 'z' lückenlose, zusammenhängende Bereiche; nutzen Sie deshalb: char cOffset = 'a' – 'A';

```
void upper(char* acInOutString)
{
  int iIndex = 0;
  char cOffset = 'a' - 'A';
  while(acInOutString[iIndex] != 0)
  {
    char cCurrent = acInOutString[iIndex];
    if('a' <= cCurrent  &&  cCurrent <= 'z')
    {
      acInOutString[iIndex] -= cOffset;
    }
    iIndex++;
  }
}
```

- Lösen Sie die Aufgaben bitte auf dem Blatt -

6. Aufgabe: Algorithmus

Was macht die nachfolgende Funktion unknown? Wann ist das Ergebnis undefiniert?

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

Hinweis: Testen Sie den Algorithmus anhand eines kleinen Arrays (z.B. der Größe 5) und beobachten Sie die Variablenwerte.

```
struct tResult
{
  int iValue1;
  int iValue2;
};

tResult unknown(int* aiInArray, unsigned int uiInAnzahl)
{
    tResult  sResult;
    sResult.iValue1 = aiInArray[0];
    sResult.iValue2 = aiInArray[0];
    for(unsigned int uiIndex=0; uiIndex < uiInAnzahl; ++uiIndex)
    {
        if(sResult.iValue1 < aiInArray[uiIndex])
        {
            sResult.iValue1 = aiInArray[uiIndex];
        }
        sResult.iValue2 = sResult.iValue2 > aiInArray[uiIndex] ? aiInArray[uiIndex] : sResult.iValue2;
    }
    return sResult;
}
```

Frage 1)
iValue1 enthält das Maximum aller Elemente des Arrays.
iValue2 enthält das Minimum aller Elemente des Arrays.

Frage 2)
Wenn uiInAnzahl gleich 0 oder größer als die Elementanzahl des Arrays ist.

- Lösen Sie die Aufgaben bitte auf dem Blatt -