

Aufgabe 1 (40 Punkte)

Gesamt: /85

Jede richtig beantwortete Frage ergibt **2 Punkte**. Jede falsch-, mehrfach- oder nicht beantwortete Frage ergibt **0 Punkte**. Je Frage ist nur eine Antwort richtig.

Viel Erfolg!

	Lösung
<p>1.1 Welche Funktion wird zur Freigabe des durch <code>malloc</code> allozierten Speichers verwendet?</p> <p>A. <code>realloc</code>; B. <code>calloc()</code>; C. <code>free()</code>; D. <code>memfree()</code>;</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.2 Gegeben seien nachfolgende Definitionen:</p> <pre># define fun(x,y) y x int e = 7;</pre> <p>Welchen Wert liefert <code>fun((5+e),5)</code>?</p> <p>A. 10 B. 11 C. 12 D. 13</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.3 Die <code>int</code>-Zeiger <code>p1</code> und <code>p2</code> zeigen auf die <code>int</code>-Variablen <code>a1</code> und <code>a2</code>. Dann wird aufgrund folgender Anweisung:</p> <pre>printf("%d\n", *(p1+p2));</pre> <p>A. zur Laufzeit ein undefinierter Wert angezeigt. B. der Wert der Summe <code>a1+a2</code> angezeigt. C. der Wert von <code>a1</code> angezeigt. D. der Compiler eine Fehlermeldung ausgeben.</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.4 Setzen Sie die folgende Anweisung in C-Quelltext um:</p> <p>"Definieren Sie einen Zeiger auf ein Feld mit vier Zeichen (<code>char</code>)".</p> <p>A. <code>char (*ptr)[4];</code> B. <code>char *ptr[3]();</code> C. <code>char *ptr[4];</code> D. <code>char (*ptr[4])();</code></p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.5 Welchen Wert hat die Variable <code>x</code>?</p> <pre>x = (64 >> 3) & (1<<3);</pre> <p>A. 2 B. 4 C. 8 D. 16</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>

<p>1.6 Welche der nachfolgenden Definitionen ist falsch?</p> <p>A. long int a; B. double int b; C. short int c; D. unsigned int;</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.7 Welche dezimale Zahl wird in nachfolgendem Programmausschnitt durch printf() ausgegeben?</p> <pre>int a = 021; int b = 0x1A; printf("Summe = %d", a+b);</pre> <p>A. 2A B. 45 C. 42 D. 43</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.8 Der Linker...</p> <p>A. bindet die mit #include angegebenen Bibliotheken in das Programm ein. B. bindet Makefiles verschiedener Programmebenen zu einer für make interpretierbaren Datei. C. bindet Objekt-Dateien und ggf. Bibliotheken zu einer ausführbaren Datei zusammen D. übersetzt den Quellcode, wodurch eine Objektdatei (*.obj oder *.o) erzeugt wird.</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.9 Das Ergebnis des folgenden Ausdrucks lautet:</p> <pre>22 & (210 >> 3);</pre> <p>A. 16 B. 0x12 C. 018 D. 0x18</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.10 Welchen Wert haben x und y nach der Ausführung folgender arithmetischen Ausdrücke?</p> <pre>int x = 3 %4 +2; int y = 9 + x++ *-2;</pre> <p>A. x = 6, y = -1 B. x = 5, y = -1 C. x = 1, y = 1 D. x = 2, y = 1</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>

<p>1.11 Der Zeiger <code>ptm</code> zeigt auf eine Struktur vom Typ <code>struct Time</code>. Dann repräsentiert der folgende Ausdruck das Element <code>sec</code> des Objekts:</p> <p>A. <code>ptm.sec</code> B. <code>ptm->*sec</code> C. <code>ptm->sec</code> D. <code>*ptm.sec</code></p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.12 Wie lautet die Ausgabe von <code>printf</code> im folgenden Code?</p> <pre>union H_BRS { int St_Augustin, Rheinbach, Hennef; }; union H_BRS Stud_2014; Stud_2014.St_Augustin = 4431; Stud_2014.Rheinbach = 1952; Stud_2014.Hennef = 319; printf("%d\n", Stud_2014.St_Augustin);</pre> <p>A. 4431 B. 1952 C. 319 D. 6702</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.13 Die Ausgabe der folgende Schleife lautet:</p> <pre>int i = 5; while(i-- >= 0) printf("%d,", i);</pre> <p>A. 4, 3, 2, 1, 0, -1 B. 5, 4, 3, 2, 1, 0 C. 4, 3, 2, 1, 0 D. 5, 4, 3, 2, 1</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.14 Welcher der folgenden Datentypen kann nicht in einer <code>switch-case</code> Anweisung verwendet werden?</p> <p>A. <code>int</code> B. <code>float</code> C. <code>enum</code> D. <code>char</code></p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>

<p>1.15 Wie oft wird „Viel Erfolg!“ ausgegeben?</p> <pre> int x; for(x=0; x<=5; x++) { if(x < 2) printf("Viel Erfolg!"); else printf("Viel Erfolg!"); break; } </pre> <p>A. 2 mal B. 3 mal C. 4 mal D. 1 mal</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.16 Die Variablen z und p seien wie folgt definiert:</p> <pre> float z = 3.14, *p = &z; printf("%f, %p, %f, %p", z, p, *p, &p); </pre> <p>z und p haben die Adressen 0x28FF08 und 0x28FF0C. Wie lautet die Ausgabe von printf?</p> <p>A. 3.14, 0x28FF08, 3.14, 0x28FF0C B. 3.14, 0x28FF08, 3.14, 0x28FF08 C. 3.14, 0x28FF0C, 3.14, 0x28FF0C D. 3.14, 0x28FF08, 0x28FF08, 0x28FF08</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.17 In welcher Stufe des C-Programmiers wird die folgende Quelltextzeile #include <stdio.h> durch den Inhalt von stdio.h ersetzt?</p> <p>A. beim Editieren B. beim Linken C. bei der Ausführung D. beim Präprozessing</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.18 Was wird unter dem Begriff „call by reference“ verstanden?</p> <p>A. Eine Funktion wird durch ihre Adresse referenziert und aufgerufen. B. Der Aufruf einer Funktion (call) liefert eine Referenz auf den Rückgabewert. C. Der aufgerufenen Funktion wird die Adresse einer Variablen als Parameter übergeben. D. Der aufgerufenen Funktion wird die Kopie einer Variablen als Parameter übergeben.</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>

<p>1.19 Das Programm prog.c wird wie folgt, auf der Konsole ausgeführt:</p> <pre>C:\ prog 5 - 10 //prog.c int main(int argc, char *argv[]){ printf("%.2f \n", (float) (atoi(argv[1])+atoi(argv[3]))/2); return 0; }</pre> <p>Folgende Zahl wird auf der Konsole ausgegeben:</p> <p>A. 15.00 B. 5.00 C. 7.50 D. 10.00</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>
<p>1.20 Wie groß ist die Wortbreite der Variable day im folgenden Code?</p> <pre>enum woche {mo=1, di, mi, do, fr, sa, so} day;</pre> <p>A. 28 Bytes B. 4 Bytes C. 7 Bytes D. 14 Bytes</p>	<p><input type="radio"/> A <input type="radio"/> B <input type="radio"/> C <input type="radio"/> D</p>

Aufgabe 2 (15 Punkte):

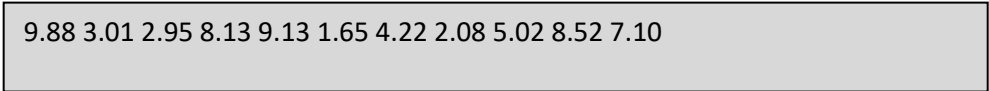
Auf den nächsten Seiten finden Sie den Quelltext des Programms `klausur`, das wie folgt erzeugt werden soll:

```
gcc -Wall klausur.c -o klausur
```

Den Inhalt der Datei `vf.txt` zeigt **Abbildung 1**.

Aufgabe:

Analysieren Sie die Semantik des Quellcodes. Geben Sie die (Bildschirm-) Ausgaben von `printf` der **Quelltextzeile 24** in der Funktion `main()` bei Ausführung des Programms `klausur` an.



9.88 3.01 2.95 8.13 9.13 1.65 4.22 2.08 5.02 8.52 7.10

Abbildung 1: Inhalt der Datei `vf.txt`

```
1 //Filename: klausur.c
2
3 #include <stdio.h>
4
5 #define RVFF 10
6 #define NOV RVFF/2
7
8
9 void MySoA(double * , int , int );
10 void rff(double*);
11 void (*pms)()=MySoA;
12
13
14 int main ()
15 {
16     int iCnt;
17     static double anArray[NOV];
18
19     rff(anArray);
20     MySoA(anArray, 0, NOV-1);
21
22     for (iCnt=0; iCnt < NOV; iCnt++)
23     {
24         printf("Value %d is %lf\n", iCnt, anArray[iCnt]);
25     }
26     return(0);
27 }
28
29 void MySoA(double p2Arr[], int iStartIdx, int iEndIdx)
30 {
31     int iUp= iStartIdx,iDown= iEndIdx;
32     double dMedVal,dTmp;
33
34     dMedVal = p2Arr[(iStartIdx+iEndIdx) / 2];
35
36     do
37     {
38         while(p2Arr[iUp] < dMedVal)
39             iUp++;
40         while(p2Arr[iDown] > dMedVal)
41             iDown--;
42
43         if( iUp <= iDown)
44         {
45             dTmp = p2Arr[iUp];
46             p2Arr[iUp] = p2Arr[iDown];
47             p2Arr[iDown] = dTmp;
48             iUp++;
49             iDown--;
```

```
50         }
51     }
52     while(iUp < iDown);
53
54     if(iStartIdx < iDown)
55     {
56         pms(p2Arr,iStartIdx,iDown);
57     }
58     if(iUp < iEndIdx)
59     {
60         pms(p2Arr,iUp,iEndIdx);
61     }
62 }
63
64 void rff(double ats[])
65 {
66     double atr[RVFF];
67     FILE * p2f;
68     int i =0;
69
70     if ((p2f=fopen("vf.txt","r")) == NULL)
71     {
72         printf("Fatal error\n");
73         return;
74     }
75     while(i < RVFF)
76     {
77         fscanf(p2f, "%lf", &atr[i]);
78         printf("Function %s : Value %d = %lf\n",__FUNCTION__, i,
79 atr[i]);
80         i++;
81     }
82     fclose(p2f);
83
84     for (i=0; i< NOV; i++)
85     {
86         if ((i%2) == 0)
87         {
88             ats[i]= atr[i];
89         }
90         else
91         {
92             ats[i]= (double)i;
93         }
94     }
95     printf("Function %s : ats[%d] = %lf\n",__FUNCTION__, i, ats[i]);
96 }
97
```




Lösungsblatt zu Aufgabe 2:

Aufgabe 3: (12 Punkte)

Der nachfolgende Quelltext (170 Zeilen) enthält zwölf syntaktische Fehler, die von Ihnen korrigiert werden sollen. Die Korrektur soll durch Angaben nach folgendem Schema auf dem beiliegenden Lösungsblatt erfolgen:

< Zeilennummer>: vollständige, korrigierte Quelltextzeile

```
1 // PiC Klausur SS2014
2 //
3 / Aufgabe 3
4 //
5 //
6
7
8 #include <stdio.h>
9 #include <math.h>
10
11 #define ModulName "C_Exam_SS2014.c"
12 #define SizeOfVArray 97
13 #define WidthOfRandNum 32
14 #define ModFRandNum pow(2,WidthOfRandNum)
15
16 #define NumOfRandNum 10
17
18 typedef enum = { FALSE, TRUE } BOOL;
19 typedef char CHAR8;
20 typedef unsigned char BYTE;
21 typedef short INT16;
22 typedef int INT32;
23 typedef unsigned short UINT16;
24 typedef unsigned int UINT32;
25 typedef float FLOAT32;
26 typedef float DOUBLE32;
27 typedef double DOUBLE64;
28
29 #define TRUE 1
30 #define FALSE 0
31
32 /* local global variables */
33 static BOOL bInitFlag = FALSE;
34 static UINT32 uiVArray[SizeOfVArray];
35
36 BYTE InitRandGen(void);
37 BYTE __InitFiboGen(void);
38 DOUBLE64 GetRandNum(void);
39 DOUBLE64 GetNegExp(void);
40 INT32 GetPoisson(DOUBLE64 );
41
42 int main(int argc, char *argv)
```

```
43 {
44     UINT32 uiCnt;
45     DOUBLE64 dRndNum;
46     FILE* pfRandFile;
47
48     if(InitRandGen() != 0x00)
49     {
50         printf("\nError: can't initialize random generator!\n
51             \nFile: %s\nFunction: InitRandGen", ModulName);
52         return(0x01);
53     }
54     /* open result file*/
55     if((pfRandFile = fopen("FiboTest.xls","w")) == NULL)
56     {
57         printf("Fatal error: Can't open result file!!\n\n");
58         return (0x01);
59     }
60
61     for (uiCnt = 0; uiCnt < NumOfRandNum; uiCnt++)
62     {
63         dRndNum = GetRandNum();
64         fprintf(pfRandFile,"%f \n",dRndNum);
65     }
66     fclose(pfRandFile)
67
68     return 0;
69 }
70
71 BYTE InitRandGen(void)
72 {
73     /* local prototypes */
74     BYTE __InitFiboGen(void);
75
76     if (bInitFlag == FALSE)
77     {
78         if(__InitFiboGen() != 0x00)
79         {
80             printf("\nError: can't initialize random generator!\n
81                 \nFile: %s\nFunction: InitRandGen",
82 ModulName);
83             return(0x01);
84         }
85         bInitFlag = TRUE;
86         return 0x00;
87     }
88     else
89     {
90         printf("\nError: re-initialization of module RandGen!\n
91             \nFile: %s\nFunction: InitRandGen", ModulName);
92         return 0x01;
```

```
93     }
94 }
95 BYTE __InitFiboGen(void)
96 {
97     UINT16 uiBitCnt,uiArrayCnt;
98     UINT32 uiRandNum;
99     UINT32 uiZn=78,uiYn,uiYnm3=12,uiYnm2=34,uiYnm1=56;
100    FILE * pfTestFile;
101
102    if((pfTestFile = fopen("FiboIniTest.xls","w")) == NULL)
103    {
104        printf("Fatal error: Can't open result file!!\n\n");
105        return (0x01);
106    }
107
108    for (uiArrayCnt = 0, uiArrayCnt < SizeOfVArray; uiArrayCnt++)
109    {
110        uiRandNum = 0;
111        for (uiBitCnt = 0; uiBitCnt < WidthOfRandNum; uiBitCnt++)
112        {
113            uiYn = (uiYnm3 * uiYnm2 * uiYnm1) % 179;
114            uiYnm3 = uiYnm2;
115            uiYnm2 = uiYnm1;
116            uiYnm1 = uiYn;
117            uiZn = (53 * uiZn + 1) % 169;
118            if(((uiZn*uiYn)%64) >= 32)
119                uiRandNum += (UINT32) pow((long) 2, (long) uiBitCnt);
120        }
121        uiVArray[*uiArrayCnt] = uiRandNum;
122        fprintf(pfTestFile,"InitValue %u : %u\n", uiArrayCnt,
123 uiVArray[uiArrayCnt]);
124    }
125    fclose (pfTestFile);
126    return(0x00);
127 }
128
129 DOUBLE64 GetRandNum(void)
130 {
131     static UINT16 uiIndex=0;
132     static UINT32 uiCn = 15362436,uiXn;
133     DOUBLE64 dRes;
134
135     uiCn=uiCn-362436069;
136     uiVArray[uiIndex] -= uiVArray[(uiIndex + 64)% 97];
137     uiXn = uiVArray[uiIndex] [1] -uiCn;
138     dRes = ((DOUBLE64) uiXn) / ModFRandNum;
139     uiIndex = (uiIndex +1) %97;
140     return(dRes);
141 }
142
```

```
143 DOUBLE64 GetNegExp(DOUBLE64 dLambda)
144 {
145     DOUBLE64 dRetVal;
146     dRetVal = -(1.0/dLambda) * log(1.0 - GetRandNum(100));
147     return (dRetVal)++;
148 }
149 }
150
151 INT32 GetPoisson(DOUBLE64 dBeta)
152 {
153     DOUBLE64 dTmp, dHelpA, dHelpB;
154     UINT32 uiX;
155     dTmp = GetRandNum();
156     /* start values */
157     dHelpA=dHelpB=(DOUBLE64) exp(-dBeta);
158     uiX = 0;
159     while{dTmp > dHelpA}
160 {
161     uiX++;
162     dHelpB = dHelpB * (DOUBLE64)(dBeta / uiX);
163     dHelpA +=dHelpB;
164 }
165
166     return (uiX);
167
168
169 }
170 // The End
```



Lösungsblatt zu Aufgabe 3:

- 1:
- 2:
- 3:
- 4:
- 5:
- 6:
- 7:
- 8:
- 9:
- 10:
- 11:
- 12:

Aufgabe 4: (18 Punkte)

Im nachfolgenden C-Programm soll das Skalar-, sowie das Kreuzprodukt von zwei dreidimensionalen Vektoren A und B berechnet werden.

Das **Skalarprodukt** von zwei Vektoren $A = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}$ und $B = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$ ist eine Zahl und wird nach der folgenden Formel berechnet:

$$s = \vec{A} \cdot \vec{B} = a_1 \cdot b_1 + a_2 \cdot b_2 + a_3 \cdot b_3$$

Das **Kreuzprodukt** von zwei dreidimensionalen Vektoren ist ein dreidimensionaler Vektor und wird nach der folgenden Formel berechnet:

$$\vec{K} = \vec{A} \times \vec{B} = \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} a_2 \cdot b_3 - a_3 \cdot b_2 \\ a_3 \cdot b_1 - a_1 \cdot b_3 \\ a_1 \cdot b_2 - a_2 \cdot b_1 \end{pmatrix}$$

- Vervollständigen Sie die Funktionsköpfe für die Funktionen „skalar_produkt“ und „kreuz_produkt“. (Zeile 41&50)
- Ergänzen Sie im Programm „skalar_kreuz_produkt.c“ die Prototypen für die Funktionen skalar_produkt und kreuz_produkt. (Zeile 4&5)
- Berechnen Sie das Skalarprodukt der Vektoren A und B. (Zeile 45)
- Berechnen Sie mit einer for-Schleife das Kreuzprodukt der Vektoren A und B. Die for-Schleife soll nur eine Zeile beinhalten. (Zeile 54&56)
- Warum wurde in der Funktion „kreuz_produkt“ das Schlüsselwort static für den Vektor k verwendet?

.....


```

1 // skalar_kreuz_produkt.c
2 # include <stdio.h>
3
4 .....
5 .....
6
7 int main(){
8     char p;
9     int i, run = 1;
10    float *k;
11    float a[3], b[3];
12
13    while(run){
14        printf("Eingabe starten(j/n)?\n");
15        fflush(stdin);
16        if (scanf("%c", &p)== 1 && p=='j' || p=='J'){
17            printf("Geben Sie Matrix A ein:");    //Eingabe Matrix A
    
```

```
18         scanf("%f %f %f", a, (a+1), (a+2));
19         printf("Geben Sie Matrix B ein:");    //Eingabe Matrix B
20         scanf("%f %f %f", b, (b+1), (b+2));
21
22         printf("\nSkalarprodukt A.B: %f\n\n",
23             skalar_produkt(a, b));    //Ausgabe Skalarprodukt
23
24
25         k = kreuz_produkt(a, b);
26         printf("Kreuzprodukt AxB:\n
27             _____\n");
28         for (i=0; i<3; i++){           //Ausgabe Kreuzprodukt
29             printf("K[%d] = %f\n",i, *(k+i));
30         }
31         printf("\n");
32         run =1;
33     }
34     else
35         run = 0;
36         continue;
37     }
38     return 0;
39 }
40
41 ..... skalar_produkt (.....)
42 {
43     float c=0;
44
45     .....
46
47     return (c);
48 }
49
50 ..... *kreuz_produkt (.....)
51 {
52     int i;           //Zählvariable
53     static float c[3]={0, 0, 0};
54     for (.....;.....;.....){
55
56         .....
57     }
58
59     return (c);
60 }
```