Aufgabe 1: Multiple-Choice Fragen (30 Punkte)

Jede richtig beantwortete Frage 1.1 bis 1.15 ergibt **2 Punkte**. Jede falsch-, mehrfach- oder nicht beantwortete Frage ergibt **0 Punkte**. Je Frage ist nur eine Antwort richtig. Bitte schreiben Sie auf Ihrem Lösungsblatt die Nummer der Frage sowie Ihre Lösung. Beispiel: 1 A, 2 B,…

1.1 Welche dezimale Zahl wird in nachfolgendem Programmausschnitt ausgegeben?

```
int x = 050, y = 0x25;
printf("x – y = %d", x-y);
```

A. 25
B. 43
C. 3
D. 19

1.2 Die int-Zeiger `p1` und `p2` zeigen auf die int-Variablen `a` und `b`. Dann wird durch folgende Anweisung:

```
printf("%d\n", *(p1 + p2));
```

A. zur Laufzeit ein undefinierter Wert angezeigt.
B. der Wert der Summe *p1+*p2, also a+b angezeigt.
C. der Inhalt der Adresse p1+p2 angezeigt.
D. der Compiler eine Fehlermeldung ausgeben.

1.3 Der Wert von x in hexadezimaler Notation beträgt:

```
x = (192 >> 2) & (~(2<<2));
```

A. 0
B. 30
C. 48
D. F7

1.4 Der Zeiger `ptr` zeigt auf eine Struktur vom Typ `struct Date`. Dann repräsentiert der folgende Ausdruck das Element `day` des Objekts:

A. ptr.day
B. *ptr.day
C. ptr->*day
D. ptr->day

1.5 Was ist die Ausgabe des folgenden Codes?

```
int fun(int a){  return a*a;}
int main(){ int k=2; printf("%d\n", fun(fun(k))); return 0;}
```

A. 16
B. 8
C. 4
D. 256

1.6 Welche Header-Datei werden Sie dem folgenden Code hinzufügen, sodass er korrekt funktioniert?

```
#include<stdio.h>
int main() {printf("%f\n", sqrt(36)); return 0;}
```

A. stdlib.h
B. time.h
C. math.h
D. dos.h

1.7 Speicher von der Größe einer Struktur wird reserviert, wenn

A. die Header-Datei mit der Strukturdefinition eingebunden wird.
B. eine Variable vom Typ der Struktur angelegt wird.
C. der Strukturtyp definiert wird.
D. eine Variable vom Typ der Struktur initialisiert wird.

1.8 Nach der Ausführung des folgenden Codes beträgt der Inhalt von `v.b`:

```
union var {int a, b, c;} v;
v.b = 10; v.c = 30;
```

A. 30
B. 10
C. 0
D. 300

1.9 Wie lautet die Ausgabe des folgenden Codes?

```
#define FUN(i, j) i##j
int main(){
      int v1=1, v10=2, v105= 3;
      printf("%d, %d\n", FUN(v1, 0), FUN(v1, 05));
      return 0; }
```

A. 1, 2
B. 2, 2
C. 2, 3
D. Error!

1.10 Welche der nachfolgenden Definitionen ist falsch?

A. `long int a`
B. `unsigned int a`
C. `short int a`
D. `double int a`

1.11 Die Ausgabe des nachfolgenden Codes lautet:

```
int a[10], *p = &a[3];
printf("%d", p - a);
```

A. 10
B. 3
C. 7
D. 0

1.12 Das Programm `prog.c` wird wie folgt auf der Konsole ausgeführt:

```
C:\ prog 13 12 11
  //prog.c
  int main(int argc, char *argv[]){
        printf("%.1f \n", sqrt(atoi(argv[1])+atoi(argv[2])));
        return 0;}
```

Welche Zahl wird auf der Konsole ausgegeben:

A. 5.000000
B. 5.0
C. 4.75
D. 4.7

1.13 Wieviel Speicherplatz belegt die Variable eye im folgenden Code?

```
enum color {blue, brown, green, hazel, amber, silver, black} eye;
```

A. 4 Bytes
B. 7 Bytes
C. 14 Bytes
D. 28 Bytes

1.14 Definieren Sie einen Zeiger auf eine Funktion mit einem int-Zeiger als Übergabeparameter und Rückgabe eines float-Zeigers.

A. `float (*ptr)( float)`
B. `float *(*ptr)(int*)`
C. `int *(ptr)*float`
D. `int *(*ptr)(float*)`

1.15 Matrix `a` ist von der Ordnung `4x3`. Welcher der folgenden Ausdrücke ist äquivalent zu `a[i][j]`?

A. `*(&a[0][0] + 3*i + j)`
B. `(*(a+i))+j`
C. `*((*(a+j)) + i)`
D. `*(a[i][j])`

Aufgabe 2 (8 Punkte)

Das folgende Programm soll die Fibonacci-Zahlen bis zum Wert von FiboSet berechnen und ausgeben. Das Programm enthält 10 syntaktische Fehler.

2.1 Finden Sie die Fehler und korrigieren Sie den Code so, dass er kompiliert und die Fibonacci-Zahlen bis zum FiboSet ausgibt.

2.2 Ändern Sie den korrigierten Code so, dass die Obergrenze der Zahlen während der Laufzeit gefragt und eingelesen wird.

```
1  #include 'stdio.h'
2  #define FiboSet 10;
3
4  void fiboGen(int, int)
5
6  int fiboNum [FiboSet];
7  int main
8  {
9      int i;
10     fiboGen(FiboSet, *fiboNum);
11
12     for (i=0;i<FiboSet; i++)
13       printf("%d: %d\n",i+1, (fiboNum+i));
14 }
15
16 int fiboGen(int set, int* num){
17     num = 1;
18     *(num+1) = 1;
19     for(int i==2, i<set, i++){
20       *(num+i) = *(num+i-2) + *(num+i-1);      }
21 }
```

Aufgabe 3 (12 Punkte):

In dieser Aufgabe soll ein C-Programm implementiert werden, das die vier Grundrechenarten Addition, Subtraktion, Multiplikation und Division mit komplexen Zahlen berechnet und ausgibt.

$x = a + i.b$
$y = c + i.d$

Hinweis:

$x + y = (a + c) + i.(b + d)$
$x - y = (a - c) + i.(b - d)$
$x.y = a.c - b.d + i.(a.d + b.c)$
$\frac{x}{y} = \frac{a.c + b.d}{c^2 + d^2} + i.\frac{(b.c - a.d)}{c^2 + d^2}$

4.1 Definieren Sie ein `struct complex`, der die zwei Komponenten einer komplexen Zahl `float real` und `float imag` beinhaltet.

4.2 Schreiben Sie eine Funktion `calcomp`, die

- zwei Zeiger auf den komplexen Zahlen `x` und `y` erhält,
- die vier Grundrechenarten mit den Zahlen `x` und `y` berechnet und speichert
- und einen Zeiger auf die vier Ergebnisse (z.B. als Zeiger auf ein Array) zurückgibt

4.3 Vervollständigen Sie Ihr Programm so, dass

- die komplexen Zahlen `x` und `y (a, b, c und d)` mit `scanf` von der Kommandozeile eingelesen werden,
- die Funktion `calcomp` aufgerufen wird und
- die Ergebnisse ausgegeben werden.

Aufgabe 4 (10 Punkte)

Schreiben Sie eine C-Funktion, die die mathematische Ackermann-Funktion abbildet. Die Ackermann-Funktion wird wie folgt definiert:

$$a(m,n) := \begin{cases} n+1 & wenn \quad m = 0 \\ a(m-1,1) & wenn \quad n = 0 \\ a(m-1,a(m,n-1)) & sonst \end{cases}$$

4.1. Schreiben Sie eine C-Funktion, die für zwei Integer-Werte m und n den Ackermann-Wert berechnet und den Wert zurückgibt.

4.2. Unter Verwendung der Funktion in 4.1 schreiben Sie ein komplettes C-Programm, das zwei Integer-Werte m und n von der Tastatur einliest, den Ackermann-Wert für $0 < m \leq 3$ und $0 < n \leq 8$ berechnet und auf der Konsole ausgibt. Wenn die Werte nicht im definierten Bereich sind, soll eine entsprechende Meldung ausgegeben werden.

Aufgabe 5 (15 Punkte)

5.1 Schreiben Sie eine C-Funktion `pascal`, die für die Zahl `n` das Pascalsche Dreieck implementiert und die letzte Zeile (im Beispiel `n=5`) zurückliefert.

5.2 Schreiben Sie eine Funktion `main`, die die Zahl `n` als Kommandozeilenparameter `(argv[1])` einliest, die Funktion `pascal` aufruft und das Ergebnis ausgibt.

Hinweis: Jeder Eintrag ist die Summe der zwei darüberstehenden Einträge.

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
1   5   10  10  5   1
```

Aufgabe 6 (10 Punkte)

Gegeben ist folgende C-Funktion:

```
double funny(int *a, int b){
    if (b <= 0)
        return a[0];
    else
        return (a[b] + b * funny(a, b-1)) / (b+1);
    }
```

Die Funktion wird wie folgt aufgerufen:

```
int a[] = {12, -8, 11};
int anzahl =  sizeof(a)/sizeof(int);
double ergebnis = funny(a, anzahl - 1);
```

6.1 Wie viele Rekursionsschritte gibt es?

6.2 Welche Werte nehmen `b` und `a[b]` in jedem einzelnen Rekursionsschritt an?

6.3 Welchen Wert nimmt die Variable `ergebnis` nach dem Funktionsaufruf an?

6.4 Welche mathematische Funktion wird mit der Funktion `funny` implementiert?