Fachhochschule Wedel
Prof. Dr. Uwe Schmidt

17. August 1999

Aufgaben zur Klausur **C** im SS 99 (II 21)

Zeit: 90 Minuten erlaubte Hilfsmittel: keine

Bitte tragen Sie Ihre Antworten und fertigen Lösungen ausschließlich an den gekennzeichneten Stellen in das Aufgabenblatt ein. Ist ihre Lösung wesentlich umfangreicher, so überprüfen Sie bitte nochmals Ihren Lösungsweg.

Sollten Unklarheiten oder Mehrdeutigkeiten bei der Aufgabenstellung auftreten, so notieren Sie bitte, wie Sie die Aufgabe interpretiert haben.

Viel Erfolg !

Diese Klausur besteht einschließlich dieses Deckblattes aus 11 Seiten

---

Aufgabe 1:

Gegeben seien die folgenden Variablen und Funktionen:

```
int x;
long int s;
float f;
char *p1;
int *p2;
void *p3;
int (*pf)(int);
int (*pf2)(double);
int f1(int x1);
int f2(int x1,int x2);
int f3(double x1);
```

Bestimmen Sie für die folgenden Ausdrücke den Typ gemäß ANSI-C. Vorsicht: Es kommen fehlerhafte Ausdrücke von. Kennzeichnen Sie diese entsprechend

p2[x] .....

p3[x] .....

p1 ? f1 : pf .....

~p2 .....

p1 && p1 .....

~s .....

s || x .....

(*pf)(5) .....

pf=f1 .....

pf=f1(x) .....

pf2=pf .....

pf==f1 .....

pf==pf2 .....

f=f3(f) .....

f1==f2 .....

*p3 .....

Aufgabe 2:

Gegeben sei das folgende C–Programm zur Verarbeitung von Mengen als Bitstrings.

```
#include <stdio.h>

typedef unsigned char Set;
#define SetMax 8

void printSet(Set s) {
    unsigned int i = SetMax;
    while ( i-- != 0 )
        printf("%1u", (unsigned int)((s >> i) & 1));
}

#define PRINT(s) { printSet(s); printf("\n"); }

#define single(i) ( (Set)(1 << (i)) )
#define first(n) (single(n) - 1)
#define interval(n,m) (first(m+1) ^ first(n))

int main(void) {
    Set s1;

    PRINT( (Set)0 );
    PRINT( (Set)-1 );
    PRINT( single(SetMax / 2 + 1) );

    PRINT( interval(0,1) );
    PRINT( interval(2,5) );
    PRINT( interval(0,SetMax) );

    PRINT( 3 | 8 );
    PRINT( 3 || 8 );
    PRINT( first(SetMax / 2 + 1) );

    s1 = interval(1,5) & ~interval(2,6); PRINT(s1);
    s1 = interval(1,5) ^ interval(2,6); PRINT(s1);

    s1 = 4 + 16 + 32;
    s1 ^= s1 & (~s1 + 1); PRINT(s1);

    return 0;
}
```

Die Mengen sind in diesem Beispiel 8 Bits lang, können also die Elemente $0, 1, \ldots, 7$ enthalten. *printSet* gibt eine Menge im Binärformat aus. Die Menge, die nur die 1 enthält würde als 00000010 ausgegeben werden. Das *PRINT* Makro gibt jeweils eine Menge pro Zeile aus.

Welche 12 Ausgabezeilen erzeugt dieses Programm?

1) .....

2) .....

3) .....

4) .....

5) .....

6) .....

7) .....

8) .....

9) .....

10) .....

11) .....

12) .....

Aufgabe 3:

Gegeben sei das folgende C–Programm

```
int x1 [] = {11,12,13};
int x2 [] = {40,20,60};
int x3 [] = {99,77,55};

int * a[] = {x3,x2,x1};
int ** pp, *p, c;

int main(int argc, char * argv[]) {

 /* 1. */ pp = a;
 /* 2. */ pp = a-1;

 /* 3. */ p = *p;
 /* 4. */ p = *(a + 1);
 /* 5. */ p = *a + 1;
 /* 6. */ p = *(a++);
 /* 7. */ pp = a, p = *(--pp);
 /* 8. */ pp = a-1, p = *(++pp);
 /* 9. */ pp = a, p = *(pp++) -1;

 /* 10. */ c = **a;
 /* 11. */ c = *a;
 /* 12. */ c = a;

 /* 13. */ c = a[2][3];
 /* 14. */ c = a[3][3];
 /* 15. */ c = a[3,3];

 return 0;
}
```

1. Ist *a* ein Feld von Zeigern ? ja ☐ nein ☐ weiß nicht ☐
2. Ist *a* ein Zeiger auf ein Feld ? ja ☐ nein ☐ weiß nicht ☐
3. Wieviel Platz benötigt *a* (ein Ausdruck mit sizeof Operator)?

. . . . . .

4. Wieviel Platz wird im Datensegment insgesamt für die Variablen $x1, x2, x3, a, pp, p$ und c benötigt (ein Ausdruck mit sizeof Operator)?

.....

.....

.....

Überprüfen sie die Anweisungen im Hauptprogramm auf korrekte Kompilierbarkeit, auf implizite Konversionen und auf nicht definierte Wirkung zur Laufzeit (Semantik undefiniert $\Rightarrow$ Ergebnisse Zufall).

1.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
2.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
3.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
4.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
5.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
6.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
7.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
8.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
9.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
10.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
11.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
12.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
13.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
14.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐
15.	Compilerfehler ☐	impl.Konversion ☐	Wirkung undefiniert ☐

Aufgabe 4:

Gegeben sind die folgenden Datentypen

```
typedef unsigned long int Element;
typedef int Bool;
typedef struct Node * Liste;
struct Node{
        Liste next;
        Element e;
        };
```

Deklarieren sie einen Datentyp *Predicate* für Funktionszeiger mit einem Wahrheitswert als Resultat und einem *Element* als Argument.

.....

Sind die Typdefinitionen für *Element* und *Bool* sinnvoll?

ja ☐ nein ☐

Begründung:

.....

.....

Schreiben Sie ein Prädikat (Boolesche Funktion), das testet ob ein *Element* gerade ist.

.....

.....

.....

.....

Schreiben Sie ein 2. Prädikat *ispower2*, das testet, ob ein *Element* eine 2-er Potenz ist. Verwenden Sie hierfür nur eine `return` -Anweisung, keine Zuweisungen, und nur Vergleiche, bedingte Ausdrücke, logische und Bitoperationen.

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Schreiben Sie eine allgemeine Suchroutine *find* mit einer Liste als Parameter und mit einem Suchkriterium (einem Prädikat) als Funktionsparameter. *find* soll als Resultat einen Booleschen Wert zurückgeben, der anzeigt ob ein Element mit der geforderten Eigenschaft in der Liste enthalten ist. Ist dies der Fall, so soll in einem zusätzlichen Parameter der erste gefundene Wert zurückgegeben werden. *find* soll iterativ arbeiten.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Gegeben sind die folgenden Variablen

Liste l_1;
Element e_2;
Bool found;

Suchen Sie in der Liste l_1 eine 2-er Potenz. Verwenden sie dabei die gegebenen Variablen.

.....