

Lösung Übung 5

infT

1.

Welcher Wert wird für `summe2` ausgegeben?

```
#include <stdio.h>

int main() {
    double summe1, summe2;
    double *p;

    summe1 = 187.5;
    p = &summe1;
    summe2 = *p;
    printf("Die Variable 'summe2' hat den Wert %lf", summe2);
    return 0;
}
```

L: Die Variable 'summe2' hat den Wert 187.500000

2.

Welchen Wert hat die Variable `x` am Ende des Programms?

```
#include <stdio.h>

int main() {
    int x;
    int *zeiger;

    x = 9;
    zeiger = &x;
    *zeiger = 10;
    printf("Die Variable 'x' hat den Wert %i", x);
    return 0;
}
```

L: Die Variable 'x' hat den Wert 10

3.

Welche Werte wird untenstehender Code liefern, wenn `x` eine Variable vom Typ `double` ist und `ptr` ein Pointer auf `double`?

```
int main() {
    double x = 2001.0;
    double *ptr;
    ptr = &x;

    printf("\nWert von ptr = %u", ptr);
    ptr++;
    printf("\nWert von ptr = %u", ptr);
    ptr++;
    printf("\nWert von ptr = %u", ptr);
    return 0;
}
```

L: Wert von ptr = 3221222844 (z.B.; Adresse von x, je nach Rechner etc.)
 Wert von ptr = 3221222852 (falls `sizeof(double): 8`)
 Wert von ptr = 3221222860

4.

Schreiben Sie eine Funktion `swapStrings`, die zwei Zeichenketten vertauscht.

L:

Die Zeichenketten selbst zu vertauschen ist ineffizient und müsste dynamisch geschehen. Es werden nur Pointer auf die Strings vertauscht. Der Name der Funktion hiesse dann auch besser `swapStringPtrs`.

```
void swapStringPtrs(char **ptr1, char **ptr2) {
    char *temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}
```

Daten und Funktionsaufruf:

```
...
char t1[] = "Text 1";
char t2[] = "Zweiter Text";
char *p1 = t1;
char *p2 = t2;

swapStringPtrs(&p1, &p2);
...
```

5.

Schreiben Sie eine Funktion, die in einem String ein Zeichen sucht und es jeweils durch ein anderes ersetzt. Benutzen Sie die folgende Deklaration zur Lösung der Aufgabe:

```
void replaceChar(char *s, char old, char new);
```

L:

```
void replaceChar(char *s, char old, char new) {
    while(*s) {
        if (*s == old) {
            *s = new;
        }
        ++s;
    }
}
```

6.

Welchen Fehler enthält das Programm?

```
#include <stdio.h>
```

```
int main() {
    int zahl, *ptr;
    zahl = 4711;
    *ptr = zahl;
    return 0;
}
```

L: `ptr` selbst wird nie initialisiert, enthält also eine zufällige Adresse.

7.

Unter der Annahme, dass `p1` und `p2` den Typ `char *` aufweisen, sind die beiden folgenden Statement ohne Inkrement- bzw. Dekrement-Operatoren neu zu formulieren:

```
a) *++p1 = *++p2;    L: p1 += 1;
                       p2 += 1;
                       *p1 = *p2;
b) *p1-- = *p2--;    L: *p1 = *p2;
                       p1 -= 1;
                       p2 -= 1;
```

8.

Schreiben Sie ein Programm `addSub(.exe)`, mit dem Sie zwei Zahlen addieren bzw. voneinander subtrahieren können. Die Übergabe der beiden Zahlen sowie der Anweisung zur Addition oder Subtraktion erfolge von der Kommandozeile aus. Bei Fehleingabe soll eine Ausgabe zum korrekten Gebrauch des Programms an die Konsole ausgegeben werden. Siehe auch die Bibliotheksfunktion `strtod` (oder `atof`).

L: siehe separate Datei

9.

Entwerfen Sie die Struktur `Complex` (für komplexe Zahlen in der Darstellung $z = a + b \cdot i$). Realisieren Sie die Funktionen für die Addition, Subtraktion, Multiplikation, Division und das Bilden der konjugiert komplexen Zahl. Nehmen Sie dazu eine günstige Aufteilung in Header- und Implementationsdateien vor. Tests wiederum sind in einer weiteren Datei zu implementieren.

L: separate Datei