Dieser Zettel hat 3 Seiten, 6 Aufgaben und einen Comic.

Aufgabe 1. Das folgende Programm sollte ein Array mit den Zahlen 1 bis 10 füllen und ausgeben. Leider haben wir dabei $6\frac{1}{2}$ Fehler gemacht. Schnapp sie dir alle!

```c
/* Array fuellen und ausgeben.
 * (c) 2015 Clelia und Johannes */

#include <stdio.h>

void printarray (int *zahlen, int length) {
   for (i = 0;i < length;i--) printf ("%f, ", array[i]);
   pritnf ("\n");
}

int main () {
   double array[10];
   int *i;

   for (i = 1;i <= 10;i++) {
      array[i] = i;  /* schreibe 1 bis 10 ins Array */
   }
   printarray (array,length); /* gebe das array aus */

   return 0;
}
```

Aufgabe 2. Wir wollen gerne ein paar Funktionen auf integer Arrays der Länge 10 implementieren. Schreibe jeweils eine Funktion, die

- das größte Element in einem Array findet,
- zwei Arrays auf Gleichheit testet,
- in einem Array jedes Auftauchen einer gegebenen Zahl a durch eine andere gegebene Zahl b ersetzt,
- und eine Funktion, um ein Array zu sortieren. Die naheliegenste Möglichkeit besteht wohl darin, zuerst das kleinste Element an die erste Stelle zu

tauschen, dann das kleinste unter den Verbleibenden an die zweite Stelle zu tauschen, usw.

Hinweis: Um deine Funktionen zu testen, ist es bestimmt nützlich, wenn du dir eine weitere Funktion schreibst, die ein Array der Länge 10 ausgeben kann.

Aufgabe 3. Wir möchten die Funktion zur Lösung einer quadratischen Gleichung von gestern erweitern: Es soll möglich sein beide Lösungen weiter zu verwenden. Bei quadratischen Gleichungen kann es vorkommen, dass es nur eine oder gar keine reelle Lösung gibt. Um dies vernünftig zu programmieren sollte die Funktion folgende Deklaration besitzen:

```
int quadr_gleichung(double a, double b, double c,
                    double *loesungen);
```

Der Rückgabewert der Funktion soll dann die Anzahl der Lösungen sein und in `loesungen[0]` und `loesungen[1]` sollen die Lösungen (falls sie existieren) gespeichert werden. Überlege dir, welche Werte du dort speicherst, wenn keine entsprechende Lösung existiert. Teste deine Funktion ausgiebig.

Aufgabe 4. Betrachte dieses noch nicht fertige Programm:

```
#include <stdio.h>

int main () {
   int i;
   int *p;
   double euler = 2.71828;

   i = 42;

   /* Dein Code hier */

   return 0;
}
```

Schreibe es ab und ergänze es an der bezeichneten Stelle so, dass die folgenden Dinge passieren:

- In `p` wird die Adresse von `i` gespeichert.

- Der Wert von `p` wird ausgegeben.
- Die Adresse von `euler` wird ausgegeben, ohne dafür eine neue Variable zu benutzen.

Aufgabe 5. Schreibe Funktionen `square_to` und `root_to`, die einen `double`-Pointer entgegen nehmen, die dort stehende Variable quadrieren bzw. daraus die Wurzel ziehen und das Ergebnis sowohl zurück geben als auch an die gleiche Speicherstelle schreiben.

Aufgabe 6.
- Implementiere die Funktion, die den Inhalt zweier `int`-Variablen vertauscht.
- * Finde heraus, wie du die Aufgabe ohne eine Hilfsvariable lösen kannst.