Beispiel 1

(3 Punkte)

Multiplikationstabelle

Erzeuge eine dreizeilige Multiplikationstabelle für eine Zahl, wobei die Zahl immer vom Benutzer abgefragt wird.

Die Ausgabe für 7:

```
  7  14  21  28  35 42  49  56  63  70
 77  84  91  98 105 112 119 126 133 140
147 154 161 168 175 182 189 196 103 210
```

Das ganze Programm soll in einer do-Schleife laufen und nach jeder Ausgabe soll der Benutzer gefragt werden, ob er eine weitere Tabelle berechnen will.

```
char choice = 'j';
do
  {
  …
  …


  printf("Wollen Sie noch eine weitere Tabelle anzeigen? (j/n): )";
  } while (choice =='j')
```

Beispiel 2

(4 Punkte)

Einlesen von Daten in ein Feld
Kopieren von Feldern
Ausgeben der Felder
- in jeweils einer Funktion

Erstelle zwei gleich große Arrays mit fünf `Integer`-Zahlen.
Initialisiere beide Felder mit 0-Werten (jeweils mit einer `for`-Schleife).

Gib in ein Feld Daten von Tastatur ein. Erstelle zum Eingeben der Daten eine eigene Funktion und übergib das Feld an diese Funktion.

Kopiere anschließend den Inhalt des einen Feldes in das andere. Leider ist eine Zuweisung

`Feld2[] = Feld1[]` oder `Feld2 = Feld1`

nicht möglich, sondern jeder einzelne Wert muss in das andere Feld geschrieben werden:

```
Feld2[i] = Feld1[i]
```

am besten mit Hilfe einer `for`-Schleife. Dabei muss natürlich darauf geachtet werden, dass das zweite Feld mindestens gleich groß wie das erste ist!

Erstelle zum Kopieren eine Funktion, der beide Felder übergeben werden.

Zum Ausgeben der Inhalte der beiden Felder erstelle ebenfalls eine eigene Funktion.

Beispiel 3

(4 Punkte)

Feldlänge feststellen

Mit der Funktion `strlen()` kann die Länge einer Zeichenkette festgestellt werden. Dies ist auf Grund des abschließenden Nullzeichens \0 in der Zeichenkette möglich.

Recherchiere im Online-Buch „C von A bis Z“ wie man die Länge von Nichtzeichenfeldern feststellen kann:

http://www.pronix.de/pronix-725.html

Erstelle ein kleines Beispiel dazu. Nenne das Programm `array_size.c`.

Beispiel 3
(zur Demonstration)

```
// programme:  pointer.c
// author:     Josef Strasser-Leitner
// date:       02.10.2005
// version:    1.00
// purpose:    to demonstrate the use of a pointer


#include <stdio.h>


// function: main function
// parameter values: none
// returned: 0
int main()
  {
  int myInteger = 1000;     // declares an integer with value 1000
  int * myIntegerPointer = &myInteger;
                           // declares a pointer to an integer
                           // and lets point it to myInteger

  printf("MyInteger: %d\n", myInteger);
                          // prints the value of the integer

  printf("MyIntegerPointer: %p\n", myIntegerPointer);
                          // prints the value of the pointer
  return 0;

  } // end main
```

Beispiel 4
(zur Demonstration)

```
// programme:  pointer2.c
// author:     Josef Strasser-Leitner
// date:       02.10.2005
// version:    1.00
// purpose:    to use pointer

#include <stdio.h>

// function: main function
// parameter values: none
// returned: 0
int main()
  {
  int number = 11;        // declares an integer with value 11
  int * numberPointer;    // declares a pointer to an integer
  numberPointer = &number;   // and lets point it to number

  printf("numberPointer points to: %d\n\n", *numberPointer);
                     //prints the number to that the pointer points:11


  // to read or write the value the pointer points to use *
    * numberPointer = 15;
    printf("numberPointer points to: %d\n", *numberPointer); //15
    printf("The value of number is: %d\n", number);          //15

  return 0;
  } // end main
```

Beispiel 5
(3 Punkte)

Zeiger kopieren

Deklariere eine String-Variable (`char`-Array) und lies ein Wort ein. Lasse einen Zeiger auf dieses Feld zeigen. Erzeuge einen zweiten Zeiger. Kopiere den ersten Zeiger in den zweiten. Gib den zweiten Zeiger aus.
Speichere das Programm mit dem Namen `copyPointer.c`.

Beispiel 6
(2 Punkte)

Auf Array mittels Zeiger zugreifen

Schreibe ein Programm, das Zeiger benützt um jedes Element eines Arrays auf 0 zu setzen. Speichere das Programm mit dem Namen `pointerIntoArray.c`.

Beispiel 7
(3 Punkte)

Verändere das Beispiel 7 - `change.c` vom zweiten Tag und versuche die Variablen als Original, also "by reference" und nicht "by value" zu übergeben.

Recherchiere im Online-Buch „C von A bis Z“, wie man eine Variable „by reference“ übergibt:
http://www.pronix.de/pronix-743.html

Speichere das Programm als `change-ref.c`.

Beispiel 8

(2 Punkte)

Verändere das Beispiel 9 (Texteingabe von Personendaten) vom Vortag so, dass anstatt der `char`-Arrays `char`-Zeiger verwendet werden.

Speichere das Programm mit dem Namen `char-pointers.c`.