Aufgabe 1: (15 Punkte)

Bei den Multiple-Choice-Fragen ist jeweils nur **eine** richtige Antwort eindeutig anzukreuzen. Auf die richtige Antwort gibt es die angegebene Punktzahl.

Wollen Sie eine Multiple-Choice-Antwort korrigieren, kreisen sie bitte die falsche Antwort ein und kreuzen die richtige an.

Lesen Sie die Frage genau, bevor Sie antworten.

a) Was ist der Unterschied zwischen den wie folgt in einer Funktion `foo` (lokal) definierten Variablen? 3 Punkte

```
void foo() {
    int a;
    static int b;
```

- ❏ Die Variable `a` ist nur für Funktionen in der Datei zugreifbar, in der `foo` definiert ist, während auf `b` auch von Funktionen in anderen Modulen des Programms zugegriffen werden kann.
- ❏ Der Speicherplatz der Variablen `a` wird jeweils beim Aufruf der Funktion `foo` angelegt und beim Verlassen wieder freigegeben, während der Speicherplatz der Variablen `b` von Programmstart bis -ende verfügbar ist.
- ❏ Die Variable `b` ist nur für Funktionen in der Datei zugreifbar, in der `foo` definiert ist. Funktionen in anderen Modulen des Programms können prinzipiell auf `a` zugreifen, hierzu muss `a` aber in dem entsprechenden Modul mit einer `extern`-Deklaration bekannt gemacht werden.
- ❏ Da der Speicherplatz der Variablen `b` für die gesamte Ausführungszeit des Programms reserviert ist, darf es in anderen Funktionen keine weitere Variable mit dem Namen `b` geben. Bei `a` gilt diese Einschränkung nicht.

b) Welche der folgenden Aussagen bzgl. der Interruptsteuerung ist richtig? 2 Punkt

- ❏ Pegel-gesteuerte Interrupts werden beim Wechsel des Pegels ausgelöst, daher der Name.
- ❏ Flanken-gesteuerten Interrupts können nicht blockiert werden, da sie völlig unvorhersehbar auftreten.
- ❏ Pegel-gesteuerte Interrupts werden immer wieder ausgelöst, so lange ein bestimmter Pegel anliegt.
- ❏ Wurde gerade ein Pegel-gesteuerter Interrupt ausgelöst, so muss erst ein Pegelwechsel der Interruptleitung stattfinden, bevor erneut ein Interrupt ausgelöst wird.

c) Welcher der folgenden Punkte ist kein Attribut einer Datei? 2 Punkte

- ❏ Dateiinhalt
- ❏ Dateigröße
- ❏ Eigentümer-Kennung
- ❏ Zugriffsrechte

d) Was passiert, wenn man das folgende Programmstück übersetzen und ausführen möchte: 2 Punkte

```
char *string;
string = "SPiC ist toll";
```

- ❏ Der Compiler wird beim Übersetzen einen Fehler melden, weil diese Art von Zugriff auf einen Zeiger nicht erlaubt ist.
- ❏ Der Variablen `string` wird der Zeiger auf die konstante Zeichenkette "SPiC ist toll" zugeordnet.
- ❏ Unter Betriebssystemen wie Unix oder Windows wird zur Laufzeit eine Schutzverletztung bzw. SIGSEGV ausgelöst, da dem Zeiger ein Wert zugeordnet wird, der Zeiger jedoch nicht initialisiert wurde.
- ❏ Die Zeichenkette "SPiC ist toll" ist konstant und darf daher keinem nicht-konstanten Zeiger zugewiesen werden.

e) Was versteht man unter auto-Variablen? 2 Punkte

- ❏ Der Speicher wird bei Betreten des Blocks / der Funktion reserviert und bei Verlassen wieder freigegeben.
- ❏ Automatische Variablen muss man nicht deklarieren, da sie automatisch angelegt werden.
- ❏ Eine auto-Variable behält automatisch ihren Inhalt für den nächsten Funktionsaufruf.
- ❏ Auto-Variablen werden bei jedem Schleifendurchlauf automatisch erhöht.
- ❏ Auto-Variablen haben immer automatisch den gewünschten Inhalt.

f) Welche Aussage zu C-Strings ist richtig? 2 Punkte

- ❏ C-Strings speichern im ersten Byte ihre Länge.
- ❏ C-Strings sind Null-terminiert.
- ❏ C-Strings werden durch den Datentyp `String` repräsentiert.
- ❏ C-Strings können maximal 16.7 Mio Zeichen enthalten.

g) In Betriebssystemen wie UNIX oder Linux unterscheidet man die Begriffe Programm und Prozess. Welche Aussage ist richtig? 2 Punkte

- ❏ Ein Programm darf immer nur einmal gleichzeitig auf einem Rechner gestartet werden.
- ❏ Ein Programm kann mehrfach gestartet werden. Jede Programmausführung erfolgt in einem Prozess. Tritt bei einer dieser Programmausführungen ein Fehler auf (z. B. Segmentation Fault), so werden alle Prozesse, die das Programm gerade ausführen abgebrochen.
- ❏ Die Begriffe Programm und Prozess bedeuten eigentlich das Gleiche. Der Begriff Programm stammt aus der Windows-Welt, während man in Linux-Systemen statt dessen meisten von Prozessen spricht.
- ❏ Ein Programm kann mehrfach ausgeführt werden. Jede Programmausführung hat dabei ihre eigenen Daten und kann auf die Daten anderer Ausführungen des gleichen Programms nicht zugreifen.

Aufgabe 2: (30 Punkte)

a) Ein Mikrocontroller bietet mehrere Ports mit jeweils 8 Anschluss-Pins an. Die Ports können über zugeordnete Register angesteuert werden. Die Register sind direkt über Speicheradressen zugreifbar.

Schreiben Sie eine Funktion

```
int get_and_set_pin(unsigned char *port, int pin, int pegel);
```

die Folgendes leistet:

Es wird der Wert des Pin Nummer **`pin`** (Wert von 0 bis 7) vom angegebenen Port **`port`** (Wert = Adresse) abgefragt. Es soll **`0`** zurückgegeben werden, wenn der Pin gesetzt ist, ansonsten **`1`**. Außerdem wird der Wert des Pins neu auf den angegebenen **`pegel`** gesetzt.

Sie können davon ausgehen, dass die Funktion nicht mit falschen Parametern aufgerufen wird.

```
/* Funktion get_and_set_pin */
```

Vervollständigen Sie die Makrodefinition für den folgenden Aufruf der Funktion. Port A ist über die Adresse 0x3B erreichbar.

```
#define PORTA
x = get_and_set_pin(PORTA, 5, 1);
```

G:

b) Schreiben Sie ein Programm **suche**, das Zeichen in ein Feld einliest und anschließend nach einem Muster sucht, welches auf der Kommandozeile als Argument übergeben wurde. Wird das Muster in der eingelesenen Zeichenkette gefunden, so wird die Position (evtl. auch mehrere) auf der Standardausgabe ausgegeben.

Beispiel:
```
suche fisch
fischers fritz fischt
^D
```
Ausgabe:
```
1 16
```

(d. h. ab dem 1. und ab dem 16. Zeichen kommt das Muster `fisch` in den eingelesenen Zeichen vor).

Beachten sie dabei Folgendes:

- Das Feld soll bis zu 1000 Zeichen aufnehmen können. Damit dieser Wert in Ihrem Programm leicht verändert werden kann, sollen Sie ihn nicht direkt verwenden, sondern hierfür ein Makro namens **`SIZE`** benutzen.
- Falls der Benutzer kein Muster beim Aufruf angibt, soll eine Fehlermeldung ausgegeben werden und das Programm abbrechen.
- Das Einlesen der Zeichen soll in einer eigenen Funktion
  `int read_input(char feld[], int feldlen);`
  erfolgen.
  Die Funktion erhält das Feld und dessen Länge als Parameter, liest Zeichen von der Standardeingabe in das Feld ein und liefert die Zahl der gelesenen Zeichen als Ergebnis zurück.

Aus der Standard-C-Bibliothek stehen Ihnen folgende Funktionen zur Verfügung:

- `int getchar(void);`
  liest ein Zeichen von `stdin`. Liefert am Ende der Eingabe `EOF`.
- `int strncmp(char *string1, char *string2, int len);`
  vergleicht die ersten `len` Zeichen von zwei Zeichenketten. Falls identisch, wird `0` zurückgeliefert, sonst ein anderer Wert.
- `int strlen (char *string)`
  liefert die Länge der übergebenen Zeichenkette.

Ergänzen Sie das folgende Codegerüst so, dass ein vollständig übersetzbares Programm entsteht.


```
#include <stdio.h>
#include <strings.h>
/* Makros, Funktiondeklarationen, etc. */

/* Funktion main */

   /* Kommandozeilen-Parameter ueberpruefen */

   /* Zeichenkette einlesen */

   /* Muster an allen Feldpositionen suchen
      und Position ausgeben, falls gefunden */

   /* Ende der main-Funktion */
```

S:

```
/* Funktion read_input
   Zeichen einlesen und Feld mit \0 abschliessen */
```

R:

Aufgabe 3: (15 Punkte)

Die folgenden Beschreibungen sollen kurz und prägnant erfolgen (Stichworte, kurze Sätze)

Es gibt zwei verschiedene Möglichkeiten, wie ein Programm mit Ereignissen an Peripherie-Schnittstellen umgehen kann: Polling und Interrupt-Verarbeitung.

a) Warum könnte ein externes Gerät die "Aufmerksamkeit" der CPU / des Programmes wünschen? Nennen sie 3 Beispiele.

b) Beschreiben Sie die beiden Verfahren (Polling und Interrupts) in Stichpunkten und geben Sie jeweils die Unterschiede / Vor- und Nachteile an.

c) Wo entsteht in diesem Zusammenhang Nebenläufigkeit, wie können dadurch Probleme entstehen und was kann man beispielsweise dagegen tun?