TU Hamburg-Harburg – Institut für Zuverlässiges Rechnen
Prof. Dr. S. M. Rump und Mitarbeiter, Wintersemester 2018/2019

Programmieren in C, Testat 4

1. Schreiben Sie ein Programm, dass eine natürliche Zahl n von der Tastatur einliest und dann ein `char`-Array `str` der Länge $n+1$ anlegt.

   Lesen Sie nun ein Wort der Länge $\leq n$ von der Tastatur in `str` ein und anschließend einen Index $k \in \{0, 1, \ldots, n-1\}$. Geben Sie dann das Restwort ab Index k aus.

   Lesen Sie jetzt eine natürliche Zahl $a \in \{0, 1, \ldots, 127\}$ von der Tastatur ein, setzen Sie `str[k] = (char)a` und geben Sie dann den gesamten String `str` aus.

   Beispiel: $n = 20$, $k = 4$, $a = 104$, `str = "Schulfach"`

   Ausgabe 1: `"lfach"`, Ausgabe 2: `"Schuhfach"`

   Testen Sie Ihr Programm mit obigem Beispiel auch für $a \in \{0, 8, 9, 10\}$ und erklären Sie die jeweilige Ausgabe genau! Es genügt nicht, nur ein lauffähiges Programm vorzuzeigen.

   Es dürfen nur in der Vorlesung behandelte Techniken verwendet werden.

2. Es sollen Autodaten in Strukturen gespeichert werden. Legen Sie zuerst eine Struktur `s_engine` zur Speicherung von Motordaten an. Sie soll die Komponenten `power` und `volume` für Motorleistung in PS und Hubraum in cm^3 haben. Definieren Sie dann eine zweite Struktur `s_car` mit folgenden vier Komponenten zur Speicherung der Autodaten:

   - `brand` : Automarke, String der Länge 30
   - `engine` : Motordaten, Datentyp `struct s_engine`
   - `speed` : Höchstgeschwindigkeit in km/h
   - `size` : Array der Länge 3 für die Abmessungen Länge, Breite, Höhe in mm

   Definieren Sie mit `typedef` einen Datentyp `pt_car` für Zeiger auf `struct s_car`. Reservieren Sie mit der Funktion `malloc` Speicher für zwei Autos und legen Sie die Anfangsadresse dieses Speicherbereichs in einem Zeiger `p` vom Typ `pt_car` ab. Lesen Sie nun mittels `p` die Daten der beiden Autos von der Tastatur in den reservierten Speicherbereich ein. Verwenden Sie dazu die Pfeilnotation `p->`. Lesen Sie zur Kontrolle den Speicherbereich mittels `p` gleich wieder aus und zeigen Sie die Autodaten auf dem Bildschirm an. Eine Ein-/Ausgabe soll beispielsweise folgendermaßen aussehen:

```
Car 1:
  brand         : BMW_X1         <--- Eingabe der Autmarke
  power  [PS]   : 140            <--- Eingabe der Motorleistung
  volume [cm^3] : 1499           ...
  speed  [km/h] : 205
  length [mm]   : 4439
  width  [mm]   : 1821
  hight  [mm]   : 1598

 BMW_X1, 140.00 PS, 1499 cm^3, 205.00 km/h, L/W/H[mm]: 4439/1821/1598  <--- Ausgabe Auto 1

Car 2:
  brand         : Audi_A1        <--- Eingabe der Autmarke
  power  [PS]   : 82             <--- Eingabe der Motorleistung
  volume [cm^3] : 999            ...
  speed  [km/h] : 177
  length [mm]   : 4029
  width  [mm]   : 1740
  hight  [mm]   : 1409

 Audi_A1, 82.00 PS, 999 cm^3, 177.00 km/h, L/W/H[mm]: 4029/1740/1409  <--- Ausgabe Auto 2
```

English translation

1. Write a program that reads an integer $n > 0$ from the keyboard and defines a `char`-array `str` of length $n + 1$.

   Read a word of length $\leq n$ from the keyboard and store it as a string in `str`. Afterwards, read an index $k \in \{0, 1, \ldots, n-1\}$ and display the trimmed word starting from index k.

   Now, read an integer $a \in \{0, 1, \ldots, 127\}$ from the keyboard, set `str[k] = (char)a`, and display the whole string `str`.

   Example: $n = 20$, $k = 4$, $a = 104$, `str = "Schulfach"`

   Output 1: `"lfach"`, Output 2: `"Schuhfach"`

   Test your program with the given example also for $a \in \{0, 8, 9, 10\}$ and explain the output in detail! Only techniques from the lecture are allowed to use.

2. Car data shall be stored in structures. First, define a structure `s_engine` for storing car engine data. It shall contain the components `power` and `volume` for engine power (unit PS) and displacement (unit cm^3). Next, define a structure `s_car` with the following components for storing the car data:

   - `brand` : string of length 30
   - `engine` : data type `struct s_engine`
   - `speed` : top speed in km/h
   - `size` : array of length 3 for car length, width, and height in mm

   Use `typedef` to define a data type `pt_car` for pointers to `struct s_car`. Use the function `malloc` to reserve storage for two cars and store the start address of that storage in a pointer `p` of data type `pt_car`. Now, read the data of the two cars from the keyboard into the reserved storage. Use the arrow notation `p->` for that. Just in order to check that everything was stored correctly, read the car data from the storage and display them on the screen. For example, an input/output of your program shall look as follows:

```
Car 1:
  brand         : BMW_X1         <--- enter car brand
  power  [PS]   : 140            <--- enter engine power
  volume [cm^3] : 1499           ...
  speed  [km/h] : 205
  length [mm]   : 4439
  width  [mm]   : 1821
  hight  [mm]   : 1598

 BMW_X1, 140.00 PS, 1499 cm^3, 205.00 km/h, L/W/H[mm]: 4439/1821/1598  <--- output car 1

Car 2:
  brand         : Audi_A1        <--- enter car brand
  power  [PS]   : 82             <--- enter engine power
  volume [cm^3] : 999            ...
  speed  [km/h] : 177
  length [mm]   : 4029
  width  [mm]   : 1740
  hight  [mm]   : 1409

 Audi_A1, 82.00 PS, 999 cm^3, 177.00 km/h, L/W/H[mm]: 4029/1740/1409  <--- output car 2
```