Übungsblatt 5

In dieser Aufgabe sollen Sie sich mit abstrakten Datentypen vertraut machen und die Vorgehensweise beim Definieren eigener Datentypen üben.

Aufgabe 1 – Inventar als doppelt verkettete Liste

In dieser Aufgabe soll ein Inventar, wie man es aus Rollenspielen in Computerspielen kennt, implementiert werden. Das Inventar soll dabei als doppelt verkettete Liste realisiert werden. Wir beginnen zunächst mit mehreren Struktur-Definitionen. Überlegen Sie sich selbst, welche Komponenten Sie sinnvollerweise als Pointer anlegen und bei welchen dies nicht notwendig ist.

In dem Inventar können Gegenstände abgelegt werden. Definieren Sie zunächst eine passende Struktur **Gegenstand**. Ein Gegenstand besteht aus einem Namen und einem Gewicht (als int).

Die Größe des Inventars und wie viel Gewicht getragen werden kann, wird über den verwendeten Rucksack definiert. Definieren Sie als nächstes eine Struktur **Rucksack**. Ein Rucksack hat eine maximale Anzahl an erlaubten Gegenständen und ein maximal erlaubtes Tragegewicht.

Das Inventar soll als doppelt verkettete Liste implementiert werden. Eine Liste besteht aus Listenelementen. Definieren Sie als nächstes eine Struktur **InventarElement**. Ein InventarElement speichert einen Gegenstand und die Zeiger auf das vorherige und nachfolgende InventarElement.

Abschließend definieren Sie noch das eigentliche **Inventar** als Struktur. Das Inventar besteht aus einem Rucksack und einem Zeiger auf das erste und letzte InventarElement der verketteten Liste. Der Rucksack in dem Inventar bestimmt also, wie viele Gegenstände in dem Inventar erlaubt sind und wie schwer diese in der Summe maximal sein dürfen.

Als nächstes deklarieren und initialisieren Sie mehrere Struktur-Variablen, wie in dem folgenden Programmabschnitt zu sehen:

```
int main()
{
    system("chcp 1252");
    Gegenstand axt = {"Axt", 5};
    Gegenstand stab = {"Zauberstab", 3};
    Gegenstand helm = {"Stahlhelm", 7};

    Rucksack kleinerRucksack = {10,3};

    Inventar *inventar = createInventar(&kleinerRucksack);

    printInventar(inventar);

    return 0;
}
```

Implementieren Sie als erstes die Funktion **createInventar** und anschließend die Funktion **printInventar**. Damit können Sie immer testen, ob die weiteren Funktionen richtig funktionieren.

```
*** Inventar ***
Rucksack:
        erlaubtes Gewicht: 10
        erlaubte Anzahl: 3
Inhalt:
        Rucksack leer
```

Implementieren Sie abschließend folgende neue Funktionen

- **countGegenstaendeUndGewicht**:
  Berechnet die Anzahl der Gegenstände und das Gesamtgewicht aller Gegenstände im Inventar

Ändern Sie die **printInventar** Funktion nun so ab, dass sie auch die Anzahl und das Gewicht der Gegenstände ausgibt.

```
*** Inventar ***
Rucksack:
        Gewicht: 0 von 10
        Anzahl: 0 von 3
Inhalt:
        Rucksack leer
```

- **addGegenstand:**
  Fügt dem Inventar den Gegenstand hinzu. Jedoch nur, wenn der verwendete Rucksack diesen Gegenstand noch aufnehmen kann (d.h. Anzahl und Gewicht sind nicht verletzt).

Fügen Sie dem Rucksack nun die Axt hinzu und testen Sie mit der print-Funktion, ob alles richtig funktioniert hat. Testen Sie weiterhin, ob Sie nicht zu viele und zu schwere Gegenstände hinzufügen können.

```
*** Inventar ***
Rucksack:
        Gewicht: 5 von 10
        Anzahl: 1 von 3
Inhalt:
        Axt
```

- **removeGegenstand**:
  Entfernt den Gegenstand aus dem Inventar

- **changeRucksack:**
  Wechselt den verwendeten Rucksack im Inventar. Sollten danach zu viele Gegenstände im Inventar sein, werden so lange Gegenstände abgeworfen, bis die Bedingung nicht mehr verletzt ist. Sollte nach dem Wechsel das maximal erlaubte Gewicht überschritten werden, dann werden so viele Gegenstände abgeworfen, bis das Gewicht nicht mehr überschritten ist.