Úloha 1

- vytvořte datovou strukturu zbozi obsahující položky:
  - nazev – název zboží
  - cena – cena za kus
  - kusu – počet kusů na skladě
- vytvořte konstantu sklad – seznam obsahující několik záznamů se zbožím

Řešení 1

```
(define-struct zbozi (nazev cena kusu))

(define sklad (list
    (make-zbozi "rohlík" 2.90 42)
    (make-zbozi "Savo" 59.90 15)
    (make-zbozi "Mattoni" 15.90 22)))
```

Úloha 2

- vytvořte funkci (**najdi-cenu** jmeno seznam), která dostane název zboží a vydá jeho cenu za kus
- pokud zboží daného jména není v seznamu, funkce vrátí 0

Řešení 2

```
(define (najdi-cenu jmeno seznam)
    (cond
        [(empty? seznam) 0]
        [(string=? jmeno (zbozi-nazev (car seznam)))
                (zbozi-cena (car seznam))]
         [else (najdi-cenu jmeno (cdr seznam))]))
```

Úloha 3

- vytvořte funkci (**celkova-hodnota** seznam), která dostane seznam datových struktur typu zbozi a vydá jeho celkovou cenu (počty kusů krát jednotková cena pro všechna zboží ze seznamu)

Řešení 3

```
(define (celkova-hodnota seznam)
    (if (empty? seznam) 0
        (+ (*  (zbozi-cena (car seznam))
               (zbozi-kusu (car seznam)))
           (celkova-hodnota (cdr seznam)))))
```

Úloha 4

- vytvořte funkci
  (**cena-objednavky** objednavka sklad)
  která pro objednávku ve tvaru
  ((jméno1 počet_kusů1) (jméno2 počet_kusů2) ...)
  spočítá celkovou cenu objednaného zboží
- např.
  (cena-objednavky '(("meloun" 2) ("jablko" 4)) sklad)
  spočítá cenu za 2 melouny a 4 jablka podle cen ze skladu

Řešení 4

```
(define (cena-objednavky objednavka seznam)
    (if (empty? objednavka) 0
        (+  (cena-polozky (car objednavka) seznam)
            (cena-objednavky (cdr objednavka) seznam))))

(define (cena-polozky polozka seznam)
    (*  (najdi-cenu (car polozka) seznam)
        (second polozka)))
```

Úloha 5

- vytvořte funkci (**vydej** seznam název kusů), která ze seznamu odebere příslušný počet kusů zboží s daným názvem
- vrací upravený seznam
- rozšíření: klesne-li počet kusů daného zboží na nulu, odstraňte je ze seznamu

Řešení 5

```
(define (vydej seznam nazev kusu)
    (cond
        [(empty? seznam) '()]
        [(string=? nazev (zbozi-nazev (car seznam)))
            (odeber-kusy seznam kusu)]
        [else (cons (car seznam)
                    (vydej (cdr seznam) nazev kusu))]))
```

Řešení 5

```
(define (odeber-kusy seznam kusu)
    (if (>= kusu (zbozi-kusu (car seznam))) (cdr seznam)
        (cons
            (make-zbozi  (zbozi-nazev (car seznam))
                         (zbozi-cena (car seznam))
                         (-  (zbozi-kusu (car seznam))
                             kusu))
            (cdr seznam))))
```

Úloha 6

- vytvořte funkci (**serad-abecedne** sklad), která seřadí položky ve skladu abecedně podle názvů zboží
- vrací sklad s upraveným pořadím zboží

Řešení 6

- použijeme insert sort – seřadíme zbytek a následně zařadíme původní 1. položku na správné místo

```
(define (serad sklad)
    (if (empty? sklad) '()
        (zarad   (car sklad)
                 (serad (cdr sklad)))))
```

Řešení 6

- sklad je seřazen, zařadíme novou položku

```
(define (zarad polozka sklad)
    (cond
        [(empty? sklad) (list polozka)]
        [(string<=? (zbozi-nazev polozka)
                    (zbozi-nazev (car sklad)))
                                    (cons polozka sklad
        [else (cons (car sklad)
                    (zarad polozka (cdr sklad)))]))
```