Софийски университет „Св. Климент Охридски“
Факултет по математика и информатика

Домашна работа 1

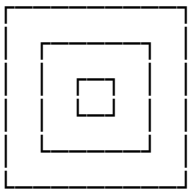
курс Функционално програмиране
за специалности Информатика и Компютърни науки (2-ри поток)
зимен семестър 2020/21 г.

Важно: Освен на това, което е описано в документа “Схема за оценяване”, решенията ви трябва да отговарят и на дадените по-долу изисквания:

1. Решете задачите на Scheme;
2. Решенията ви трябва да се изграждат успешно и да работят на Racket Scheme с избран език R5RS;
3. За всяка задача трябва да предадете точно един файл съдържащ решението. Кръстете файловете си 1.scm, 2.scm и 3.scm;
4. В решенията не може да се използват списъци, нито деструктивни операции;
5. Освен ако не се изисква в задачата или не е абсолютно наложително, не използвайте (begin) или (display). Докато разработвате решението си, може да бъде полезно да трасирате, като извеждате информация чрез (display), но в предадените от вас работи не трябва да се извежда друг текст освен този, който се изисква в условието.

Задача 1

Напишете функция (squares n), която извежда на екрана поредица от концентрични квадрати. Когато реализирате функцията (squares) или други помощни функции, не реализирайте сами рекурсия, а използвайте подходящи обръщения към (accumulate). Например (squares 3) трябва да изведе:



За да оформите фигурата, използвайте дадените по-долу символи. Забележете, че за да се получи квадрат, броят на хоризонталните черти трябва да бъде по-голям от вертикалните (източник на символите: https://unicode.org/charts/PDF/U2500.pdf).

250C	┌	BOX DRAWINGS LIGHT DOWN AND RIGHT
2510	┐	BOX DRAWINGS LIGHT DOWN AND LEFT
2514	└	BOX DRAWINGS LIGHT UP AND RIGHT
2518	┘	BOX DRAWINGS LIGHT UP AND LEFT
2500	─	BOX DRAWINGS LIGHT HORIZONTAL
2502	│	BOX DRAWINGS LIGHT VERTICAL

Задача 2

Множество от естествени числа можем да представим чрез едно естествено число, което ще наричаме негово представяне. Числото n се съдържа в множеството тогава и само тогава, когато n+1-вият разряд в представянето е 1. Например

{5, 1, 0}	35; двоичният запис на 35 е $100011_{(2)}$
{4, 3, 1}	26; двоичният запис на 26 е $11010_{(2)}$
∅	0; двоичният запис на 0 е $0_{(2)}$

Реализирайте дадените по-долу функции. Всяка от тях трябва да връща резултата от прилагането на съответната операция, освен предикатите, които връщат #t или #f.

```
(set-add set elem)      ; Добавя елемент в множество.
(set-remove set elem)   ; Премахва елемент от
множество.
(set-contains? set elem) ; Проверява дали елемент се
съдържа в множество
(set-empty? set)        ; Проверява дали дадено
множество е празно.
(set-size set)          ; Намира размера на дадено
множество
```

```
(set-intersect s1 s2)    ; Намира сечението на две
множества
(set-union s1 s2)        ; Намира обединението на две
множества
(set-difference s1 s2)   ; Намира разликата s1 \ s2.
```

Упътване: обърнете внимание, че става дума за множества. Например ако добавим 5 към множеството {1, 2, 5}, резултатът е пак {1, 2, 5}.

Използвайте функциите, за да решите следната задача:

Предполагаме, че са ни дадени число n, което представя брой предмети. Дадени са и следните две функции:

(w i) -- връщат теглото на i-ия предмет ($0 \le i \le n$)
(p i) -- връща цената на i-ия предмет ($0 \le i \le n$)

Напишете функция (knapsack c n w p), когато решава задачата за раницата за n предмета и капацитет на раницата c. Функцията трябва да върне множеството от индексите на елементите, които се включват в решението. Ако задачата няма решение, да се върне празното множество.

Задача 3

Аритметичен израз ще представяме като символен низ по следния начин:

- Празният низ е валиден израз.
- Ако n е естествено число, то n е валиден израз.
- Ако a и b са аритметични изрази, то и следните изрази също са: a+b, a-b, a*c, a/c, a^c. Те представят съответно събиране, изваждане, умножение, деление и степенуване.

Преди и след всяко число в израз позволяваме да има произволен брой празни (whitespace) символи. По-долу са дадени примери за валидни изрази:

```
""
"10"
"    10   "
"10+5*2"
"  10 + 5       *2"
```

Изразите оценяваме съгласно стандартните правила на аритметиката. За пълнота считаме, че (1) празният израз има оценка нула, (2) всички операции са ляво-асоциативни.

Реализирайте следните функции:

```
(expr-valid? expr) ; проверява дали expr е валиден
израз. Например:
                   ; (expr-valid? "10   + 20") → #t
                   ; (expr-valid? "10 20 + 5") → #f

(expr-rp expr)     ; връща представянето на expr в
обратен полски запис.
                   ; За разделител между аргументите
да се използва запетая.
                   ; Например:
                   ;   (expr-rp "10+20*30") → "+10,*20,30"

(expr-eval expr)  ; Връща стойността на израза expr.
Например:
                  ; (expr-eval "10+20*30") → 610
                  ; (expr-eval "") → 0
```