Задача 1. Дадено е двоично дърво с върхове, съдържащи низ. Да се напише булева функция check_tree, която проверява дали при конкатенацията на низовете във върховете на всяко ниво (в реда, в който те са подредени), се получава едно и също изречение (в корена е цялото изречение).

Задача 2. Дадено е двоично дърво от двусвързани списъци от цели числа. Да се дефинира функция, която проверява дали списъците, в които завършват два пътя в дървото, започващи от корена и зададени с низ, са огледални един на друг. Пътят съдържа символите L и R, които означават съответно движение към лявото и дясното поддърво.

Вход: дърво [(1) [(3) [(4) [] [(1 2 3) [] []]] []] [(5) [(3 2 1) [] []] [ ]]] пътища: LLR и RL

Резултат: true (пътят LLR завършва във възела (1 2 3), пътят RL завършва във възела (3 2 1), огледални са)

Задача 3. Дадени са бинарните операции:

a$b = min(a,b) и a@b = (a+b)%10,

където a и b са едноцифрени числа. Да се напише програма, която изчислява изрази от вида:

<израз> = <цифра>|(<израз><операция><израз>)

<операция> = $|@

Пример: 5; (2$3); (2@(3$4)) и т.н.

Задача 4. Път в дадено двоично дърво наричаме последователност от възли от корен до листо на дървото. Казваме, че един списък е огледален образ на друг, ако вторият списък съдържа елементите на първия в обратен ред. Да се напише функция, която по дадени две двоични дървета намира път в първото дърво, който е огледален образ на някой път във второто дърво.

Задача 5. Да се дефинира функция

bool sentenceOnLevel ([подходящ тип]t, int k, char *fname),

която за дадено ниво k от двоичното дърво t проверява дали елементите на нивото k в t, четени от ляво на дясно, образуват изречение от файла с име fname.

Задача 6. Да се дефинира функция

bool sequenceOnLevel ([подходящ тип]t, int k, char *fname),

която за дадено ниво k от двоичното дърво t проверява дали елементите на нивото k в t, четени от ляво на дясно, образуват редица над файла с име fname.

Задача 7. Да се създаде (чрез включване) двоично наредено дърво от интервали от числа ($[a, b]$, $a \leq b$). Наредбата да е по дължината на интервала. Да се намери интервалът, съдържащ всички интервали, които участват в дървото.

Задача 8. Дадено е двоично наредено дърво с върхове - цели числа. Да се напише функция, която разменя левия и десния наследник на всеки от върховете на дървото и заменя всяка стойност x на връх на дървото с –x.

Задача 9. Да се напише функция, която пресмята сумата на две неотрицателни числа с произволна дължина, записани под формата на двоични дървета, прочетени в реда ДКЛ(дясно, корен, ляво). Резултатът да се отпечата на стандартния изход, а функцията да връща подходяща стойност, чието значение да бъде успешното ѝ изпълнение.

Задача 10. Да се реализира шаблон на опашка като се използват два стека. Всяка операция, валидна за една опашка да бъде реализирана само чрез функционалността на структурата от данни стек. Старайте се да не правите излишни операции, както и да не съхранявате един елемент на повече от едно място.

Задача 11. Дадена е редицата числа, чиито членове се получават по следния начин:

- първият елемент е N;
- вторият се получава като съберем N с 1;
- третият – като се умножи първия с 2 и така последователно всеки елемент се събира с 1 и се добавя в края на редица, след което се умножава по 2 и отново се добавя в редицата.

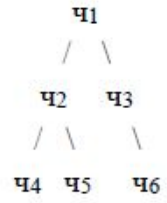
Да се напише програма, която за дадено N и p намира p-тия пореден елемент на редицата.

Задача 2

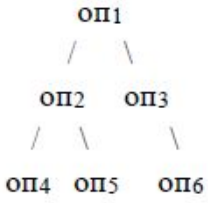
Дадени са две двоични дървета с еднаква структура – дърво от числа и дърво от операции, представени със символите +, - и *.

По-долу са дадени примери за такива дървета.

Дърво от числа:



Дърво от операции:



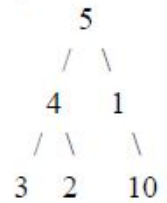
Да се напише функция, която пресмята сумата:

$$\sum_{i=1}^{n} s_i,$$

където n е броят на върховете в кое да е от дърветата, а $s_i = \text{ч}_i \, \text{оп}_i c_i$, а c_i е броят на върховете в поддървото с корен върха ч_i (включително върха ч_i).

Пример:

Дърво от числа:



Дърво от операции:

Резултат: (5 * 6) + (4 + 3) + (1 – 2) + (3 + 1) + (2 * 1) + (10 – 1).