Домашна работа № 3 по Функционално програмиране
специалност „Информационни системи“, I курс
2023/2024 учебна година

Решенията трябва да са готови за автоматично тестване. Важно е програмният код да бъде добре форматиран и да съдържа коментари на ключовите места. Предайте решенията на всички задачи в *един* файл с наименование *hw3_<FN>.hs*, където *<FN>* е Вашият факултетен номер.

Домашните работи се предават като изпълнение на съответното задание в курса по ФП в Moodle (https://learn.fmi.uni-sofia.bg/course/view.php?id=10042) най-късно до 23:55 ч. на 04.06.2024 г. (вторник).

Приятна работа и успех!

Файловата система е дърво от файлове и директории (които могат да съдържат други директории и файлове). Най-външната директория има името `"/"`. Обхождането на файловата система винаги започва от директорията `"/"`. То се извършва с командата **`cd`** за промяна на текущата активна директория и командата **`ls`** за извеждане на всички файлове и директории, непосредствено съдържащи се в текущата активна директория. Редовете с команди започват със символа **`$`**, като:
`$ cd x` намира в текущата директория директорията с име **`x`** и я прави текуща директория;
`$ cd ..` намира директорията, която съдържа текущата директория, и я прави текуща директория;
`$ ls` отпечатва информация за всички файлове и директории, непосредствено съдържащи се в текущата директория, на отделен ред по следния начин:

- **`123 abc`** означава, че текущата директория съдържа файл с име `abc` с размер `123`;
- **`dir xyz`** означава, че текущата директория съдържа директория с име `xyz`.

Нека разгледаме следната последователност от команди и върнатите от тях резултати:

```
1: $ cd /                 9: dir e                17: $ cd ..
2: $ ls                  10: 29116 f              18: $ cd d
3: dir a                 11: 2557 g               19: $ ls
4: 14848514 b.txt        12: 62596 h.lst          20: 4060174 j
5: 8504156 c.dat         13: $ cd e               21: 8033020 d.log
6: dir d                 14: $ ls                 22: 5626152 d.ext
7: $ cd a                15: 584 i                23: 7214296 k
8: $ ls                  16: $ cd ..
```

Обходената чрез горните команди файлова система изглежда така:

```
- / (директория)
  - a (директория)
     - e (директория)
          - i (файл, размер=584)
     - f (файл, размер=29,116)
     - g (файл, размер=25,57)
     - h.lst (файл, размер=62,596)
  - b.txt (файл, размер=14,848,514)
  - c.dat (файл, размер=8,504,156)
  - d (директория)
     - j (файл, размер=4,060,174)
     - d.log (файл, размер=8,033,020)
     - d.ext (файл, размер=5,626,152)
     - k (файл, размер=7,214,296)
```

Тя се състои от четири директории: **/** (най-външната директория), **a** и **d** (които са в **/**) и **e** (която се намира в **a**). Възможно е да има директории и файлове с едно и също име, стига те да се намират в директории с различни имена. Общият размер на една директория е сумата от размерите на файловете, които съдържа, пряко или непряко.

Нека са дефинирани следните типове:

```
type Command = String
type Size = Int
type Name = String
```

Типът `Size` представя размер на файл или директория. Типът `Name` представя име на файл или директория. Имената могат да се състоят само от малки латински букви и символа **'.'**. Типът `Command` представя команда или върнат от командата резултат.

Примерната последователност от команди и върнати резултати може да се представи като следния списък в Haskell:

```
commands :: [Command]
commands = ["$ cd /","$ ls","dir a","14848514 b.txt","8504156
c.dat","dir d","$ cd a","$ ls","dir e","29116 f","2557 g","62596
h.lst","$ cd e","$ ls","584 i","$ cd ..","$ cd ..","$ cd d","$
ls","4060174 j","8033020 d.log","5626152 d.ext","7214296 k"]
```

Нека е дефиниран следният алгебричен тип, който представя дървовидната структура на файловата система:

```
data FileSystem = Directory Name [FileSystem] | File Name Size
  deriving (Eq, Show)
```

Задача 1. Дефинирайте функция **`generateFileSystem :: [Command] -> FileSystem`**, която получава коректна последователност от команди и върнатите от тях

резултати, представена като списък от низове. Функцията трябва да върне съответната им дървовидна структура.
Списъкът в конструктора `Directory` трябва да е сортиран по следния начин:

- всички стойности с конструктор `Directory` трябва да са в началото на списъка и да са подредени по азбучен ред на тяхното име, без значение от малки и главни букви;
- всички стойности с конструктор `File` трябва да са в края на списъка и да са подредени по азбучен ред на тяхното име, без значение от малки и главни букви.

Пример:

```
generateFileSystem commands → Directory "/" [Directory "a"
[Directory "e" [File "i" 584], File "f" 29116, File "g" 2557, File
"h.lst" 62596], Directory "d" [File "d.ext" 5626152, File "d.log"
8033020, File "j" 4060174, File "k" 7214296], File "b.txt" 14848514,
File "c.dat" 8504156]
```

Задача 2. Дефинирайте функция **`getParentSize :: FileSystem -> Name -> Size`**, която получава дървовидно представяне на файлова система `fs` и име на файл `filename`. Функцията трябва да върне размера на най-малката по размер директория на `fs`, в която се намира файлът с име `filename`.
Функцията да връща стойност -1, ако такъв файл не съществува.

Пример:

```
getParentSize (generateFileSystem commands) "i"     → 584
getParentSize (generateFileSystem commands) "g"     → 94853
getParentSize (generateFileSystem commands) "b.txt" → 48381165
getParentSize (generateFileSystem commands) "abc"   → -1
```