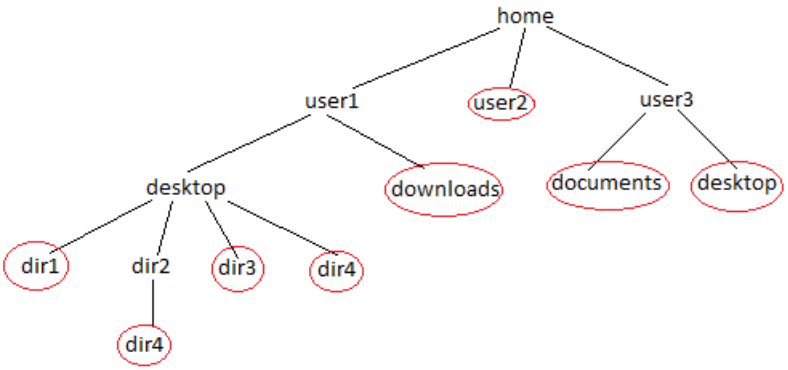
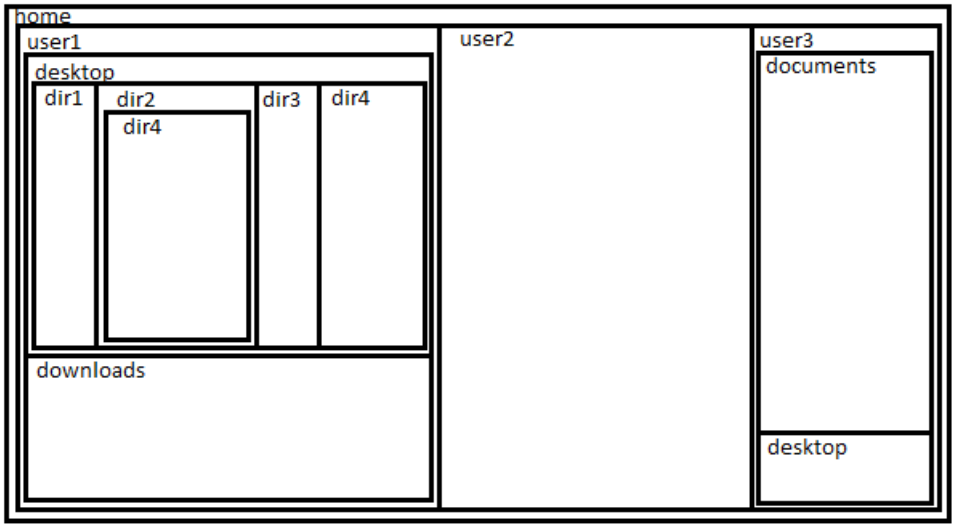
CENG242

Homework #2

(Due: April 1st, Friday 23:55)

Assume that we want to visualize the storage size (size on disk) of each subfolder of a given directory on a rectangular space (for ex: your laptop's screen). The hierarchy of these folders is going to be displayed with sub-rectangles, where the area of each sub-rectangle will be proportional to the storage size of the subfolder it corresponds to. The structure that displays the hierarchical data as a set of nested rectangles is called **treemap**. Our goal is to form a treemap representation of the specified directories by creating rectangles for each subfolder. We want to make the aspect ratio of rectangles (i.e. width / height) to be close to 1, in order to avoid creating rectangles that are too thin or too tall. Below there is a tree and treemap representation of the home directory.





In this homework, you will be given a list of directories (for ex: directories of red circled folders), storage size of them (for ex: sizes of red circled folders), a width, a height (dimensions of the rectangular space) and the requested directory respectively. And you will be asked to return the treemap representation of the requested directory. In this representation, each node should contain the folder's name, the width and the height of the rectangle used for that folder, and treemaps of the subfolders. Constructing the treemap should start from the

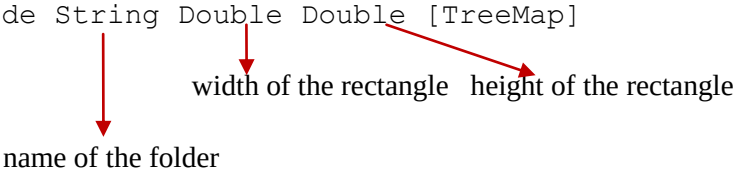
requested directory and then you should divide the rectangular area with respect to the storage sizes of the children directories and so on. **Since you're just given the storage sizes of the leaf/innermost folders, you should calculate the storage sizes of the parent nodes**. Here you can assume that, the size of a folder is equal to the sum of the sizes of its subfolders.

For creating the treemap, we will use the following tiling algorithm: If the aspect ratio is greater than or equal to 1 then the rectangular space should be divided vertically into sub-rectangles (a sub-rectangle for each subfolder) according to the size of the subfolders else it should be divided horizontally. If a treeMap node (rectangle) is divided vertically (its aspect ratio ≥ 1) then the node should be represented as `HorizontalNode`, and if a treeMap node is divided horizontally then the node should be represented as `VerticalNode`. For the leaf nodes, again if the node's aspect ratio ≥ 1 then it is a `HorizontalNode`,else it is a `VerticalNode`.

To make your job easier, the path of the directories will be given in a string list and each folder in the directory path will be an element of the list. For example; the directory path for *dir3* folder from the above figure will be given as ["home", "user1", "desktop", "dir3"].

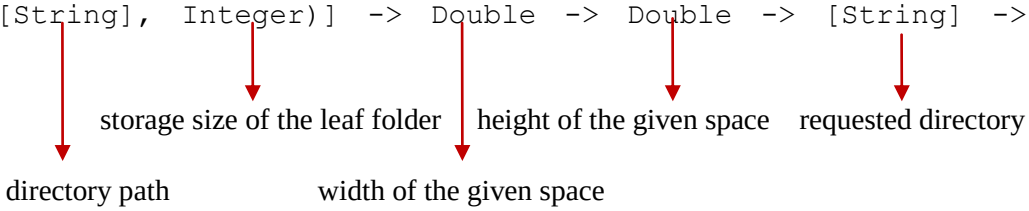
Assume that the treemap is defined as;

```
data TreeMap = VerticalNode String Double Double [TreeMap] |
               HorizontalNode String Double Double [TreeMap]
```



You will write a function called `hw2` in the form:

```
hw2 :: [([String], Integer)] -> Double -> Double -> [String] ->
TreeMap
```



Assumptions and Constraints:

- All directory paths will start with the same folder.
- At least one directory will be given.
- Aspect ratio is width / height.
- The requested directory will definitely be an existing directory.
- The dimensions of the given rectangular space and the resulting dimensions of the sub-rectangles don't have to be integers.

- There can be folders with the same name but their full paths cannot be same.
- Don’t forget that the area for each folder should be proportional to their storage size and it affects the aspect ratio of the rest.
- The order of the children treemaps isn’t important.

Examples:

```
> hw2 [([“home”, “user1”, “desktop”, “dir1”], 5), ([“home”, “user2”], 40), ([“home”, “user1”, “desktop”, “dir2”, “dir4”], 15), ([“home”, “user1”, “desktop”, “dir3”], 5), ([“home”, “user1”, “desktop”, “dir4”], 10), ([“home”, “user3”, “documents”], 21), ([“home”, “user1”, “downloads”], 21) ([“home”, “user3”, “desktop”], 3)] 30.0 16.0 [“home”]

(HorizontalNode “home” 30.0 16.0 [(VerticalNode “user1” 14.0 16.0 [(HorizontalNode “desktop” 14.0 10.0 [(VerticalNode “dir1” 2.0 10.0 []), (VerticalNode “dir2” 6.0 10.0 [(VerticalNode “dir4” 6.0 10.0 [])]), (VerticalNode “dir3” 2.0 10.0 []), (VerticalNode “dir4” 4.0 10.0 [])]), (HorizontalNode “downloads” 14.0 6.0 [])]), (VerticalNode “user2” 10.0 16.0 []), (VerticalNode “user3” 6.0 16.0 [(VerticalNode "documents" 6.0 14.0 []), (HorizontalNode "desktop" 6.0 2.0 [])])])

> hw2 [([“home”, “user1”, “desktop”, “dir1”], 5), ([“home”, “user2”], 40), ([“home”, “user1”, “desktop”, “dir2”, “dir4”], 15), ([“home”, “user1”, “desktop”, “dir3”], 5), ([“home”, “user1”, “desktop”, “dir4”], 10), ([“home”, “user3”, “documents”], 21), ([“home”, “user1”, “downloads”], 21) ([“home”, “user3”, “desktop”], 3)] 10.0 35.0 [“home”, “user1”, “desktop”]

(VerticalNode “desktop” 10.0 35.0 [(HorizontalNode “dir1” 10.0 5.0 []), (VerticalNode “dir2” 10.0 15.0 [(VerticalNode “dir4” 10.0 15.0 [])]), (HorizontalNode “dir3” 10.0 5.0 []), (HorizontalNode “dir4” 10.0 10.0 [])])

> hw2 [([“home”, “user1”, “desktop”, “dir1”], 5), ([“home”, “user2”], 40), ([“home”, “user1”, “desktop”, “dir2”, “dir4”], 15), ([“home”, “user1”, “desktop”, “dir3”], 5), ([“home”, “user1”, “desktop”, “dir4”], 10), ([“home”, “user3”, “documents”], 21), ([“home”, “user1”, “downloads”], 21) ([“home”, “user3”, “desktop”], 3)] 1 1 [“home”, “user1”, “desktop”, “dir2”]

(HorizontalNode “dir2” 1.0 1.0 [(HorizontalNode “dir4” 1.0 1.0 [])])

> hw2 [([“home”, “user1”, “desktop”, “dir1”], 5), ([“home”, “user2”], 40), ([“home”, “user1”, “desktop”, “dir2”, “dir4”], 15), ([“home”, “user1”, “desktop”, “dir3”], 5), ([“home”, “user1”, “desktop”, “dir4”], 10), ([“home”, “user3”, “documents”], 21), ([“home”, “user1”, “downloads”], 21) ([“home”, “user3”, “desktop”], 3)] 10.0 6.0 [“home”, “user2”]

(HorizontalNode “user2” 10.0 6.0 [])
```

```
> hw2 [(["home", "user1", "desktop", "dir1"], 5), (["home", "user2"], 40), (["home", "user1", "desktop", "dir2", "dir4"], 15), (["home", "user1", "desktop", "dir3"], 5), (["home", "user1", "desktop", "dir4"], 10), (["home", "user3", "documents"], 21), (["home", "user1", "downloads"], 21) (["home", "user3", "desktop"], 3)] 60.0 40.0 ["home", "user3"]

(HorizontalNode "user3" 60.0 40.0 [(HorizontalNode "documents" 52.5 40.0 []), (VerticalNode "desktop" 7.5 40 [])])
```

Specifications:

- All the work should be done individually.
- Your codes should be written in Haskell and have the name "hw2.hs". You should create a module named Hw2.
- In evaluation, black box method will be used. So be careful about the name of functions, data structures etc.
- You will submit your code through Cow system.
- You should test your codes in inek machines with hugs before submitting.