Hydraulic System

Design and implement a program to manage the simulation of an hydraulic system.
All the classes must belong to the module *`Hydraulic`*.

R1 - Elements and pipes

A hydraulic system is composed of elements (of different types) connected to each other through pipes. A system is represented by an object of class **`HSystem`** that provides the function **`addElement()`** that accepts as argument an object of class **`Element`** and adds it to an internal list.
By means of function **`getElements()`** it is possible to get an array of the elements that compose the system. The length of the array corresponds to the number of elements added to the system.

R2 - Products

The simple elements are represented by the classes **`Source`**, **`Tap`**, and **`Sink`**.
All elements have a name that can be read through the getter method **`getName()`**.
It is possible to connect the output of an element to the input of another element by means of the function **`connect()`** that accepts as argument the element whose input should be connected to the output of the subject: *`a.connect(b);`* connects the output of element *`a`* to the input of element *`b`*.
The invocation of function connect on class **`Sink`** has no effect and simply prints an error message.
Given an element, it is possible to know to which other output element it is connected by means of the function *`getOutput()`* that returns an **`Element`** object.

R3 - Complex elements

In addition to the simple elements described above, there are some complex elements. The "*`T`*" element, represented by class **`Split`**, allows splitting the input flow into two output flows. For this class, the **`connect()`** method accept an additional argument specifying which determines the output that is being connected, such parameter may assume the values *`0`* for the first output and *`1`* for the second output.
For this class, it is possible to know which elements are connected to the outputs, by means of the function **`getOutputs()`** that returns an array.

R4 - Simulation

Given a *valid* system, i.e. a tree with a source as the root and where each path ends with a sink, it is possible to compute flow and how it is split among the distinct paths.
The computation takes place into two phases: in the first phase the parameters of the elements belonging to the system are defined, in the second phase the simulation starts.
The exit flow of a **Source** can be defined through the function **setFlow()** that accepts a number representing the flow in cubic meters per hour.
Taps can be set open or closed using the function **setOpen()** that accepts a boolean argument. When a tap is open the output flow is equal to the input flow, otherwise the output is zero.
For the T split the input flow is divided into two equal output flows (each a half of the input flow).
The **simulate()** function of class *HSystem*, performs the flow computations for each element in the system starting from each source and prints, for each element, the name, the input, and the output flow. (for the source, only the output flow, and for the sink just the input flow).

R5 - Visualization

Function **layout()** of class *HSystem* returns a string containing the layout of the system using ASCII characters and spacing.
For instance, a system composed of a *Source*, connected to a *Tab*, that is connected to a *Split* whose outputs are connected to *Sinks*, would return a layout like the following one:

```
[Src]Source -> [R]Tap -> [T]Split +-> [sink A]Sink |
                                  |
                                  +-> [sink B]Sink |
```