Probleem analyse:

- Glasplaat moet steeds compleet worden afgebroken. **De Guillotine cut**
  Daarom lastiger dan bijv patronen uitsnijden in ijzer/hout etc.

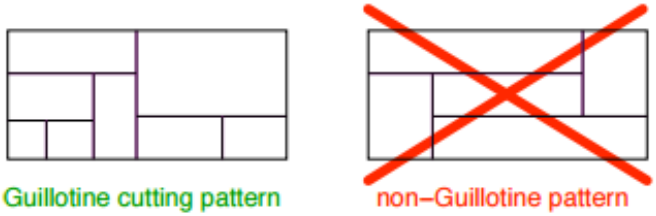


Figure 2: An example of a guillotinable layout

- De **volgorde** van snijden is dus belangrijk. **(Cutting tree)**
- Ruiten mogen 90 graden roteren -> meer opties

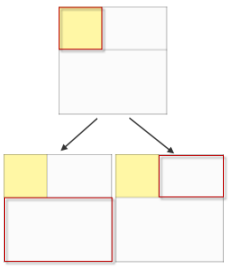
Oplossingen:

- Er is in ieder geval een oplossing mogelijk: voor *n* ruiten heb je **maximaal** *n* glasplaten nodig.
- De waarschijnlijk beste aanpak is de **The First Fit Heuristic (grootste eerst)**
  (grootste oppervlakte of grootste omtrek?)
- The 2D glass cutting problem is a particular case of a more general problem, namely the **2D Bin Packing Problem.**
- 2BP—R—G problem: the items may be rotated by 90 degree (R) and guillotine cutting is required;

<u>Pseudocode Bin Packing Problem:</u>

Sorteer de ruiten op grootte. (oppervlakte of omtrek?)

build a binary tree. Each branch in the tree contains a sprite. Each leaf node represents available space. Initially the tree has just the root node, which represents all available space. To add a sprite to the tree, search the tree for an unoccupied (leaf) node big enough to hold the sprite. Turn that node from a leaf into a branch by setting the sprite as the node's occupant and giving the node two children:



One child represents the remaining space to the right of the sprite; the other represents the remaining space below the sprite and the first child.

(zie http://codeincomplete.com/posts/bin-packing/ voor basic javascript examples)