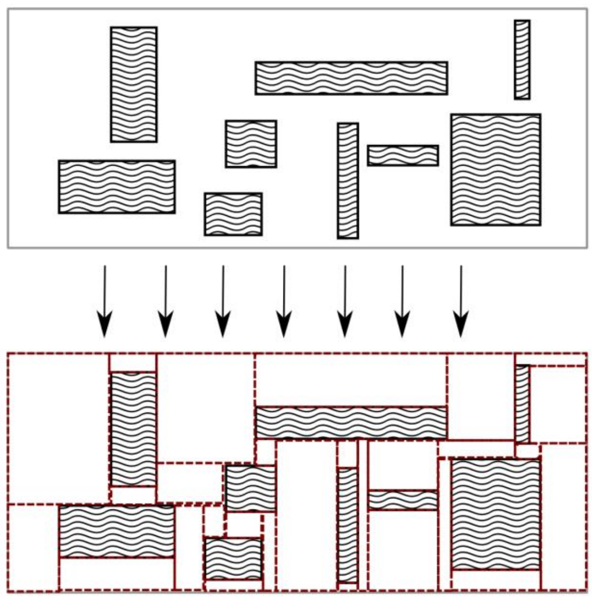
- In general, the task is to implement an algorithm which takes as input a set of non-overlapping rectangles within a larger frame, and fills the gaps with complementary, non-overlapping rectangles, such that any point inside the frame is contained within one of the rectangles. An example is shown below, with the dashed red lines marking the sides of the complementary rectangles.



- **Specifics:**

  Given the coordinates of an axis-aligned rectangular frame **R**, and **N**>=0 smaller, non-overlapping rectangles **{r_n}** (also axis-aligned) bound by **R:**

  - Return a list of complementary non-overlapping rectangles **{c_n }** such that the union of **{c_n}** and **{r_n}** will cover the entirety of **R**.
  - Write a unit test function to test your algorithm with randomly generated input:
    - Randomized **R** and **{r_n}** may have 0 area. They are to be considered lines and are legal input (in some cases the output list might have lines as well).
  - Notes:
    - All input and output rectangles are axis-aligned.
    - **{r_n}** can be empty, or already filling the whole of **R**.
    - *There is no single correct way* to fill the frame, and there are no optimization requirements (e.g., minimal number of output rectangles), or any other restrictions. So which method you opt for is up to you.

- Bonus: what changes need to be made to the function if **R** is not axis-aligned (all the smaller rectangles - **r_n** and **c_n** - are axis-aligned to **R**).