Programming Exercise 2

Exercise P.2. Implement an algorithm based on the *sequence pairs* idea for the following problem: Find a rectangular chip image of minimum area in which a disjoint placement exists for a given set of n rectangles. Rotating rectangles is not allowed.

n
$w_1 \ \ h_1$
$w_2 \ \ h_2$
$\dots$
$w_n \ \ h_n$

(a) Input file format.

$W \ \ H$
$x_1 \ \ y_1$
$x_2 \ \ y_2$
$\dots$
$x_n \ \ x_n$

(b) Output format.

Figure P.1

Your program will take one argument: the path to a file as in Figure P.1a. Each number in the input file is a positive integer that can be represented by 32-bit. The first line of the input file encodes the number of rectangles n to be placed. Each of the remaining n lines defines width w_i and height h_i of the i-th rectangle.

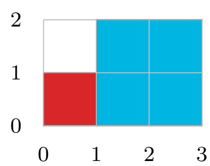
The output consists of $2n + 2$ nonnegative integers, which your program will write to the standard output in the form specified by Figure P.1b. Here W and H denote the width and height of the computed chip area $[0, W] \times [0, H]$. The remaining n lines encode the coordinates $(x_i, y_i) \in [0, W - w_i] \times [0, H - h_i]$ for the lower left corner of the i-th rectangle.

```
2
1 1
2 2
```

(a) Input file.

```
3 2
0 0
1 0
```

(b) Output.



(c) Resulting placement.

Figure P.2: Example instance with two squares with edge length 1 and 2, and one possible output.

A concrete example instance is described in Figure P.2.

The program must be written in `C` or `C++` and must compile and run on Linux. You are allowed to use any `C++` standard including `C++14`. You can use any tool available in the standard library. Your program must compile with either Clang (any version $\geq$ 3.4.2) or Gcc (version $\geq$ 4.8.3) with `-Wall -Wextra -Wpedantic -Werror` and cannot link to any external library.

Your program shall check for correct input and return with exit code 1 if this is not the case. In particular, you should detect unreadable instance files and instance files in a wrong format. The program has to achieve the theoretical running time of $\mathcal{O}((n!)^2 \cdot n^2)$. To achieve the maximum score, your program must not leak any memory and must be well documented.

(40 points)

Deadline: July 9$^{\text{th}}$, via email to ochsendorf@or.uni-bonn.de. The websites for the lecture with all exercises and test instances can be found at

`http://www.or.uni-bonn.de/lectures/ss17/chipss17.html`