

Correctness test for Task1:

program P1-1 test1 test2 test3

```
-----  ----  -----  -----  -----  
  
A4      700 20      38 6  
A5      700 20 38 6  
A6      700 20 38 6  
A10     700 19 X37 X6  
A17     700 12 X14 X6  
A21     700 20 38 6  
A25     700 20 38 6  
A26     700 20 38 6  
A27     700 20 40 X5 X  
A28     700 20 39 X8 X  
A29     700 19 X37 X6  
A30     700 18 X38 6  
A31     700 20 42 X10 X  
A32     700 20 41 X8 X  
A34     700 SEGV 38 SEGV  
A40     700 20 38 6  
A46     700 20 42 X10 X  
A55     700 20 38 6  
A62     700 20 38 6
```

* 'X' is for wrong output.

*A34 will SEGV for some special permutation of the same input data.

Performance test for Task 1:

program P1-2 P1-3

```
-----  
A4 9s 20s  
A5 3m45s 6m02s  
A6 19s 40s  
A10 11s >16hr  
A17 2m04s 32s
```

A21 9m55s 22m30s
 A25 18s 2m29s
 A26 18s 2m40s
 A27 1m33s 2m38s
 A28 21m07s 1m05s
 A29 SEGV SEGV
 A30 12s >10hr
 A31 10s >17hr
 A32 10s >12hr
 A34 18s 5m32s
 A40 9h26m 1m18s
 A46 >10hr SEGV
 A55 18m32s 16m30s
 A62 58m09s 3m43s

According to the above result, we select the following program for further evaluation:

A4 A5 A6 A21 A25 A26 A55 A62

Correctness test for Task 2

*Number of correct polygon:

program P1-1 test1 test2 test3

program	P1-1	test1	test2	test3
A4	700	20	38	6
A5	609 X	14 X	17 X	1 X
A6	700	20	38	6
A21	*699 X	20	38	6
A25	700	20	38	6
A26	626 X	18 X	33 X	5 X
A55	700	7 X	23 X	5 X
A62	91 X	8 X	23 X	5 X

*For P1-1, Other programs with correct output is A17, A27, A29, A40

*In fact, A5's result is almost correct.

Just for any sigle rectangle, its output swaps x and y coordinators.

A5 calls external binary 'sort' for sorting data. The time is not measured.

*a21 is too many splits for a perticular rectangle.

Final evaluation for A4, A5*, A6, A21*, A25:

Other performance tests: Task2 of P1-2 and P1-3, Task 1 of Test4 and Test5

program	P1-2	Memory	P1-3	Memory	Test4	Memory	Test5	Memory
A4	12s	65M	1m18s	137M	1h20m	94M	5h42m	94M
A5	3m49s	103M	6m10s	100M	6m50s	100M	7m01s	100M
A6	20s	167M	56s	187M	Out_of_Mem	>4G	Out_of_Mem	>4G
A21	10m7s	91M	22m50s	445M	26m16s	188M	26m15s	313M
A25	7m8s	76M	2m41s	103M	>22h	105M	>18h	104M

*Test4 and Test5 are randomly modified from P1-3 test case. They all have the same data size.

Data-structures and Algrithms:

A4: Quad array: A quad-list quad-tree based data-structure.

Disjoint-set datastructure.

A5: Scan-line algorithm with interval trees.

A6: Modified quad-list quad-tree.

A21: Divide-and-conquer.

Some Comments:

A4 passes all test cases and uses reasonable memory. For the offical test data, A4 is the fastest one among all. It use a data structure called Quad array, which is a variant of the quad-list quad-tree datastructure. A4 is also using the disjoint-set datastructure. The problem is well analyzed. The self testing is well done. A visual debug tool is also provided. But the algorithm still encounters some worst cases. See the final evaluation for Test4 and Test5 cases.

A5 is also good. A5 has a careless bug when reporting the coordinators of a sigle rectangle. Its output swaps x and y coordinators. Otherwise, A5 passes

all testcases. For the official test cases, A5 runs slower than A4. But for Test4 and Test5, A5 runs fastest. The running time is near P1-3's. It means A5's running time is more stable with respect to the problem size. Hence, I think A5 has much lower possibility to encounter a worst case. A5 uses a scan-line algorithm. It uses the interval tree datastructure and a hash table. A5 also provides a visualized tool.

A6 runs fast for official test cases. For Task 2 part, it seems to run faster than A4. But it uses more memory. For Test4 and Test5, it runs out of memory very quickly.

A6 uses a quad-list quad-tree datastructure as A4. It also uses an interval tree like datastructure to transform a 2D problem into a 1D one.

A21 runs slower and usually uses more memory than A5. For Task2, A21's answer is sometimes not correct. It wrongly splits a rectangle into two rectangles occasionally. Its running time is stable with respect to the problem size. It uses the Divide-and-Conquer strategy and the Red-Black tree datastructure.

A25 runs through all official test cases. But it cannot complete Test4 and Test5 within reasonable time. A25 uses a 2D dynamic array and queues.

Some programs correctly compute Task1 but failed with Task2. They are A26, A55 and A62.

A28 uses Interval trees datastructure and Disjoint-set (union-find) datastructure. It runs very fast for Task4, but the answer is not correct.

The following lists the main datastructures for each program:

A4: Quad-tree, Disjoint-set

A5: Interval tree, Hashing (scan-line)

A6: Interval tree

A10: Dynamic array

A17: Linked list (scan-line)

A21: Red-black tree (divide-and-conquer)

A25: 1D and 2D dynamic array

A26: Array

- A27: Interval tree
- A28: Interval tree (scan-line)
- A29: Interval tree
- A30: Interval tree
- A31: Linked list
- A32: Linked list?
- A34: Linked list?
- A40: Array
- A46: Linked list?
- A55: Quad-tree
- A62: Array?