Chapter 1

Readme

1.1 Tester 1 (test1.exe)

Usage:
Click to run.
Input:
Input file name of exe, for example "algorithm1". Then input N.
Output:
Tester will check your output and answer automatically, if a mistake is found, program will pause, then *in.txt* is the input, *out.txt* is your output, and *ans.txt* is answer.
Generator: gen1

1.2 Tester 2 (test2.exe)

Usage:
Click to run.
Input:
Input two file name of exe, for example "algorithm1" and "algorithm2". Then input N.
Output:
Tester will check two outputs automatically, if difference is found, program will pause, then *in.txt* is the input, *out1.txt* is your first output, and *out2.txt* is your second output.
Generator: gen2

1.3 Tester 3 (test3.exe)

The same as test2, use gen3.

1.4 Random Generator 1 (gen1.exe)

Usage:
`gen1 [N] [random seed]`
Input:
None
Output:
1.Out put a set of random input for this problem to stdout.
2.Out put the answer to ans.txt.
Generating method:
Generate 4/5N-N random integers, sort them, and input into a hash table.
Note:
Use this method to generate data, the answer is always in increasing order.

1.5 Random Generator 2 (gen2.exe)

Usage:
`gen2 [N] [random seed]`
Input:
None
Output:
Out put a set of random input for this problem to stdout.
Generating method:
Generate 0-N random integers, and input into a hash table.
Note:
1. None answer is generated, we use this gerator to compare whether two algorithms output the same way.
2. Empty block is always -1.
3. 0 is possible.

1.6 Random Generator 3 (gen3.exe)

Usage:
`gen3 [N] [random seed]`
Input:
None
Output:
Out put a set of random input for this problem to stdout.
Generating method:
Generate 0-N random integers, and input into a hash table.
Note:
1. None answer is generated, we use this generator to compare whether two algorithms output the same way.

2. Empty block is NOT always -1.
3. 0 is possible.