

Нарастващи подредици

Иванчо има пред себе си пермутация **P** на числата от 1 до **N**. Числата са подредени от ляво надясно. Дълго време той се чудеше какво да прави с тази пермутация и накрая реши да взема различни нарастващи подредици от нея. Всяка подредица се определя от последователност от индекси

$$i_1 < i_2 < \dots < i_k,$$

за които трябва да е изпълнено

$$p[i_1] < p[i_2] < \dots < p[i_k].$$

Иванчо иска да намери дължината на най-дългата възможна нарастваща подредица, която може да вземе от пермутацията, както и дължината на най-дългата нарастваща подредица, за която е изпълнено, че последното число е точно фиксирано **A**.

Вход: На първия ред от входния файл **maximal.in** се въвежда естественото число **N** - дължината на пермутацията **P**, както и числото **A**, на което искаме да завършва втората търсена от нас най-дълга нарастваща подредица. На втория ред от входния файл са въведени **N** числа от 1 до **N**, всяко се среща по веднъж - началната пермутация **P**.

Изход: На единствения ред от изходния файл **maximal.out** изведете две числа. Първото е дължината на най-дългата нарастваща подредица в пермутацията. Второто - дължината на най-дългата нарастваща подредица, която завършва с числото **A**.

Ограничения:

$$1 \leq N \leq 5\,000$$

$$1 \leq A \leq N$$

Ограничение за време: 1 секунда

Ограничение за памет: 256 MB

Пример:

maximal.in	maximal.out
5 3 1 5 3 2 4	3 2

Обяснение на примера:

Най-дългата нарастваща подредица а например 1 2 4. Дължината ѝ е 3.

Най-дългата нарастваща подредица, която завършва на числото 3 е 1 3 с дължина 2.