Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2283 - Diseño y Análisis de Algoritmos - 2do semestre 2022

TAREA 2

Publicación:	Lunes 12 de septiembre.
GitHub Classroom:	https://classroom.github.com/a/K8lax8KS
Entrega:	**Viernes 23 de septiembre 23:59 horas**.

Indicaciones

- La tarea es estrictamente individual.
- La solución debe ser entregada en el archivo `t2.py` del repositorio privado asignado mediante GitHub Classroom para esta tarea. Se revisará el último *commit* subido antes de la entrega al repositorio. Se usará Python 3.10.X para la revisión.
- El *input* para el programa debe ser obtenido desde *standard input*. El *output* debe ser entregado mediante *standard output*.
- La corrección se realizará mediante *tests* automatizados acordes al formato de *input* y *output* especificado. Cada *test* tendrá un *timeout* según lo que se especifica como tiempo esperado.
- Un *test* se considerará **reprobado** en caso de que 1) dado el *input* el *output* sea incorrecto, 2) exista un error de *runtime* durante su ejecución, o 3) el *timeout* se cumpla durante su ejecución. En otro caso, el *test* se considerará **aprobado**.
- No se permite el uso de librerías externas a la librería estándar de Python *a priori*. Consultar en las *issues* del repositorio oficial del curso en caso de requerir una.
- **Para esta tarea sí aplica la política de atrasos descrita en el programa del curso.**

Problema

Sea A un arreglo de enteros $a_1, a_2, ..., a_n$. En este problema se puede realizar sólo la siguiente operación sobre el arreglo A cuántas veces sea posible:

Operación:

1. Elegir un par de dos elementos iguales vecinos $a_i = a_{i+1}$ (si hay al menos un par de este tipo).
2. Sustituirlos por un elemento con valor $a_i + 1$.

Después de cada operación de este tipo, la longitud del arreglo disminuirá en uno (y los elementos se renumeran en consecuencia). Por ejemplo se puede obtener el arreglo $4, 3, 6, 5, 6, 8$ a partir del arreglo $4, 6, 4, 4, 5, 5, 7, 7$ aplicando las siguientes operaciones

$$4, 6, \mathbf{4}, \mathbf{4}, 5, 5, 7, 7 \rightarrow 4, 6, 5, \mathbf{5}, \mathbf{5}, 7, 7 \rightarrow 4, 6, 5, 6, \mathbf{7}, \mathbf{7} \rightarrow 4, 6, 5, 6, 8$$

Sin embargo se puede obtener un arreglo más pequeño aplicando una lista distinta de operaciones

$$4, 6, \mathbf{4}, \mathbf{4}, 5, 5, 7, 7 \rightarrow 4, 6, \mathbf{5}, \mathbf{5}, 5, 7, 7 \rightarrow 4, \mathbf{6}, \mathbf{6}, 5, 7, 7 \rightarrow 4, 7, 5, \mathbf{7}, \mathbf{7} \rightarrow 4, 7, 5, 8$$

La pregunta a resolver es: ¿Cuál es la mínima longitud posible del arreglo A que se puede obtener luego de aplicar la operación varias veces?

Input

La primera línea contiene sólo un entero n ($1 \leq n \leq 500$) correspondiente a la longitud inicial del arreglo A.

La segunda línea contiene n enteros $a_1, a_2, ..., a_n$ ($1 \leq a_i \leq 1000$) que corresponden a las entradas del arreglo inicial A.

Output

Imprime un único entero m que corresponde a la longitud mínima posible que puede obtener después de realizar la operación descrita anteriormente cualquier número de veces.

Tiempo esperado

Se espera que la solución se ejecute en un tiempo **menor o igual a 3 segundos** para cualquier instancia de input según las restricciones dadas.

Complejidad esperada

Se espera que la solución posea una complejidad de $\mathcal{O}\left(n^3\right)$.

Hint: Intente separar el problema en dos partes y utilice programación dinámica para resolver.

Ejemplo

Los siguientes tests están ya cargados a GitHub Classroom con corrección automática mediante GitHub Actions. **Los tests para la corrección serán distintos a estos.**

Input
5 4 3 2 2 3
Output
2

Input
7 3 3 4 4 4 3 3
Output
2

Input
3 1 3 5
Output
3

Input
1 1000
Output
1

Nota: Para el primer ejemplo, una de las posibles secuencias óptimas es

$$4\ 3\ 2\ 2\ 3 \to 4\ 3\ 3\ 3\ 3 \to 4\ 4\ 3 \to 5\ 3$$

Para el segundo ejemplo, una de las posibles secuencias óptimas es

$$3\ 3\ 4\ 4\ 4\ 3\ 3 \to 4\ 4\ 4\ 4\ 3\ 3 \to 4\ 4\ 4\ 4\ \to 5\ 4\ 4\ 4 \to 5\ 5\ 4 \to 6\ 4$$

Para el tercer y cuarto ejemplo no es posible realizar la operación en ninguna posición.