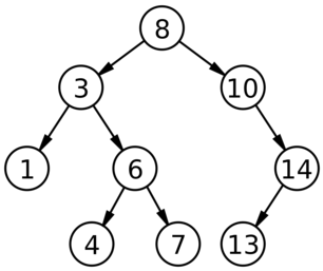
E. Navegação em Árvores Binárias de Busca

time limit per test: 1 second
memory limit per test: 256 megabytes

Árvore Binária de Busca (ABB ou BST) é uma estrutura baseada em nós, onde todos os nós da subárvore à esquerda possuem um valor numérico inferior ao nó raiz e todos os nós da subárvore à direita possuem um valor superior ao nó raiz (e assim sucessivamente).

Como a organização de uma BST depende da ordem de inserção dos elementos, é possível que uma mesma sequência de elementos possa gerar diferentes BSTs. Por exemplo, ao inserir em uma BST os elementos $8, 3, 1, 6, 10, 14, 13, 4, 7$ nessa ordem, obtemos a seguinte árvore:



Dada a estrutura de uma árvore, percorrer os seus elementos tem diversas aplicações. Um exemplo é que o percurso em ordem (infixo) resulta em uma sequência ordenada dos elementos.

Nesta atividade, você receberá a ordem de inserção dos elementos em uma BST e deverá construir uma árvore binária de busca e imprimir os elementos nas ordens infixa, prefixa e posfixa.

Input

A entrada contém um único caso de teste. A primeira linha contém um inteiro N ($1 \leq N \leq 500$) que indica a quantidade de nós que deve compor a árvore e a segunda linha contém N inteiros distintos e não negativos V ($0 \leq V \leq 10^9$), separados por um espaço em branco, que são os valores dos nós na ordem em que foram inseridos na árvore.

Output

A saída deve conter três linhas. A primeira linha deve conter os valores dos nós da árvore em ordem infixa, a segunda linha deve conter os valores dos nós da árvore em ordem prefixa e a terceira linha deve conter os valores dos nós da árvore em ordem posfixa, conforme os exemplos.

Examples

input
9 8 3 1 6 10 14 13 4 7
output
In.: 1 3 4 6 7 8 10 13 14 Pre: 8 3 1 6 4 7 10 14 13 Pos: 1 4 7 6 3 13 14 10 8

input
5 1 2 3 4 5
output
In.: 1 2 3 4 5 Pre: 1 2 3 4 5 Pos: 5 4 3 2 1

input
5 5 4 3 2 1

output Copy Copy Copy Copy Copy Copy

```
In.: 1 2 3 4 5
Pre: 5 4 3 2 1
Pos: 1 2 3 4 5
```

input Copy

```
5
1 2 5 4 3
```

output Copy

```
In.: 1 2 3 4 5
Pre: 1 2 5 4 3
Pos: 3 4 5 2 1
```

Note

Este exercício provavelmente demandará a implementação da BST. Recomenda-se utilizar a linguagem que esteja mais confortável para a implementação.

IDP - TAA - 2025/02

Private

Participant

→ About Group

Este grupo tem o objetivo de organizar as atividades de programação da disciplina de Técnicas de Programação e Análise de Algoritmos.

Group website

→ Group Contests

- TAA - LEA 05
- TAA - LEE 05
- TAA - LEA 04
- TAA - LEE 04
- TAA - AS 01
- TAA - LEA 03
- TAA - LEE 03
- TAA - LEA 02
- TAA - LEE 02
- TAA - LEA 01
- TAA - LEE 01
- ET - Exercício de Testes

TAA - LEE 04

Finished

Contestant

→ Last submissions

Submission	Time	Verdict
344765364	Oct/19/2025 21:55	Accepted

Codeforces (c) Copyright 2010-2025 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: Nov/11/2025 23:46:24 UTC-3 (k2).
Desktop version, switch to mobile version.
Privacy Policy | Terms and Conditions

Supported by