/ 20 P

Programmieraufgabe P2.

Passwort für Einschreibung: `asymptotic`

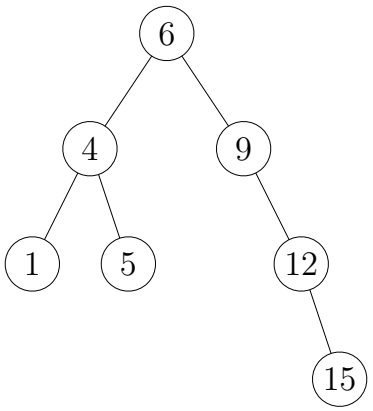
Einreichung: `https://judge.inf.ethz.ch/team/websubmit.php?cid=28784&problem=BSTReord`

BST Reordering

Consider a Binary Search Tree T whose vertices store distinct integer keys in $\{1, \ldots, 10000\}$. You are given the height $h \leq 500$ of T (i.e., the number of edges on the longest path between the root of T and a leaf) and the list of the stored keys, as visited by a *preoder traversal* of T. Your task is to output the stored keys as visited by a *Breadth-First traversal* of T.

Both traversals prefer left over right children when selecting the next vertex to visit, namely if u is a vertex having v as its left child and w as its right child, then v must be visited before w.

Beispiel



Consider the above Binary Search Tree T. A preorder traversal of T visits the (vertices corresponding to the) stored keys in the following order: 6, 4, 1, 5, 9, 12, 15. A Breadth-First traversal of T visits the stored keys in the following order: 6, 4, 9, 1, 5, 12, 15.

Anforderung

Overall, you can obtain a maximum of 20 judge points for this programming task. Any correct solution requiring $O(n^2)$ time (with reasonable hidden constants) can obtain full points. Partial points can be achieved for the following subtasks:

Subtask 1: You can obtain up to 8 points by correctly solving instances in which the tree T is a *complete* binary tree of height $h \leq 10$.

Subtask 2: You can obtain up to 8 additional points by correctly solving instances in which the tree T has height $h \leq 10$. Here T is not necessarily complete.

One possible solution for the general task consists of two steps. The first step reconstructs the Binary Search Tree T from the preorder traversal using the provided `Vertex` class (see the Instruktionen section below). The second step consists of a Breadth-First visit of T.

Instruktionen For this exercise, we provide a program template as an Eclipse project in your workspace. The template already contains the code needed to read the input and write the output.

Moreover, the template also contains a `Vertex` class, which you might find useful to implement your solution. While no changes to the `Vertex` are needed to correctly solve the task, you are still free to modify, extend, remove, or ignore it altogether.

Finally, the project contains data for your local testing and a `Judge.java` program that runs your `Main.java` on all the local tests – just open and run `Judge.java` in the project. The local test data are different from the data that are used in the online judge.

Submit only your `Main.java`.

Die Ein- und Ausgabe werden von der Vorlage verarbeitet – Sie sollten den Rest dieses Texts nicht benötigen.

Eingabe Die erste Zeile der Eingabe enthält einzig die Anzahl der Tests.

Each test case consists of two lines: The first line contains two integer n and h: the number of vertices of T and the height of T, respectively; The second line is a list of the keys stored in the vertices T, as visited by a preorder traversal. Keys are separated by a space.

Ausgabe For each test case, output a single line containing the keys stored in the vertices of T, as visited by a Breadth-First traversal of T. Keys must be separated by a single space.

Beispiel-Eingabe (corresponding to the previous example):

```
1
7 3
6 4 1 5 9 12 15
```

Beispiel-Ausgabe:

```
6 4 9 1 5 12 15
```

Platz für Ihre Notizen. Diese werden nicht bewertet. Nur was auf dem Judge eingereicht wird zählt für diese Aufgabe.