CS29003 ALGORITHMS LABORATORY
Assignment No: 1
Last Date of Submission: 31-July-2014

In this assignment, you work with binary trees. We assume that each node in the tree contains only a positive integer value and two child pointers (left and right). No parent pointers or additional values can be stored in the nodes.

Part I

You are given an array *A* with *n* positive integer values. Your task is to convert the array to a binary tree *T* storing the values in *A*. The first element of *A* (that is, A[0]) is stored at the root. Now, we make a random choice. With probability 1/3, the left subtree of *T* is empty, so the right subtree contains all of the remaining *n* – 1 values. With probability 1/3, the right subtree is empty, that is, the left subtree contains the remaining *n* – 1 values. Finally, with probability 1/3, both the subtrees are non-empty. In this case, each subtree consists of a half of the remaining *n* – 1 values: the left subtree the first half, and the right subtree the second half. Recursively, construct the left and/or right subtrees, whichever is/are not empty. You may read the array *A* from a file, or generate it randomly.

Which traversal of the created tree *T* produces the same listing of the values as stored in the original array *A*? Implement that traversal in order to check the correctness of your construction of *T*.

Part II

Let *r* be the root of the tree, and *v* any node in the tree *T*. There is a unique path from *r* to *v* in *T*. The *weight* of *v* is defined as the sum of all the values stored on this unique *r*-to-*v* path. Your task is to locate the maximum of the weights of all the nodes in *T*. Since the values stored in the nodes are positive, a maximum-weight node must be a leaf node. Write a *linear-time* recursive function to compute the largest weight. Do not use any global or static variables. Pass a parameter to store the sum of weights of all nodes on the path from the root to the current node (or its parent). During recursive calls, increment this parameter appropriately. Alternatively, you may use the fact that the maximum weight of a tree is the value stored at the root plus the larger of the maximum weights of the two subtrees of the root.

Part III

Let *w* be the maximum weight returned by the function of Part II. Write a recursive function to locate a path from the root to a leaf such that the sum of the values encountered on the path is *w*. If you have parent pointers, then such a path can be located in linear time. Likewise, if you store some indicator (like which subtree has the largest-weight leaf), then also it is easy to locate the path in linear time. But in your case, you do not have these options. If the function of Part II reorganizes the tree links so as to let the left subtree at every node contain the largest-weight leaf under it, then the maximum-weight path is the leftmost path in the restructured tree. But this destroys the original tree. Do not do that too.

Nevertheless, you can manage to locate a maximum-weight path by an expected linear-time (but worst-case quadratic-time) algorithm. Write a function of this complexity.

Sample Output

```
+++ INPUT ARRAY
 941 999 153 295 461 734 175 738 235 405 966 151 815 391 504  12 537 688 985 390
 139 981 257 693 338  78 715 269 643 598 569 584 597 721 598 776 455 772 514 408
 177 199 278 991 590 781 722 127 188 426 235 327 408 491 738 464 288 173 732 930
 770  20 233  86 740 830 861 913 322  95  41 217 293 318 209 882  99 649 727 287
  75 680 332 482 890  70 664 178 242 396 826  12 134  59 815 592 608 677 225 929
 771 265 146 782 582  73 665 680 721 111 685 796 791  17 996 681 805 661 858  47
 775 403 777 628 461 592 220 787
+++ TREE TRAVERSAL LISTING
 941 999 153 295 461 734 175 738 235 405 966 151 815 391 504  12 537 688 985 390
 139 981 257 693 338  78 715 269 643 598 569 584 597 721 598 776 455 772 514 408
 177 199 278 991 590 781 722 127 188 426 235 327 408 491 738 464 288 173 732 930
 770  20 233  86 740 830 861 913 322  95  41 217 293 318 209 882  99 649 727 287
  75 680 332 482 890  70 664 178 242 396 826  12 134  59 815 592 608 677 225 929
 771 265 146 782 582  73 665 680 721 111 685 796 791  17 996 681 805 661 858  47
 775 403 777 628 461 592 220 787
+++ MAXIMUM WEIGHT = 10280
+++ VALUES ON THE MAX-WEIGHT PATH
941 + 999 + 830 + 861 + 677 + 225 + 929 + 771 + 265 + 146 + 782 + 582 + 685 + 796 + 791
```

Submit a single C/C++ file.