Computing Systems Lab (IT69101)

(Autumn 2010)

Assignment #3 **4/08/2010**

1. Consider three sets of 20 or more integer data as follows.

i. In ascending order
ii. In random order
iii. Almost in order except few (say 25%) are out of order

Input (common to all subparts of problem 1)

The input consists of a number of test cases (program execution runs) each separated by a blank line. For each test case three file names will be given – each on a separate line. Open the files to read the list of numbers. Each file consists of a space-separated list of numbers.

Ascend – contains a list of numbers in ascending order (subpart i).

Rand - contains a list of numbers in rand order (subpart ii).

Scatter - contains a list of numbers that are about 25% out of order (subpart iii).

The test cases are separated by a blank line.

<number of test cases, N>
<ascend1.txt>
<rand1.txt>
<scatter1.txt>
<ascend2.txt>
<rand2.txt>
<scatter2.txt>
...
...
...
<ascendN.txt>
<randN.txt>
<scatterN.txt>

Sample Input

2
ascend1.txt
rnd.txt
sta.txt
asc.txt
rn.txt
scattered.txt

(a) Define a C function to create binary search trees for each of the above. Your function should produce a output of a tree (Use GraphViz software (open source from AT&T at http://www.graphviz.org/ to display the trees).

Output

The output consists of the *.png and *.dot files. It is required to name the files with the number of the test case number it corresponds to. It is also required to provide the appropriate extensions.

Eg: 1i.png and 1i.dot for the ascending order of the first test case.

Sample Output

// 1i.png and 1i.dot

// 1ii.png and 1ii.dot

// 1iii.png and 1iii.dot

// 2i.png and 2i.dot

// 2ii.png and 2ii.dot

// 2iii.png and 2iii.dot

(Note that this is not to be outputted to the standard output. Rather this is to indicate the file naming convention. We will only looking at the file contents.)

(b) For the binary search trees in the above three cases, obtain the following tree traversals (write a function for each traversal and then call the function).

i. Pre-order tree traversal

ii. In-order tree traversal

iii. Post-order tree traversal

Output

Output the space-separated list of numbers in separate lines with the appropriate prefix. Give a blank line between each test case.

pre<Test case number>: <space-separated list in pre order>

in<Test case number>: <space-separated list in pre order>

post<Test case number>: <space-separated list in pre order>

Sample Output

pre1: 5 2 8 7 9 11

in1: 2 5 7 8 9 11

post1: 2 7 11 9 8 5

pre2: 21 12 5 26 30 36 40

in2: 5 12 21 26 30 36 40

post2: 5 12 40 36 30 26 21

(c) Write C functions for each of the following. Consider the trees as obtained in 1(a).

i. To count the average values of all nodes in the input tree.

ii. To find the minimum value stored in the tree.

iii. To find the maximum value stored in the tree.

Output

For each test case output the average value, the minimum value and the maximum value each on a new line (in the order: average, minimum, maximum). Each test case must be separated by a blank line.

Sample Output

2. Take two inputs as In-order and Pre-order traversals of a binary tree. Write C implementation to obtain the following.
i. Show the binary tree
ii. Obtain the Post-order traversal

Input

<number of testcases, N>
<number of nodes for TC1>
<inorder>
<preorder>

<number of nodes for TC2>
<inorder>
<preorder>
...
...
...
<number of nodes for TCN>
<inorder>
<preorder>

Sample Input

2
6
2 5 7 8 9 11
5 2 8 7 9 11
7
5 12 21 26 30 36 40
21 12 5 26 30 36 40

Output

For each test case, create two files (*.png, and *.dot) and output the post order to the standard output. The file name for each test case is the test case number with the appropriate extension.

Sample Output

// create 1.png and 1.dot (do not print this line onto the standard output)
2 7 11 9 8 5
// create 2.png and 2.dot (do not print this line onto the standard output)
5 12 40 36 30 26 21

3. Given a text in English (read any ASCII file), count the frequency of different characters in it (you may exclude all numeric characters). Write C programs for the following.

Input (common to all subparts of problem 3)

For each test case a file name is provided. Read the file to obtain the string of characters. While doing so -

1. Do NOT count numbers.

2. Make sure to consider special characters (*, %, etc.)

<Number of test cases, N>

<file name for testcase 1>

<file name for testcase 2>

...

...

...

<file name for testcase N>

Sample Input

3

aaa.txt

aab.txt

aac.txt

(a) Taking the frequencies of the characters draw a Huffman tree (and show the tree with values in all internal and external nodes).

Output

Generate the *.png and *.dot files using the test case number as the filename.

Sample Output:

1.png and 1.dot

2.png and 2.dot

3.png and 3.dot

(Note that this is not to be outputted to the standard output. Rather this is to indicate the file naming convention. We will only looking at the file contents.)

(b) Obtain the Huffman coding for all the characters in the text.

Output

For each test case generate a list of (character, code) tuples having the following format.

<character-1>\t<code>

<character-2>\t<code>

<character-3>\t<code>

...

...

<character-n>\t<code>

The tuples may be listed in any order. Characters not specified in the input need not be mentioned in the output. Each test case is to be separated by a blank line.

Sample Output

a 23

e 48

* 2

s 12

w 34

/ 4

a 54

3

b 56

u 23

(c) Average code length

Output

For each testcase output only the average code length on to the standard output. Each test case must be outputted on a new line.

Sample Output

3

6

8

(d) Total bytes to be transmitted for the given input text

Output

For each testcase output only the total bytes to be transmitted on to the standard output. Each test case must be outputted on a new line.

Sample Output

3

6

8

(e) Percentage of savings in the encryption with variable length coding with respect to the fixed length coding.

Output

For each testcase output only the percentage of savings on to the standard output. Each test case must be outputted on a new line.

Sample Output

3

6

8

Last date of submission: 18/08/2010