MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG213 Data Structures – Programming Assignment 4

Date handed out: Thursday 16 May 2019
Date submission due: Friday 31 May 2019

"Indexing Websites!"
This assignment aims to help you practice binary search tree ADT and AVL Tree and make a comparison between both data structures through complexity analysis. You will write a program that creates an index of a given list of the top popular websites and their IP addresses.

Requirements:
In this assignment, you are given a text file that includes the list of the top popular websites and their IP addresses. When you typically request a website from your browser by typing in a URL[1], that URL is converted to an IP address[2] that is used to retrieve that page from the given URL.

In this assignment, you will process this external file with URLs and IP addresses, and then you will create an index of those pages. The attached file includes the list of IP addresses as follows:

```
google.com > 216.58.219.206
youtube.com > 172.217.4.206
facebook.com > 31.13.69.228
baidu.com > 123.125.114.144
yahoo.com > 206.190.36.45
amazon.com > 54.239.17.6
wikipedia.org > 208.80.154.224
qq.com > 61.135.157.156
google.co.in > 216.58.219.195
```

As you can see here, the first page is the main Google page and the last page is the Google page in Italy (because of the extension "co.in" in the IP address).

Your task here is to process this file, and generate an AVL tree and a BST that indexes the list of pages and their IP addresses. However, in this file, when you process the page URLs, you need to consider that google.com and google.co.in are both belong to Google. For this, you need to just retrieve the name of the page which is the one until the first dot in the URL. For example, once you insert the 9 entries given above you should have the following AVL tree.

[1] Uniform Resource Locator (URL), https://docs.oracle.com/javase/tutorial/networking/urls/definition.html

[2] What is an IP address? http://computer.howstuffworks.com/internet/basics/question549.htm

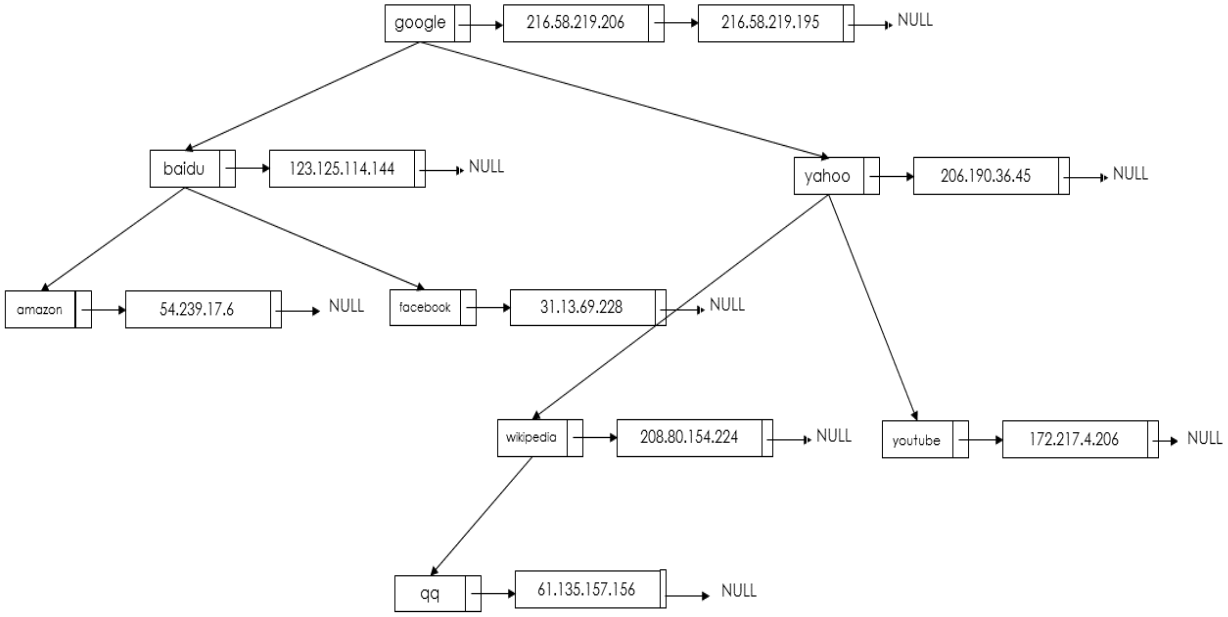

Figure 1 Data Structure Representation

You need to write a program that provides the following functionalities to the user:

1. Display the full index: This will display the full AVL tree constructed and Full BST. For example, for the first 9 entries given above it will show the following.

```
amazon, IP addresses: 54.239.17.6
baidu, IP addresses: 123.125.114.144
facebook, IP addresses: 31.13.69.228
google, IP addresses: 216.58.219.206, 216.58.219.195
qq, IP addresses: 61.135.157.156
wikipedia, IP addresses: 208.80.154.224
yahoo, IP addresses: 206.190.36.45
youtube, IP addresses: 172.217.4.206
```

2. Search for a URL: This will allow the user to enter a page and then retrieve the list of all IPs for that page. For example, for the first 9 entries given above, it will say:

```
Please enter the page: google
IP addresses for google: 216.58.219.206, 216.58.219.195
```

3. Search for an IP address: This will allow the user to enter an IP address and find the page that it belongs to:

```
Please enter the URL:  216.58.219.
IP has to be in this format xxx.xxx.xxx.xxx
Please enter the URL:  216.58.219.206
It belongs to google
```

Programming Requirements:

You will start this programming by taking the file name as a command line argument and then you will need to implement at least the following functions:

- `read_ip_data`: This function will mainly process the external file. As an input, it will take the file name and it will return an AVL tree and simple BST.
- `insert_page`: This function will take an AVL tree and a simple BST, a page, and its IP address, and then it will try to insert it to the AVL tree and BST seperately. If the page is already in the tree, then it will update the existing node with the IP. You cannot again make assumptions about the number of IP address here (i.e., you do not know how many IP addresses a website has), so make sure that you use a dynamic list (you are encouraged to use a linked list here). If the page is not in the AVL tree and BST, then the functions will create a new node and add it to the tree.
- `display_index`: This function will mainly take an AVL tree and BST and display the index of the pages in an alphabetical order.
- `search_url`: This function will mainly take an AVL tree , BST and a page, and will return the list of IPs of that page.
- `search_ip`: This function will mainly take an AVL tree and an IP and BST and an IP, and will return the page with that IP.
- `Comparison`: This function will mainly take an AVL tree and BST and perform a comparison of run time of both trees.

Please note that in this assignment, you can make use of the functions in the **string.h** library and similar external libraries. Please also note that your solution should be able to take any file in the **given format** and can create the index. You cannot assume about the number of pages and IPs given in this external file.

Submission Requirements:

Create a project under the **CNG213_Assignment_4** folder. Separate the major functionality of your Abstract Data Types into header files and put them under the **CNG213_Assignment_4/Project_Lib** folder. Project submission should be compressed version of **CNG213_Assignment_4** folder. If you do not follow this structure, you will lose %10 from the overall grade.

Programming Style Tips!

Please follow the modular programming approach. In C programming, we use functions referred to modules to perform specific tasks that are determined/guided by the solution. Remember the following tips!

- Modules can be written and tested separately!
- Modules can be reused!
- Large projects can be developed in parallel by using modules!

- Modules can reduce the length of the program!
- Modules can also make your code more readable!

Sample run:
You can find the sample run of the program for the first 9 entries given on the first page.

```
"Indexing Websites!"
Your file has been loaded, the index has been created!
--------   MENU    --------
   1. Display the full index in AVL tree
   2. Display the full index in BST tree
   3. Search for a URL From AVL tree
   4. Search for a URL From BST tree
   5. Search for an IP address From AVL tree
   6. Search for an IP address From BST tree
   7. Comparison of Run time
   8.  Exit
---------------------------
Option: 1,2

amazon, IP addresses: 54.239.17.6
baidu, IP addresses: 123.125.114.144
facebook, IP addresses: 31.13.69.228
google, IP addresses: 216.58.219.206, 216.58.219.195
qq, IP addresses: 61.135.157.156
wikipedia, IP addresses: 208.80.154.224
yahoo, IP addresses: 206.190.36.45
youtube, IP addresses: 172.217.4.206

--------   MENU    --------
   1. Display the full index in AVL tree
   2. Display the full index in BST tree
   3. Search for a URL From AVL tree
   4. Search for a URL From BST tree
   5. Search for an IP address From AVL tree
   6. Search for an IP address From BST tree
   7. Comparison of Run time
   8.  Exit
---------------------------
Option: 3, 4

Please enter the page: google
IP addresses for google: 216.58.219.206, 216.58.219.195

--------   MENU    --------
   1. Display the full index in AVL tree
   2. Display the full index in BST tree
   3. Search for a URL From AVL tree
```

```
   4. Search for a URL From BST tree
   5. Search for an IP address From AVL tree
   6. Search for an IP address From BST tree
   7. Comparison of Run time
   8.  Exit
--------------------------
Option: 5, 6

Please enter the URL:  216.58.219.
IP has to be in this format xxx.xxx.xxx.xxx
Please enter the URL:  216.58.219.206
It belongs to google

--------   MENU    --------
   1. Display the full index in AVL tree
   2. Display the full index in BST tree
   3. Search for a URL From AVL tree
   4. Search for a URL From BST tree
   5. Search for an IP address From AVL tree
   6. Search for an IP address From BST tree
   7. Comparison of Run time
   8.  Exit
--------------------------
Option: 7

Run time of AVL tree readıp function:
Run time of AVL tree search URL:
Run time of BST readıp function:
Run time of BST tree search URL:
--------   MENU    --------
   1. Display the full index in AVL tree
   2. Display the full index in BST tree
   3. Search for a URL From AVL tree
   4. Search for a URL From BST tree
   5. Search for an IP address From AVL tree
   6. Search for an IP address From BST tree
   7. Comparison of Run time
   8.  Exit
--------------------------
Option: 8

GoodBye!
```

Grading

Your program will be graded as follows:

Grading Point	**Mark (100)**
BST Tree Data Structure	5 pts
Processing page/IP data txt file (`read_ip_data`) to BST	5 pts
Inserting/Updating a node in the tree `(insert_page) in BST`	20 pts
`search_url from BST`	5 pts
`search_ip from BST`	5 pts
`display_index from BST`	5 pts
AVL Tree Data Structure	5 pts
Processing page/IP data txt file (`read_ip_data`) to AVL	5 pts
Inserting/Updating a node in the tree `(insert_page) in AVL`	20 pts
`search_url from AVL`	5 pts
`search_ip from AVL`	5 pts
`display_index from AVL`	5 pts
`main`	10 pts

NOTE: Remember to have good programming style (Appropriate comments, variable names, formulation of selection statements and loops, reusability, extensibility etc.). Each of the items above will include10% for good programming style. Good programming style also includes modularity approach explained above.