MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG213 Data Structures – Programming Assignment 1

Date handed out: **Monday 24 October 2016**
Date submission due: **Monday 7 November 2016**

"A-Eye: Analysing Eye tracking Data"

This assignment aims to help you practice the linked list data structure by using its basic and advance operations. The overall goal here is to create a C program that takes in data collected by an eye tracker and allows the analysis and interpretation of the data.

Eye tracking is a way of recording eye movements of users by using infrared signals. Eye trackers typically generate quite a lot of data that needs to be analysed and interpreted. In this assignment, your task is to implement a C program that can be used to analyse and interpret eye tracking data collected with Tobii eye tracker[1].

Problem Definition:

In this program, you will need to first import raw eye tracking data which is in the format as shown in Table 1 below.

Fixation Index	Timestamp	Fixation Duration	Fixation Point X	Fixation Point Y	Stimuli Name
…	…	…	…	…	…
2	318	817	536	450	ScreenRec
3	1135	140	538	346	www.bbc.co.uk
4	1274	498	669	189	www.bbc.co.uk
5	1773	160	487	164	www.bbc.co.uk
6	2092	100	473	182	www.bbc.co.uk
…	…	…	…	…	…
99	30861	379	621	654	www.godaddy.com

Table 1 Sample Eye tracking Data

Table 1 shows the recording of a user at a particular session, specifically a series of points where the user looked at. These points are referred to as fixations. Each line of the table shows a fixation made by the user. For each fixation, the following details are provided: its index entry number which is automatically assigned by the eye tracker, its timestamp which is the time of the recording, its duration in milliseconds (i.e., how long the user looked at the point), and its X and Y coordinates (Fixation Point X and Fixation Point Y) and the stimulus name (in our case, the web page address) that the user looked at. Please note that one user can visit more than one page in one session. When the stimuli name of a fixation is "ScreenRec" or "No Media", the fixation should be directly ignored as it does not belong to any web page used.

[1] http://www.tobii.com/group/about/this-is-eye-tracking/

You will need to read raw data and represent it in your program as follows. You should create a linked list to store the pages, and for each page, you need to create a linked list where each node stores information about a particular fixation (i.e., about each entry in the given Table 1).

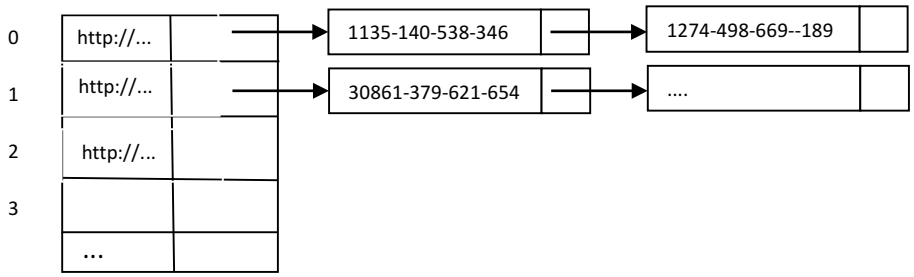

Figure 1 Data Structure Representation

Once you read raw eye tracking data from a file, you will need to implement some basic operations of linked lists, such as the insertion and deletion of nodes. Once you create the required data structure with its basic operations, you will then need to provide some statistics about the eye tracking data to the users and also allow them to interact with your application given below.

Programming Requirements:

First of all, you need to define a data structure to represent the data structure given in Figure 1. Please note that you need to use a linked list to represent the data points in a given page and you will need another one to keep track of these linked lists. You should not make any assumption about the number of data points recorded for a particular page and also you should not make any assumption about the number of pages used in a particular session. We will assume that a particular web page can be shown only once during one session (but of course, we can have many fixations on that page). Your program will need to follow these steps:

1. You need to read eye tracking data from an external txt file. Please note that you cannot make an assumption about the number of data points in a file. Assume that the file is given in the format in Table 1.
2. You will then need to create the data structure in Figure 1 to represent the list of fixations on each page.
3. You will then ask the users whether they would like to clean the data or not. If they would like to clean the data then the program will ask them to enter a threshold value for a fixation duration and you will then delete all the fixations which are below that fixation duration.
4. You will then display the following statistics to the user:
    - The number of fixations on a particular web page, basically the number of nodes created for a particular page.
    - The total duration of the fixations on a particular web page.
5. Finally, you will ask user if they would like to get these statistics for a particular area (Area of Interest or AOI) on a web page. If they do, then you need to get this area information from them. Basically, the user will enter the top left X, Y and the bottom X, Y coordinates, and you will give them the statistics (the number of fixations and the total duration of the fixations on the area).

In order to achieve these steps you need to have the following functions. **We will not accept your solution if you do not consider the specifications!**

- **`load_data_points`:** This function will take the file name as an input. It will then load the data from the file to your data structure defined in Figure 1. This data structure will be returned by this function.

```
/*input: file name that will be used to load points
return: a data structure to represent the overall data, see Figure 1*/
```

- **`clean_data_points`:** This function will take the data structure represented in Figure 1 and a threshold value. It will then remove all the nodes from all the pages that have fixation duration below that threshold value.

```
/*input: a data structure to represent the overall data, see Figure 1
return: void*/
```

- **`show_page_statistics`:** This function will take the data structure created in the function load_data_points, and ask for a page name. You will then use this page name which is entered as text to search the "Stimuli Name" column (see Table 1). The user does not need to enter the full name. If they enter, for example, BBC, then you will search for text "BBC" in the stimula column and return page "www.bbc.co.uk". It will then display the statistics on the page by using the `total_fixation_duration` and `count_fixation_points` functions.

```
/*input: a data structure to represent the overall data, see Figure 1
return: void, just display statistics on the screen*/
```

- **`total_fixation_duration`:** This function will get a linked list of fixations recorded for a page and it will return the total duration of the fixations in that list.

```
/*input: a linked list of fixations for a particular page
return: The total duration of the fixations */
```

- **`count_fixation_points`:** This function will get a linked list of fixations recorded for a page and it will return the number of fixations (i.e., nodes) in that list.

```
/*input: a linked list of fixations for a particular page
return: The number of fixations*/
```

- **`show_AOI_statistics`:** This function will get a linked list of fixations recorded for a page and the coordinates of an AOI. It will then calculate and display the statistics.

```
/*input: a linked list of fixations for a particular page, coordinates
of top left X,Y coordinates and bottom X, Y coordinates.
return: void, just display statistics on the screen*/
```

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: [Inputs are shown in bold]

```
A-Eye: Analysing Eye tracking Data
Enter file name: P1.txt
The recording is successfully loaded.

------------------------
A-Eye Menu
------------------------
1. Clean Data Points
2. Show Page Statistics
3. Show AOI Statistics
4. Exit from A-Eye
------------------------
Enter your option: 1
Enter a threshold value in milliseconds: 100
The number of fixations removed: 1

------------------------
A-Eye Menu
------------------------
1. Clean Data Points
2. Show Page Statistics
3. Show AOI Statistics
4. Exit from A-Eye
------------------------
Enter your option: 2
Enter a page name: BBC
The number of fixations: 95
The total duration of fixations: 28047 milliseconds

------------------------
A-Eye Menu
------------------------
1. Clean Data Points
2. Show Page Statistics
3. Show AOI Statistics
4. Exit from A-Eye
------------------------
Enter your option: 3
Enter a page name: BBC
Please enter the top [X,Y] and bottom [X,Y] coordinates of the AOI: [139,763][107,366]
The number of fixations: 22
The total duration of fixations: 5717 milliseconds

------------------------
A-Eye Menu
------------------------
1. Clean Data Points
2. Show Page Statistics
3. Show AOI Statistics
4. Exit from A-Eye
------------------------
Enter your option: 4
Goodbye!
```

Please note that when the program asks for a page name, the user will enter a page name, not its address. The program will analyse the addresses to find the relevant one. For example, if the user enters BBC, your program should find www.bbc.co.uk as a relevant address. If the user enters a page name which is not included in the data, the program will show a warning message (Not found!) and show the main menu again.

Grading:

Your program will be graded as follows:

Grading Point	**Points (100)**
load_data_points	30 points
clean_data_points	20 points
show_page_statistics	5 points
count_fixation_points	5 points
total_fixation_duration	5 points
show_AOI_statistics	20 points
Main function	15 points

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.