CS 2413 Data Structures – Spring 2016 – Programming Project 1
Due February 8, 2016 – 11:59 PM

Objectives

1. [10 pts] Use basic input/output in C++ (cin and cout).
2. [10 pts] Use arrays in C/C++.
3. [10 pts] Create at least 2 C++ classes.
4. [50 pts] Design and implement the program *according to the project description*. 50 pts are distributed as follows:
   a. [10 pts] Read the input file using *redirected input*; print out each line read from the file.
   b. [10 pts] Add url to appropriate browserTab and corresponding webAddressInfo
   c. [10 pts] Implement the forward and backward methods.
   d. [10 pts] Implement the display method in browserTab class.
   e. [10 pts] Implement the display method in webAddressInfo class.
5. [20 pts] Document your project thoroughly as the examples in the textbook. This includes but not limited to header comments for all classes/methods, explanatory comments for each section of code, meaningful variable and method names, and consistent indentation.

Project Description

A browser window consists of a many tabs. On each tab a user can enter a new web address (url) to go to, press the back button to go a url that was just visited, or press the forward button to go the url that was visited before the back button was pressed. A list of urls corresponding to a particular tab is kept in an array of characters. The goal of this project is to create appropriate classes and simulate the actions of a user on a web browser. When a user chooses forward or backward we need to just print the urls and do not have to write complex code to actually go to a website.

Given the above, we will first start with the description of the input. All input lines will be stored in a text file (file with txt extension) and we will use redirected input to read the contents of the file. Each line of input consists of three items (each separated by a single space): tab number (integer), action (single character), and the url (depending on the action). The possible actions that needs to be implemented for this project are N (for adding a url), B (pressing the backward button), and F (pressing the forward button), and P (for printing the urls).

For each action, print the action that is invoked, followed by the tab number, and followed by the result of the action. Both forward and backward actions should print the url that they go to. Add should provide the url that being added. The print P should print all the urls in the current tab.

Here is an example of the input file.

```
1 N http://www.msn.com/?ocid=iehp
1 N http://www.microsoftstore.com/store/msusa/en_US/home
1 B
1 F
1 P
3 N https://www.udacity.com/course/front-end-web-developer-nanodegree--nd001
1 N http://www.cnn.com/2016/01/24/us/weather-winter-snowstorm/index.html
1 P
1 B
1 B
1 P
2 N http://www.cnn.com/2016/01/24/us/weather-winter-snowstorm/index.html
2 N http://www.sas.com/en_us/software/university-edition.html
2 B
2 F
2 P
```

We will assume that the number of commands in the input file is unknown. Additionally, we will assume that the maximum number of tabs is 20 and the number of characters in each url is limited to 200.

In this project, you will write a C++ program that allows the users to perform all the actions described above. You will implement the C++ classes as described in this section and as well as the `main()` method which will read and execute the commands from standard I/O.

webAddressInfo Class

The webAddressInfo class is used to store the information on the urls in a particular tab, it is defined as follows:

```
class webAddressInfo
{
   private:
      char url[201]; //allow a maximum of 200 characters
      // other private methods if necessary for this class
   public:
      webAddressInfo ();
      webAddressInfo (char* inputString);
      void setWebAddressInfo(char* inputString);
      char* getWebAddressInfo();
      void display();
      // and other public methods if necessary
};
```

browserTab

The browserTab class is used to store the web address information for each tab, it is defined as follows:

```
class browserTab {
   protected:
      int numAddress; //Current number of web addresses in this tab
      webAddressInfo webAddresses [20]; //Web addreses in this tab
      int currentAddress; //index of current location in webAddresses
      // other private methods if necessary for this class
   public:
      browserTab();
      browserTab(char* inputString); //creates a new tab with the inputString
      webAddressInfo& forward();
      webAddressInfo& backward();
      void addAddress(char* inputString);
      void display();
      // and other public methods if necessary
};
```

main() method

The main() method reads different user actions from standard I/O as described earlier.
You will implement the main() method as the code provided below.

```
#include <iostream>
using namespace std;

// define and implement the required classes or functions here,
// or in separate files, you will need to include the headers
/  for these classes in the latter case.

int main()
{
   char buffer[256];
   browserTab myTabs[20];
   // other local variables used to store data temporally
```

```
    int tabNumber;
    char blank;
    while( !cin.eof())  // while end of line is not reached
    {
       while (!cin.eof()) {
             cin >> tabNumber;
             cin.get(blank);
             cin.get(action);
             switch (action) {
                   case 'N': { //New url
                               break;
                             }
                   case 'F': { //Forward
                               break;
                             }
                   case 'B': { //Backward
                               break;
                             }
                   case 'P': { //Print current
                               break;
                             }
                   default: { //illeagal action}
             }
       }

    }
    return 0;
}
```

Redirected Input

Redirected input provides you a way to send a file to the standard input of a program without typing it using the keyboard. To use redirected input in Visual Studio environment, follow these steps: After you have opened or created a new project, on the menu go to project, project properties, expand configuration properties until you see Debugging, on the right you will see a set of options, and in the command arguments type "< **input filename**". The < sign is for redirected input and the **input filename** is the name of the input file (including the path if not in the working directory). A simple program that reads character by character until it reaches enf-of-file can be found below.

```
#include <iostream>

using namespace std;

//The character for end-of-line is '\n' and you can compare c below with this
//character to check if end-of-line is reached.

int main () {

      char c;

      cin.get(c);
      while (!cin.eof()) {
            i = c;
            cout << c;
            cin.get(c);
      }
      return 0;
}
```

C String

A string in the C Programming Language is an array of characters ends with '\0' (NULL) character. The NULL character denotes the end of the C string. For example, you can declare a C string like this:

```
char aCString[9];
```

Then you will be able to store up to 8 characters in this string. You can use **cout** to print out the string and the characters stored in a C string will be displayed one by one until '\0' is reached. Here are some examples:

0	1	2	3	4	5	6	7	8	cout result	Length
u	s	e	r	n	a	m	e	\0	username	8
n	a	m	e	\0					name	4
n	a	m	e	\0	1	2	3	4	name	4
\0									(nothing)	0

Similarly, you can use a for loop to determine the length of a string (NULL is NOT included). We show this in the following and also show how you can dynamically create a string using a pointer

```
char aCString[] = "This is a C String.";  // you don't need to provide
                                           // the size of the array
                                           // if the content is provided
char* anotherCString;                      // a pointer to an array of
                                           // characters

unsigned int length = 0;

while( aCString[length] != '\0')
{
   length++;
}
// the length of the string is now known

anotherCString = new char[length+1];  // need space for NULL character

// copy the string

for( int i=0; i< length+1; i++)
   anotherCString[i] = aCString[i];

cout << aCString << endl;       // print out the two strings

cout << anotherCSring << endl;

delete [] anotherCString;     // release the memory after use
```

You check http://www.cs.bu.edu/teaching/cpp/string/array-vs-ptr/, other online sources or textbooks to learn more about this.

Constraints

1. In this project, the only header you will use is #include <iostream>.
2. None of the projects is a group project. Consulting with other members of this class on programming projects is strictly not allowed and plagiarism charges will be imposed on students who do not follow this.