CSCE2014 Programming Foundations II
Homework One
By Wing Ning Li

1 Problem Description

One of the basic data structures in computer programming is array, which we have studied extensively in PFI. In C/C++ programming language, we have a "data type" called cstring, which is used to store a string or a sequence of characters. In fact cstring is an array of characters with the ending character the null character, which is represented as `'\0'`. C/C++ language has many library functions manipulating cstring. In this assignment, we will implement, that is to write the code for, three functions that process cstring and array of characters. The first function should be used to implement the last two functions, which in term will be used in the next assignment to implement the standard library function **strtok**. The three functions are:

```
bool is_char_in_the_set(char c, const char *set);

char *find_first_not_in_the_set(char *str, const char *set);

char *find_first_in_the_set(char *str, const char *set);
```

2 Purpose

Review array and array processing. Use functions to decompose a problem solution. Understand and use pointers, arrays, and functions to solve a bit complex problem; in particular pointers to char, arrays of char, C-style strings, and functions taking C-string as parameters and returning a pointer to char. Practice writing functions that are similar to standard library functions to show our ability and command of basic data structures (arrays) and programming in C/C++. Build our confidence in using and programming arrays.

3 Design

For this part, the work has been done for us. Let us take a closer look at the interface and prototypes of the three functions. From the following description we try to understand what the functions do, or the outside view of the functions. When you implement these functions, the name of the function should be exactly the same as well as all the arguments and return.

```
bool is_char_in_the_set(char c, const char *set);
```

Given a character `c` (as a character including the possibility of the null character) and a cstring `set` (as a set of characters and the set of characters include the

terminating null character) as two arguments, the function returns value true if the character in `c` appears in the set of characters given by `set`, and value false otherwise.

```
char *find_first_not_in_the_set(char *str, const char *set);
```

Given a cstring `str` (as a string) and another cstring `set` (as a set of characters), return the address of the first character in `str` such that this character does not show up in `set` parameter. If no such character exists in `str`, return the NULL pointer value. We may assume `str` and `set` have proper pointer values that are not NULL.

```
char *find_first_in_the_set(char *str, const char *set);
```

Given a cstring `str` (as a string) and another cstring `set` (as a set of characters), return the address of the first character in `str` such that this character shows up in `set` parameter. If no such character exists in `str`, return the NULL pointer value. We may assume `str` and `set` have proper pointer values that are not NULL.

Sample test cases:

```
char test_string[15];
char test_set[10];
char *p; // pointer to the return character
strcpy(test_string, "Hi Everyone!");
strcpy(test_set, "e");
char x = 'h';
if ( is_char_in_the_set(x, test_string) )
    cout << x << " appears in the following string: " << test_string << endl;
else
    cout << x << " does not appear in the following string: " << test_string << endl;
//the output is h does not appear in the following string: Hi Everyone!
p = find_first_in_the_set(test_string, test_set);
cout << *p << endl;
// the above output should e and *(p-1) is v and *(p+1) is r
p = find_first_not_in_the_set(test_string, test_set);
cout << *p << endl; // the output should be H
```

4 Implementation

We are not allowed to use any global variable! No credit is given for using global variables. We are also not allowed to use any library function or class. No credit is given for using library functions or classes. For example, we cannot use string type or class.

We assume that someone (you or other folks) will use the three functions in their programs (main program or other functions). We will create two files:

helper.h and helper.cpp. Conceptually, helper.h is the header file to be included for any code that uses our functions just as we have to include other headers in order to use the library functions or types. The helper.cpp file (containing the code of the three functions given in the design section) is used to create the object file, say helper.o, to be linked together by the compiler to create the executable file. We will have another file called homework1.cpp that contains the main function for testing our implementation of the two functions. Here is the idea or hint about how to implement the last two functions (from what you learned in PFI we hope you have an idea of how to write the code for the first function and please ask for a hint if that is not the case)

```
char *find_first_not_in_the_set(char *str, const char *set);
```

1. loop through each character in `str` one by one.
2. if the current character in `str`, does not appear in `set` return the address of the current character
3. return NULL after the loop

```
char *find_first_in_the_set(char *str, const char *set);
```

1. loop through each character in `str` one by one.
2. if the current character in `str`, appears in `set` return the address of the current character
3. return NULL after the loop

Note that the second step in the pseudo code needs a loop as well. In general you may write a nested loop or introduce an additional function to address the logic. But for this assignment, you must to use the first function to help with the logic.

5 Test and evaluation

The goal of testing is to find bugs in our code. Your main program should have several test cases for different conditions of using the two functions. You may use the test cases given in the design section to begin the testing. After that you need to think about other test cases, for example, what if the `set` parameter contains more than one character. We hope to design the test cases such that each test case covers a new situation yet considered.

6 Report and documentation

A short report about things observed and things learned and understood. The report should also describe the test cases used in the main program and the reasons for each test case selected. Properly document and indent the source code. The source code must include the author name and as well as a synopsis of the file.

7 Project submission

Use Blackboard to submit all source files (helper.h, helper.cpp, homework1.cpp) and the project report in either .pdf or .doc or .docx file.