Notes for Programming in C Lab Session #2

9 October 2017

1 Introduction

The purpose of this lab session is primarily to ease you into programming in C, by writing a small example program.

2 Overview

In the first lecture, we saw how to reverse a whole string using a for loop. That is, given a variable `s` holding the string:

```
"University of Cambridge!"
```

we were able to modify the string so that it had the contents:

```
"!egdirbmaC fo ytisrevinU"
```

In this lab session, you will write a function that does not reverse the *whole* string, but instead reverses *each word* within the string. So instead, you will seek to modify the original string so that its contents are:

```
"ytisrevinU fo egdirbmaC!"
```

Moreover, in the second recorded lecture, we saw how to define C functions. So you will do this by implementing a small library of functions whose prototypes and specifications are given in `revwords.h`, and whose implementation should go in `revwords.c`.

3 Instructions

1. Download the `lab2.tar.gz` file from the class website.
2. Extract the file using the command `tar xvzf lab2.tar.gz`.
3. This will create a `lab2/` directory. Change into this directory using the `cd` command.
4. In this directory, there will be files `lab2.c`, `revwords.h`, and `revwords.c`.
5. There will also be a file `Makefile`, which is a build script which can be invoked with the command `make`. It will automatically invoke the compiler and build the `lab2` executable.
6. Run the `lab2` executable, and see if your program works.

4 The Functions to Implement

`void reverse_substring(char str[], int start, int end)`

`reverse_substring(s, start, end)` function takes a string s, and two integer indices start and end identifying the start and end of a substring of s. The function may assume that start and end are both valid indices nto the string.

`int find_next_start(char str[], int len, int i)`

`find_word_start(s, len, i)` takes a string s of length len, and an index i (which must be strictly less than len). It then returns the index k which is the starting position of the next word beginning at position i or later. If no such index exists, then it should return -1.

`int find_next_end(char str[], int len, int i)`

`find_word_end(s, len, i)` takes a string s of length len, and an index i (which must be strictly less than len). It returns the first index k past the end of the word starting at i.

`void reverse_words(char s[])`

`reverse_words(s)` takes a string s, and reverses all of the words in it. Here, a ”word” is defined as a contiguous sequence of alphabetic characters.