PROJECT 1 - INVERTED INDEX
Computer Systems Principles

Emery Berger Mark Corner

September 13, 2010

1 Overview

This project will give you experience writing a simple C++ program using the STL.

For this assignment, you will write a program in C++ that generates an "inverted index" of all the words in a list of text files. (See `http://en.wikipedia.org/wiki/Inverted_index` for more details.) The goal of this assignment is to ensure that you are sufficiently up to speed in C++ to handle the rest of the course. We will also use this program in subsequent assignments.

2 Inverter

2.1 Input

Your inverter will take exactly one argument: a file that contains a list of filenames. Each filename will appear on a separate line.

Each of the files described in the first file will contain text that you will build your index from.

For example:

inputs.txt:

```
foo1.txt
foo2.txt
```

foo1.txt:

```
this is a test. cool.
```

foo2.txt:

```
this is also a test.
boring.
```

2.2 Output

Your inverter should print all of the words from all of the inputs, in "alphabetical" order, followed by the document numbers in which they appear, in order. For example (note: your program must produce exactly this output):

```
a: 0 1
also: 1
boring: 1
cool: 0
is: 0 1
test: 0 1
this: 0 1
```

Alphabetical is defined as the order according to ASCII. So "The" and "the" are seperate words, and "The" comes first. Only certain words should be indexed. Words are anything that is made up of only alpha characters, and not numbers, spaces, etc. "Th3e" is two words, "Th" and "e".

Files are incrementally numbered, starting with 0. Only valid, openable files should be included in the count. (`is_open` comes in handy here)

Your program should absolutely not produce any other output. Extraneous output, or output formatted incorrectly (extra spaces etc.) will make the autograder mark your solution as incorrect. Please leave yourself extra days to work these problems out.

3 Implementation Hints

Implement the data structure using C++ Standard Template Library (STL) as a map of sets, as in:

```
map<string, set<int> > invertedIndex;
```

Use C++ strings:

```
#include <string>
```

and file streams:

```
#include <fstream>
```

Remember, your program needs to be robust to errors. Files may be empty, etc. Please handle these cases gracefully and with no extra output.

The `noskipws` operator may be useful in parsing the input: `input >> noskipws >> c;`

4 Handing Project In

Remember, make sure that your project uses an `ifstream`, not an `fstream`.

Your project will be handed in using the autograding system. Please see the web page for details on how to submit your solution.