URL Resolver

Introduction

Every web page has an address that can be used to view it in a web browser. Web page addresses are specified as URLs (Uniform Resource Locators), such as `http://www.byu.edu/webapp/home/index.jsp`.

What makes the web so interesting is that web pages can contain links to other pages, thus allowing users to easily navigate to related information. The content of web pages is written in HTML (Hyper-Text Markup Language). In HTML, links between web pages are represented as URLs. The HTML code for a web page may contain links to any number of other web pages, with each link specifying the URL of the link's target page.

The URL of a link's target page can be specified using either an **absolute URL** or a **relative URL**. An absolute URL fully specifies the address of the target page. These are the URLs you would manually type into a web browser to visit a web page, such as `http://www.cnn.com/`. An absolute URL contains all the information necessary for a web browser to download the page.

Alternatively, a relative URL specifies the address of the link's target page *relative to the address of the page containing the link*. For example, `../../images/football.jpg` is a relative URL. This URL means, "Start in the directory containing the current page (i.e., the page containing the link), go up two levels, go down into the images directory, and look for a file named football.jpg." A relative URL does not contain all the information needed for a browser to download the page, and it must first be converted (or resolved) to an absolute URL before the target page can be downloaded.

For example, the URL of the CS 240 home page is `http://students.cs.byu.edu/~cs240ta/index.php`. This page contains links to other pages containing information about course policies and programming projects. The absolute URLs for these pages are `http://students.cs.byu.edu/~cs240ta/info/policies.php` and `http://students.cs.byu.edu/~cs240ta/projects.php`. The HTML code for the home page could specify these links using either absolute URLs or relative URLs, as shown below:

http://students.cs.byu.edu/~cs240ta/index.php

```
<a href="http://students.cs.byu.edu/~cs240ta/info/policies.php">Course Policies</a>

<a href="http://students.cs.byu.edu/~cs240ta/projects.php">Programming Projects</a>
```

OR

http://students.cs.byu.edu/~cs240ta/index.php

```
<a href="info/policies.php">Course Policies</a>

<a href="projects.php">Programming Projects</a>
```

In this case, when a user clicks on the Course Policies link, the browser must determine the absolute URL of the link's target page so it can be downloaded into the browser. If the link was specified with an absolute URL, no extra work is necessary; the browser can find the link's absolute URL directly in the HTML code. On the other hand, if the link was specified with a relative URL, the browser must first translate (or resolve) the relative URL into an absolute URL before the target page can be downloaded. This would be done by starting with the directory containing the CS 240 home page, which is `http://students.cs.byu.edu/~cs240ta/`, and combining that with the relative URL of the policies page, which is `info/policies.php`. Putting these together, the resolved absolute URL for the policies page would be `http://students.cs.byu.edu/~cs240ta/info/policies.php`. Now that the relative URL has been resolved to an absolute URL, the target page can be downloaded by the browser.

Details of Relative URL Resolution

This section provides specific details on how to resolve relative URLs.

Absolute URLs

An absolute URL fully specifies the address of a web page. The general format of an absolute URL is:

<scheme>://<net_loc>/<path>?<query>#<fragment>

If present, the <query> and <fragment> parts will always appear in the order shown.

<scheme>	For this project, <scheme> will always be "http" or "file". An http URL refers to a web page. A file URL refers to a local file.
<net_loc>	Specifies the domain name and, optionally, the port number of the web server. This applies only to http URLs and will always be empty for file URLs (e.g., `file:///home/users/bob/photo.jpg`)
<path>	Specifies the hierarchical location of the web page.
<query>	Specifies query parameters that may be used by a web server to dynamically generate the content of the page.
<fragment>	Specifies a section name within the page. This is used to tell browsers which section of the page should be displayed when the page is loaded.

For example, consider the following absolute URL:

```
http://www.espn.com:80/basketball/nba/index.html?team=dallas&order=name#Roster
```

The parts of this URL are:

<scheme>	http

<net_loc>	www.espn.com:80
<path>	/basketball/nba/index.html
<query>	team=dallas&order=name
<fragment>	Roster

Several parts of an absolute URL are optional and will not appear in every URL. Specifically, the <path>, <query>, and <fragment> parts might not be present. If <path> is missing, it is assumed to be "/". If <query> is missing, it is assumed to be empty (in this case there will be no question mark). If <fragment> is missing, it is assumed to be empty (in this case there will be no pound sign).

The <scheme> and <net_loc> parts are case insensitive, but the <path>, <query>, and <fragment> parts are case sensitive. This means that

```
http://www.cnn.com/index.html
HTTP://WWW.CNN.COM/index.html
```

are equivalent, but

```
http://www.cnn.com/index.html
http://www.cnn.com/INDEX.HTML
```

are not equivalent.

Relative URLs

For this project, any URL that does not begin with http:// or file:// is a relative URL. Unlike absolute URLs, relative URLs do not fully specify the address of a web page. Rather, they specify the address of a page relative to the address of the document containing the link (the base document).

Here are some examples of relative URLs:

/apphome/images/nasdaq.jpg
./images/nasdaq.jpg
../../images/nasdaq.jpg
images/nasdaq.jpg
#HEADLINES

Resolving a relative URL is done by combining the absolute URL of the base document (the base URL) with the relative URL, as described below.

If the relative URL begins with a "/", the absolute URL is constructed by pre-pending file:// or http://<net_location> from the base URL (whichever applies). For example,

BaseURL	http://www.cnn.com/news/financial/index.html
RelativeURL	/images/nasdaq.jpg
ResolvedURL	http://www.cnn.com/images/nasdaq.jpg

BaseURL	file:///news/financial/index.html
RelativeURL	/images/nasdaq.jpg
ResolvedURL	file:///images/nasdaq.jpg

If the relative URL begins with "./", the URL is relative to the directory containing the base document. For example,

BaseURL	http://www.cnn.com/news/financial/index.html
RelativeURL	./images/nasdaq.jpg
ResolvedURL	http://www.cnn.com/news/financial/images/nasdaq.jpg

BaseURL	file:///news/financial/index.html
RelativeURL	./images/nasdaq.jpg
ResolvedURL	file:///news/financial/images/nasdaq.jpg

If the relative URL begins with "../", the URL is relative to the parent directory of the directory containing the base document. For example,

BaseURL	http://www.cnn.com/news/financial/index.html
RelativeURL	../images/nasdaq.jpg
ResolvedURL	http://www.cnn.com/news/images/nasdaq.jpg

BaseURL	file:///news/financial/index.html
RelativeURL	../images/nasdaq.jpg
ResolvedURL	file:///news/images/nasdaq.jpg

Note that a relative URL can begin with any number of “./” or "../" in any combination, such as "./../../../images/nasdaq.jpg". Your program should process these correctly, where each “./” means stay in the directory you’re already in, and each "../" means go up a level in the directory hierarchy.

If the relative URL begins with "#", the URL is relative to the base document itself. In this case, the fragment represents a specific location within the base document. For example,

BaseURL	http://www.cnn.com/news/index.html
RelativeURL	#HEADLINES
ResolvedURL	http://www.cnn.com/news/index.html#HEADLINES

BaseURL	file:///news/index.html
RelativeURL	#HEADLINES
ResolvedURL	file:///news/index.html#HEADLINES

If none of the preceding cases apply (i.e., it doesn't begin with "/", "./", "../", or "#"), the URL is relative to the directory containing the base document, just as if it had started with "./". For example,

BaseURL	http://www.cnn.com/news/financial/index.html
RelativeURL	images/nasdaq.jpg
ResolvedURL	http://www.cnn.com/news/financial/images/nasdaq.jpg

BaseURL	file:///news/financial/index.html
RelativeURL	images/nasdaq.jpg
ResolvedURL	file:///news/financial/images/nasdaq.jpg

Functional Specification

Write a program named `url-resolver` that resolves relative URLs. The command-line syntax for running your program should be as follows:

```
url-resolver <base-url> <relative-url>
```

where <base-url> is the absolute URL of the base document, and <relative-url> is the relative URL of a link. Your program should resolve <relative-url> relative to <base-url> and print out the resulting absolute URL.

For example,

```
url-resolver http://students.cs.byu.edu/~cs240ta/index.php info/policies.php
http://students.cs.byu.edu/~cs240ta/info/policies.php
```

Your program should validate the number of command-line arguments, and print out a usage statement and exit if the number is wrong. Beyond checking the argument count, you may assume that your program will be run with valid inputs.

Restrictions

You may not use the C++ string class, any non-standard C++ libraries, or the classes and functions from the Standard Template Library (vector, list, map, set, etc.). Specifically, the following header files may not be used:

<string> (BUT, you SHOULD use <cstring> or <string.h> to access C-string manipulation functions)
<algorithm>
<deque>
<list>
<map>
<queue>
<set>
<stack>
<vector>

CppCheckStyle

In order to pass off, your source code must satisfy CppCheckStyle (i.e., no errors).