

EECS 498 W26 System Design of a Search Engine

12 Robots.txt

Due Sunday, March 29, 2026, 11:59 pm

The purpose of this assignment is to create a RobotsTxt class that can parse a robots.txt file, compiling the rules and extracting any sitemaps, and allow user-agent names and paths or full URLs to be tested against the rules to determine if access is allowed.

Starter files

You are given these starter files in RobotsTxtStarter.zip.

RobotsTxt.h	An explanation of how a robots.txt file works and the definition RobotsTxt class for parsing and evaluating one.
RobotsTxt.cpp	A starter file for an implementation of the RobotsTxt class. You do not have to use this starter code. You are free to create a completely original solution. The autograder will only test that your implementation matches the description in RobotsTxt.h.
Robots.cpp	A simple demo program that can be used to test your RobotsTxt class.
Utf8.h	The same Utf8.h header you've seen before. Needed only the definition of the Utf8 type and Utf-8 BOM
SampleRobots.txt	A robots.txt file for testing.
Queries.txt	A set of queries consisting of user-agent name and URL pairs to be tested against the SampleRobots.txt file.
ExpectedResults.txt	Expected output from ./Robots SampleRobots.txt < Queries.txt
pause.cpp	A function that prints, "Press Enter to continue ..." and waits for the user to do that. Useful for pausing to allow attaching from a debugger.
Makefile	A makefile to build and test your RobotsTxt implementation.

You will need to supply your own Utf8.cpp to build RobotsTxt. (On the autograder, a working one will be provided.) But it's unlikely you'll need it, since Utf8 * can be safely cast to char * for functions like strlen() or atoi().

The assignment

You are to implement the RobotsTxt class defined in RobotsTxt.h and upload it as RobotsTxt.cpp to the autograder at <https://autograder.io/web/project/3494>.

The Robots utility

The Robots utility is a simple wrapper for the RobotsTxt class for testing. Here is an example of how it works. What I typed is in *red italics*.

```
tcsh-4% ./Robots
```

```
Usage: Robots [ -p ] filename
```

Compile the specified file as a robots.txt file, then compare read user and url pairs read from stdin against it, reporting whether access is allowed and any crawl delay.

-p Check paths, not full URLs.

```
tcsh-5% head SampleRobots.txt
```

```
User-agent: LinuxGet
```

```
User-agent: ZEBRA
```

```
User-agent: xx
```

```
Disallow: /foo
```

```
Allow: /foo/bar
```

```
sitemap: https://foo.x:50/y.xml
```

```
Allow: /foo/*ze*bra
```

```
Crawl-delay: 5
```

```
User-agent: *
```

```
tcsh-6% ./Robots SampleRobots.txt
```

```
The following sitemaps were found.
```

```
https://foo.x:50/y.xml
```

```
http://zork.com/map.xml
```

```
LinuxGet http://x.y/foo
```

```
Checking user = LinuxGet, URL = http://x.y/foo
```

```
Access denied
```

```
LinuxGet http://x.y/foo/bar
```

```
Checking user = LinuxGet, URL = http://x.y/foo/bar
```

```
Access allowed, crawl delay = 5
```

```
tcsh-7%
```