Dr. Mark Humphrys

School of Computing. Dublin City University.

Home Blog Teaching Research Contact

Online coding site: Ancient Brain

1,706 coders 5,262 JavaScript worlds

Search:

Search

CA170 CA2106 CA686 CA686I

Online AI coding exercises

Project ideas

Einstein - Search engine

1. Write an **offline** search engine to search **offline** web pages in your file system, and produce an **offline** output web page where you can click on links.
2. You may not have web pages to search, so we will test it on my **sample corpus of the works of Shakespeare**.
3. Call it `gweb` ("grep web"). Usage like:

```
gweb string
```

4. It searches the test corpus for the input string.
5. It sends its output into an (offline) output web page in your home directory called:

```
gweb.output.html
```

Set up

Start with this skeleton script:

```
# when testing, comment/uncomment the following "exec" line
# comment - output goes to screen
# uncomment - output goes to file


#       exec > $HOME/gweb.output.html
```

```
cd /users/tutors/mhumphrysdculab/share/shakespeare

echo '<pre>'
grep -i "$1"  */*html
echo '</pre>'
```

1. Test that it works with a sample search, like:

   ```
   gweb france
   ```

2. Comment/uncomment the line sending output to file or screen.

Escape the HTML

OK, so something works. Now you have to fix it.
First, the output looks a mess because it has HTML tags in it.

1. You need to pipe the grep output to a "sed" command to **print the HTML tags without interpreting them**.
2. This is sometimes called **"escaping"** the HTML tags.

Upload to Einstein

We are not finished yet, but this should already pass on Einstein.

1. Before upload, we need to make a change.
   Einstein is a different server.
   `/users/tutors` does not exist on Einstein.
   Change the Shakespeare directory to:

   ```
   /shared/humphrys/shakespeare
   ```

2. Rename it to gweb.sh
3. Upload and let Einstein test if it works.

Make the files clickable

For top marks on Einstein, make the files clickable.

1. The basic grep above gives output like this:

   ```
       file.html: hit
   ```

2. We are going to pipe the grep output to a **Shell function** called

2. We are going to pipe the grep output to a **Shell function** called "clickable", which constructs links to the files.
Start off by putting this shell function at the top of the script:

```
clickable()
{
 while read line
 do
  echo "$line"
 done
}
```

and pipe the above output to "clickable":

```
 grep -i "$1"  */*html  | sed ... | clickable
```

3. When that is working, we try to get "clickable" to separate the filename from the hit.
See **how to use cut** with grep output.
The function will look like:

```
clickable()
{
 while read line
 do
  file=` echo "$line" | [CUT BEFORE THE COLON] `
   hit=` echo "$line" | [CUT AFTER THE COLON]  `

  echo "$file"
  echo "$hit"
 done
}
```

You need to replace the **[CUT]** sections with the correct "cut" commands.

4. When that is working, we will construct the actual link.
The function will look like:

```
clickable()
{
 while read line
 do
  file=` echo "$line" | [CUT BEFORE THE COLON] `
   hit=` echo "$line" | [CUT AFTER THE COLON]  `
  echo "<a href='$file'>$file</a>: $hit <br>"
 done
}
```

5. You can now click on hits in the output page to see them (offline).
6. But wait, clicking on that link does not work. Why? How do you fix it?
   Recall where the Shakespeare pages are on the student server.
   And recall where the Shakespeare pages are for the script you upload to Einstein.

Test

1. For example:

   ```
   gweb northumberland
   ```

   will make an output file, showing all lines in the corpus where "northumberland" appears in any case, where you can click on the hits to see the file.

2. Upload to Einstein for top marks.
3. Follow the pattern above: **A linked filename, a colon, and the hit, all on one line. One of these lines per hit.** While it would be nice to deviate from this pattern - for example group multiple hits under one linked filename - this may confuse Einstein and it may not be able to tell your solution is correct.