PIC 40A: Homework 6 (due 5/29 at 5pm)

Like on previous homeworks, it is important that you meet the following requirements.

- You must upload your files to **Gradescope** before the deadline.
- You must upload your files to the **PIC server** in the appropriate directory before the deadline.
- Both submissions must be **identical** (down to the character).

  Never make changes to the PIC server submission after the deadline.

  (We can see when a file was last modified.)
- You must tell us (me and the grader) your **PIC username**.
- You must **validate** your HTML using https://validator.w3.org/.

  Ideally, with PHP, you should check that, after removing the PHP parts, the remaining HTML validates.

In this assignment you will submit two files (yes, just two!)...

1. `README.txt`. This will contain your PIC username.
2. `tidy.php`.

As mentioned above, you should submit all files to Gradescope before the deadline.
You should also submit the files to the PIC server. Save them in the directory

`/net/laguna/???...???/your_username/public_html/HW6`

(in the folder `HW6` within `public_html`). We should all be able to view your live webpage at

www.pic.ucla.edu/~your_username/HW6/tidy.php

Now, I am just left to tell you what I want `tidy.php` to achieve. Go over the page for that!

What `tidy.php` should do...

Here's a screenshot of what my solution looks like.

So, in terms of the HTML...

- `<title>` can be set to "PIC 40A Demo".
- There should be a heading within the header saying "PIC 40A Demo - Tidy Trailing Space".
- There should be a web form accepting a file submission.
  - It should `accept .txt` files.
  - It should redirect to `$_SERVER['PHP_SELF']`.
- The page should have the footer that all your other assignments have.
- The HTML should validate (if you remove the PHP parts and the `action` attribute) or if you give the URL to the validator website.

In terms of the PHP...

- Clicking "Submit" before selecting a file should result in the page reloading with no changes.
- Selecting a file and submitting it should cause a new file to be downloaded.
  - The file should have the same name as the file submitted, but with `tidy_` prepended to it. So if `mac.txt` is submitted, `tidy_mac.txt` should be downloaded.
  - The downloaded file should be the same as the submitted file *except for whitespace*. The downloaded file should be the result of "tidying" the submitted file. I'll explain what I mean by "tidying" over the page.
  - After the file has downloaded, no new files should be present on the PIC server.
  - Occasionally, a submission may make the Sucuri Website Firewall unhappy; that'll probably only happen if you try and submit files containing code.

"Tidying"

Imagine you are typing a novel. You type line by line. Each line, you type something and then

- either you press the enter key
- or you finish your novel!

Suppose you start a new line. The moment before pressing the enter key you have a string `s` which *does not contain* `"\r"` *or* `"\n"`. If you choose to press the enter key, one of two things might happen:

1. `"\n"` is appended to `s`;
2. `"\r\n"` is appended to `s`.

Your operating system and text editor settings will determine which happens.

- Macs, by default, have lines of the form `s."\n"` and `s.""`. (Recall that `.` is concatenation.)
- Windows machines, by default, have lines of the form `s."\r\n"` and `s.""`.

By "tidying" we mean that for each line, we replace `s` by `rtrim(s)`. (`rtrim` is a function in PHP). So for each line, we apply a function `tidy` which takes

```
s . ""      ⟼  rtrim(s) . ""
s . "\n"    ⟼  rtrim(s) . "\n"
s . "\r\n"  ⟼  rtrim(s) . "\r\n"
```

(where `s` does not contain `"\r"` or `"\n"`). I have not specified what the `tidy` function should do for the input `"abc \n def \r\n tuv \n"`. That does not matter because when you loop through the file line by line, such a string will never show up.

I have given you `mac.txt` and `tidy_mac.txt` so you can see an example of "tidying" on a file created by my Mac. Observe the file sizes: 587 bytes and 477 bytes.

$$587 - 477 = 8 + 28 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 + 0 + 28 + 8 + 2.$$

I have given you `windows.txt` and `tidy_windows.txt` so you can see an example of "tidying" on a file created by my PIC Windows machine. Observe the file sizes: 616 bytes and 506 bytes.

$$616 - 506 = 8 + 28 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 + 0 + 28 + 8 + 2.$$

Further tidying

As noted above, you should make sure not to leave any newly created files behind on the server. You can use `unlink` to delete submitted files from the `tmp` directory where they are saved.

The nicest solutions to the problem will download the tidied file without creating any copies of it on the server. However, some of you may find it easier to create a temporary file on the server. If you do this, you should call the temporary file `_0_123_tmp_4_8888_8888.txt` because I can't imagine an important file being called that, and you should make sure to delete it afterwards.

Being honest

I have not specified what the `tidy` function should do for the input `"abc \r def \n"` because I have assumed `"\r"` will not show up in the middle of a line. In reality, this could show up as a line, but I am ignoring it because who types `"\r"` these days?? According to the top answer here:

> The only reason to use `"\r"` is if you're writing to a character terminal (or more likely a "console window" emulating it) and want the next line you write to overwrite the last one you just wrote (sometimes used for goofy "ascii animation" effects, e.g. progress bars) – this is getting pretty obsolete in a world of GUIs, though.

I am happy for you to ignore such weirdness if it'll make the question easier for you.
However, I can handle all cases by using only the functions `substr` and `rtrim`.

Grading

One point for each of the following...

- The page looks like it is supposed to and clicking "Submit" before selecting a file should result in the page reloading with no changes.
- When choosing files, only `.txt`-like files are allowed.

  These include, for example, `.php` files, but **not** `.pdf`s.
- Upon choosing a file and clicking submit, we remain on the same page, and a file is downloaded.
- The downloaded file has the correct name.
- The downloaded file is related in some way (however incorrect) to the file that was uploaded.
- The file downloaded is correct when a Mac type `.txt` file is uploaded.
- The file downloaded is correct when a Windows type `.txt` file is uploaded.
- No extra files have been created on the PIC server. Any temporary files have been deleted.