Homework #2

Red Correction Date: 2024/02/29 13:00

Due Time: 2024/03/04 (Mon.) 21:59

Contact TAs: `vegetable@csie.ntu.edu.tw`

Instructions and Announcements

- **NO LATE SUBMISSION OR PLAGIARISM IS ALLOWED.**
- Discussions with others are encouraged. However, you should write down your solutions **in your own words**. In addition, for **each and every** problem you have to specify the references (the URL of the web page you consulted or the people you discussed with) on the first page of your solution to that problem.
- Some problems below may not have standard solutions. We will give you the points if your answer is followed by reasonable explanations.

Submission

- Please place your answers in the same order as the problem sheet and do not repeat problem descriptions, just organize them by problem number in a tidy manner.
- Please zip all the files, including one PDF and one shell script, name the zip file "`{your_student_id}.zip`", and submit it through NTU COOL. The zip file should not contain any other files, and the directory layout should be the same as listed below:

```
{your_student_id}/
+-- compare.sh
+-- report.pdf
```

Grading

- There are some tests for each of the first to the fifth parts. You'll get the score of a part iff you pass all the tests of that part.
- We'll use `diff` command to compare your output and the answer of a test. Passing a test means that your output is the same as the answer.
- There are two kinds of tests: public tests and private tests. Public tests are released in test.zip; while private tests will be released after the homework deadline.
- It's possible you don't get full credits on the report. You should show how you write the codes step by step and list the references.
- Tidiness score: 3 bonus points (depends on the report), graded by TA.
- Final score = script score + report score + tidiness score.

System Administration

這題的目標是要寫出一個程式 `compare.sh`，比較兩個檔案或兩個目錄的差別。

0 注意事項

1. 請在 script 前加上 shebang 並開啟 script 的執行權限 (chmod +x)。
2. 你的 script 必須能在以下幾種 shell 中之一執行，bash、sh、zsh、fish、tcsh 或 ksh。
3. 我們會在工作站 (ws1) 測試你的 script，請確認所有使用到的指令在工作站上皆可以正常運作。
4. 此部分需全部使用 shell script 完成，不可在中途執行自己的 python 腳本或使用其他程式語言，如果不確定是否可以使用某些指令，請寄信詢問助教。
5. 請勿在 shell script 內新增檔案，也請勿夾雜惡意程式碼，並在一分鐘內成功執行。
6. 列出檔案只能使用 ls，禁止使用 tree 或類似功能的指令，若不確定指令是否可以使用，請先詢問助教。
7. 本作業中的檔案只會出現三種：regular file, symbolic link file, directory file (定義可以參考這個連結)。
8. 檔案名稱只含有 [0-9a-zA-Z] 、”.” 、”-” 和”_” 字元，且檔名開頭不會是”-”。
9. 所有檔案的大小總和 $\leq$ 1 MB，且單個檔案的總行數 $\leq$ 1000。
10. 以下子題 1-5 是以逐步加上功能的方式進行。也就是完成子題 1 的功能後，子題 2 將子題 1 的 compare.sh 再加上新的功能。最後只需要繳交完成版的 compare.sh，但是我們會針對每個子題的需求逐一驗證進行評分。

1 參數檢查（10pts）

下列方框中為 `compare.sh` 指令的說明。當使用`-h` 參數時，或者使用者所給的參數錯誤時，將會印出：

```
usage: ./compare.sh [OPTION] <PATH A> <PATH B>
options:
-a: compare hidden files instead of ignoring them
-h: output information about compare.sh
-l: treat symlinks as files instead of ignoring them
-n <EXP>: compare only files whose paths follow the REGEX <EXP>
-r: compare directories recursively
```

參數所需符合的條件或相關說明如下：

1. 如果 `[OPTION]` 中有`-r`，則 `<PATH A>`、`<PATH B>` 都要是目錄，否則都不能是目錄。
2. 如果 `[OPTION]` 中有`-a` 或`-n`，則必需有`-r`。
3. `<PATH A>`、`<PATH B>` 需出現在所有 `[OPTION]` 的後面。
4. 不能有方框中沒有定義的參數。

5. 所有方框中的參數皆需照其定義的方式出現，例如`-n` 後面要接額外的字串 `<EXP>`，而`-l` 後面不能有額外的字串。
6. 所有 symlink 皆視為非目錄。

如違反上述條件，或是 `[OPTION]` 中有`-h`，則輸出方框中的說明內容並結束執行。否則，不能輸出方框中的內容並且繼續執行程式。這個小題只要求能在對的時機輸出方框中的內容即可得分。

(Hint: `getopts`)

2 比較檔案（20pts）

這個小題的指令皆為

```
./compare.sh <FILE A> <FILE B>
```

的型式。注意到只要兩者都是檔案，即使是隱藏的檔案，都算是合法的參數。而如果其中有 symbolic link ，需有`-l` 參數才會視為是檔案（在這個小題中不會有這個參數，但在第五小題中可能有），否則視為檔案不存在，需輸出上題方框中的內容。在判斷參數合法之後：

- 如兩檔案內容一樣，則什麼都不需要輸出。
- 如兩檔案內容有不同之處，則需輸出

  ```
  changed ${x}%
  ```

其中 `${x}` 計算方法如下：

1. 如果 `<FILE A>` 或 `<FILE B>` 其中之一並非文字檔(即兩者使用 `diff` 指令比較之後會輸出 `binary files <FILE A> <FILE B> differ`)，則 `${x}`$:= 100$。
2. 如果 `<FILE A>` 及 `<FILE B>` 皆為文字檔，則計算要將 `<FILE A>` 修改成 `<FILE B>` 所需要的編輯數量：需要刪除 a 行、插入 b 行、保留 c 行。則計算 `${x}`$:= \lfloor \frac{100\max(a,b)}{\max(a,b)+c} \rfloor$ 的最小可能值，即當 c 有最大值時。

舉例來說設兩個檔案的內容分別為

```
hello
world
and
hello
kitty
```

以及

```
hi
world
and
kitty
```

可以發現當 $(a,b,c) = (2,1,3)$ 時，$\lfloor \frac{100\max(a,b)}{\max(a,b)+c} \rfloor$ 有最小可能值 40，所以 `${x}`$= 40$。
(Hint: `diff -d` 指令的輸出會保證刪除/插入的行數的最小性)

3 比較目錄（30pts）

這個小題只會使用 `compare.sh -r <PATH A> <PATH B>` 的執行方式，而 `<PATH A>` 及 `<PATH B>` 都需為目錄。

考量 `<PATH A>` 底下或者 `<PATH B>` 底下的每個檔案，假使任一檔案 `<FILE X>` 相對於 `<PATH A>` 或 `<PATH B>` 的相對路徑為 `<PATH X><FILE X>`，則以下列規則輸出。以下存在的定義是指其為非隱藏的 regular file。

1. 若 `<PATH X><FILE X>` 存在於 `<PATH A>` 底下，但不存在於 `<PATH B>` 底下，則輸出

   `delete <PATH X><FILE X>`

2. 若 `<PATH X><FILE X>` 不存在於 `<PATH A>` 底下，但存在於 `<PATH B>` 底下，則輸出

   `create <PATH X><FILE X>`

3. 若 `<PATH X><FILE X>` 同時存在於 `<PATH A>` 底下及 `<PATH B>` 底下；假設 `<PATH A><PATH X><FILE X>` 為 `<FILE A>`，`<PATH B><PATH X><FILE X>` 為 `<FILE B>`。則：

   (a) `<FILE A>` 和 `<FILE B>` 檔案內容相同，則完全不輸出。

   (b) `<FILE A>` 和 `<FILE B>` 檔案內容不同，則輸出

   `<PATH X><FILE X>: ${Y}`

   其中 `${Y}` 為 `./compare.sh <PATH A><PATH X><FILE X> <PATH B><PATH X><FILE X>` 的輸出。

以上的所有輸出的順序，皆需照 `<PATH X><FILE X>` 之字典序由小到大輸出。

舉例來說，假設 `dir1`、`dir2` 兩個目錄的內容如下：

```
dir1
├── code
│   ├── a.out
│   ├── helloworld.cpp
│   ├── output1.txt
│   └── output2.txt
├── .code.backup
│   ├── a.out
│   ├── helloworld.cpp
│   ├── output1.txt
│   └── output2.txt
├── hello.txt
├── link1 -> code/helloworld.cpp
├── link2 -> .code.backup
├── link3 -> link2
└── run.sh
```

```
dir2
├── code
│   ├── a.out
│   ├── helloworld.cpp
│   ├── output1.txt
│   └── output2.txt
├── .code.backup
│   ├── a.out
│   ├── helloworld.cpp
│   ├── output1.txt
│   └── output2.txt
├── hi.txt
├── link1 -> code/helloworld.cpp
├── link2 -> .code.backup/a.out
├── link3 -> link2
└── run.sh
```

則

```
./compare.sh -r dir1 dir2
```

需輸出

```
code/a.out: changed 100%
code/helloworld.cpp: changed 8%
code/output2.txt: changed 40%
delete hello.txt
create hi.txt
run.sh: changed 11%
```

(其中檔案的內容請參閱 `test.zip`)

4 隱藏的檔案與 Symlink（15pts）

這個小題 `[OPTION]` 一定會出現 `-r`，以及可能出現 `-a`、`-l`。若 `[OPTION]` 中有 `-a`，則將隱藏的檔案視為存在，否則視為不存在；若 `[OPTION]` 中有 `-l`，則將任何 symlink 都視為檔案（即使其指向的可能是目錄），否則視為不存在。其中 symlink 的內容定義成是其指向的路徑。因此，如果兩個 symlink 的內容同時是絕對路徑或同時是相對路徑，且其絕對/相對路徑一樣，即 `readlink` 的結果完全相同，則視為一樣（不輸出），否則視為 100% 不一樣。而如果是比較一個一般的檔案和一個 symlink 時，亦視為 100% 不一樣。

舉例來說，假設 `dir1`、`dir2` 兩個目錄的內容跟上題一樣，則

```
./compare.sh -r -l -a dir1 dir2
```

需輸出

```
.code.backup/a.out: changed 100%
.code.backup/helloworld.cpp: changed 8%
.code.backup/output2.txt: changed 40%
code/a.out: changed 100%
code/helloworld.cpp: changed 8%
code/output2.txt: changed 40%
delete hello.txt
create hi.txt
link2: changed 100%
run.sh: changed 11%
```

5 僅比較特定的檔案（15pts）

這個小題 `[OPTION]` 沒有任何限制。若 `[OPTION]` 中有`-n <EXP>`，則對於第三小題中所需判斷的 `<FILE X>` 或 `<FILE Y>`，只需檢查檔名滿足 `<EXP>` 這個 regular expression 的檔案，即在 shell script 中滿足 `<FILE X> =~ <EXP>` 或 `<FILE Y> =~ <EXP>`。

6 Report（10+3pts）

1. 列出你所參考的網站與哪位同學討論
2. 寫下你做了以上哪些小題
3. 解釋你的程式碼各個部份的功能