Homework 2
Shell Script

tsclu

國立陽明交通大學資工系資訊中心

Information Technology Center, Department of Computer Science NYCU

Usage

- Please provide a executable shell script with following available options:

```
$ hw2.sh -a
hw2.sh -i INPUT -o OUTPUT [-c csv|tsv] [-j]

Available Options:

-i: Input file to be decoded
-o: Output directory
-c csv|tsv: Output files.[ct]sv
-j: Output info.json
```

Error Return Code

- Invalid arguments should be rejected with a non-zero status code, with the exact help message outputted to **`stderr`**.

```
$ hw2.sh -a
hw2.sh -i INPUT -o OUTPUT [-c csv|tsv] [-j]

Available Options:

-i: Input file to be decoded
-o: Output directory
-c csv|tsv: Output files.[ct]sv
-j: Output info.json
```

Input File Specification - Introduction

- The to-be-decoded file should have a extension **`.hw2`**
  which is actually a YAML file.

Input File Specification - Definition

```
name: example.hw2
author: generator
date: 1696164685
files:
  - name: example.txt
    type: file
    data: >-

TG9yZW0gaXBzdW0gZG9sb3Igc2l0IGFtZXQsIGNvbnNlY3RldHVyIGFkaXBpc2NpbmcgZWxpdC4gTWF1cmlzIGVsZWlmZW5k
IHNlbSBsZW8sIGlkIHRpbmNpZHVudCBsb3JlbSB2ZW5lbmF0aXMgZWdldC4=
    hash:
      md5: 9ce543926bc4f3d67368b58e61fb7710
      sha-1: 375177a2829947c4cefbf756443c6451daab25e3
```

example.hw2

Input File Specification - Definition

```
name: example.hw2
author: generator
date: 1696164685
files:
  - name: example.txt
    type: file
    data: >-

TG9yZW0gaXBzdW0gZG9sb3Igc2
IHNlbSBsZW8sIGlkIHRpbmNpZH
    hash:
      md5: 9ce543926bc4f3d
      sha-1: 375177a282994
```

example.hw2

- Note:
  - `files.length` >= 0.
  - `files[*].name` will always not exact equal to either **`files.csv`** or **`info.json`** since they are reserved.
  - `files[*].type` is always "file" or "hw2".
  - `files[*].name` may imply a non-flatten folder structure. In this case, the required folder should be made. See `generated.hw2` in the Sample Input.
  - The .hw2 files are not guaranteed to be pretty-printed.

Expected Output files - Structure

- You should write the required files to the designated output directory as output.
- The following file tree is expected if **`example.hw2`** is the input file.

```
.
└── outputDir/
    ├── example.txt
    ├── files.csv
    └── info.json
```

Expected Output files - Contents

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris eleifend sem leo, id
tincidunt lorem venenatis eget.
```

example.txt

```
filename,size,md5,sha1
example.txt,116,9ce543926bc4f3d67368b58e61fb7710,f1d2e348391c502880eb246184f542556005874b
```

files.csv

Expected Output files - Contents

```
filename    size    md5    sha1
example.txt    116    9ce543926bc4f3d67368b58e61fb7710
f1d2e348391c502880eb246184f542556005874b
```

files.tsv

- '\t' will not included in filename.

```
{
    "name": "example.hw2",
    "author": "generator",
    "date": "2023-10-01T20:51:25+08:00"
}
```

info.json

Shellcheck

- ShellCheck is a static analysis tool for shell scripts. It helps catch errors and suggests improvements to make your scripts better.
- shellcheck -s sh -a [your_script] (should return zero)
- Version : 0.9.0

Recursive Decoding

```
name: example2.hw2
author: generator
date: 1696164685
files:
  - name: example.hw2
    type: hw2
    data: >-

bmFtZTogZXhhbXBsZS5odzIKYXV0aG9yOiBnZW5lcmF0b3IKZGF0ZTogMTY5NjE2NDY4NQpmaWxlczoKICAtIG5hbWU6IGV4
YW1wbGUudHh0CiAgICB0eXBlOiBmaWxlCiAgICBkYXRhOiA+LQogICAgICBURzl5WlcwZ2FYQnpkVzBnWkc5c2IzSWdjMmww
SUdGdFpYUXNJR052Ym5ObFkzUmxkSFZ5SUdGa2FYQnBjMk5wYm1jZ1pXeHBkQzRnVFdGMWNtbHpJR1ZzWldsbVpXNWtJSE5s
YlNCc1pXOHNJR2xrSUhScGJtTnBaSFZ1ZENCc2IzSmxiU0IyWlc1bGJtRjBhWE1nWldkbGRDND0KICAgIGhhc2g6CiAgICAg
IG1kNTogOWNlNTQzOTI2YmM0ZjNkNjczNjhiNThlNjFmYjc3MTAKICAgICAgc2hhLTE6IDM3NTE3N2EyODI5OTQ3YzRjZWZi
Zjc1NjQ0M2M2NDUxZGFhYjI1ZTMKCg==
    hash:
      md5: 97b3bb3e66dc3423e651523294fbfce0
      sha-1: 1f100bee54f9f1cc53ac7d052992e9b19a69e70e
```

- Note:
  - Recursively decode it when `type` is `hw2`.
  - The recursive rule should be activated if option `-c` and `-j` are both not enabled.

Recursive Decoding - After decode

```
name: example.hw2
author: generator
date: 1696164685
files:
  - name: example.txt
    type: file
    data: >-

TG9yZW0gaXBzdW0gZG9sb3Igc2l0IGFtZXQsIGNvbnNlY3RldHVyIGFkaXBpc2NpbmcgZWxpdC4gTWF1cmlzIGVsZWlmZW5k
IHNlbSBsZW8sIGlkIHRpbmNpZHVudCBsb3JlbSB2ZW5lbmF0aXMgZWdldC4=
    hash:
      md5: 9ce543926bc4f3d67368b58e61fb7710
      sha-1: 375177a2829947c4cefbf756443c6451daab25e3
```

Recursive Decoding - Expected Output Files' Structure

- You should write the required files to the designated output directory as output.
- The following file tree is expected if **`example2.hw2`** is the input file.

```
.
└── outputDir/
    ├── example.txt
    └── example.hw2
```

Requirements

- Please place your script at **`/home/judge/hw2.sh`**, with executable bit set.
- The script should start with a proper shebang (**`#!/bin/sh`**, other shells are not allowed.)
- Make sure **`/tmp`** is writable for the **`judge`** user.
- SFTP support for the SSH server is required.
- Your script should return the count of *Invalid Files* (which **any** of its checksum mismatches).

Restrictions

- Must not call network tools (such as curl, wget...)
- Must not use any other interpreters, compilers or programming languages (such as Python, Ruby, Node.js, Golang, Rust, Perl, GCC, Clang…)
- Must not call any other self-written scripts, binaries or executables.
- Only one shell, `sh`, is allowed.
- Common tools (e.g. `date`, `openssl`, `yq`, etc.) are allowed.
- If you are not sure whether a tool is allowed, please ask TA on Google Groups.

Grading

Automated grading (Online Judge), 104 pts.

- Usage
  - Invalid options
    - Exit Code (3%)
    - Help Message (4%)
  - Invalid files (8%)
- Shellcheck (10%)

- Input and Output files
  - Arbitraty argument position (10%)
  - Extract single file (10%)
  - Extract multiple files (15%)
  - Output nested directories (10%)
  - Extract & Output **`file.csv`** (8%)
  - Extract & Output **`info.json`** (10%)
  - Extract & Output **`file.tsv`** (6%)
  - Recursion (10%)

HW 2: Sample input

https://nasa.cs.nycu.edu.tw/sa/2023/slides/hw2-sample.tar

SH(1): getopts

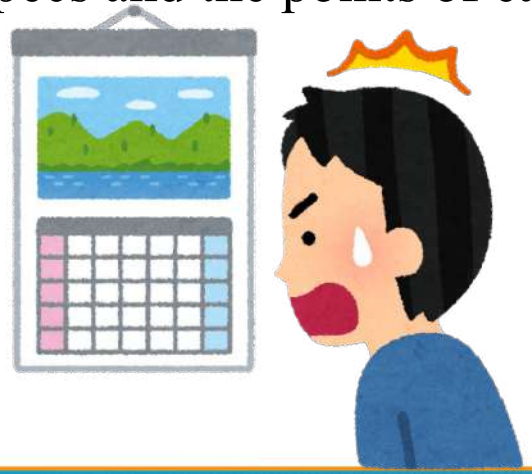
`getopts` *optstring var*

The POSIX `getopts` command. The getopts command deprecates the older `getopt(1)` command.

The first argument should be a series of letters, each possibly followed by a colon which indicates that the option takes an argument. The specified variable is set to the parsed option. The index of the next argument is placed into the shell variable `OPTIND`. If an option takes an argument, it is placed into the shell variable `OPTARG`.

Attention!

- Set the IP address 10.113.$ID.11
- You are restricted to use only sh to complete your work
  - That is, no other shell and no other programming language is allowed.
  - If you're not sure what's allowed, contact TAs.
  - TAs reserve the right of final explanations. Specs and the points of each sub-judges are subject to change in any time.
- Start from: Thu, 10/5 19:00
- Due date: Wed, 10/25 23:59



Help me! TA!

- Questions about this homework
  - Ask questions on https://groups.google.com/g/nctunasa
  - We MIGHT give out hints on google group
    - Be sure to join the group :D
  - Do not email us directly
  - Do not use E3 to email us
- How To Ask Questions The Smart Way
  - https://github.com/ryanhanwu/How-To-Ask-Questions-The-Smart-Way

Good Luck!

國立陽明交通大學資工系資訊中心

Computer Center of Department of Computer Science, NYCU