Assignment 1: Familiarization with shell scripting and shell commands

Nakul Aggarwal 19CS10044

Hritaban Ghosh 19CS30053

In this README we provide you with the readable version of our shell scripts for each part of the assignment. In order to achieve the minimum number of words, we had compromised on the readability, and thus we are providing you with an alternative way to understand the shell scripts.

Part a

```
x=$1;
d=2;
while [[ $d -lt $x || $d -eq $x ]];
do
    while [[ $(($x % $d)) -eq 0 ]];
    do
        echo -n " "$d ;
        x=$(($x / $d)) ;
    done;
    d=$(($d + 1)) ;
done;
echo "";
```

Part b

```
indir="1.b.files"
dir="1.b.files.out"
mkdir $dir
for f in $indir/*.txt;
do
    fn=(${f//// })
    fnn=${fn[-1]}
    sort -n $f -o $dir/$fnn
done;
cat $dir/*.txt | sort -n | uniq -c | awk '{print $2" "$1}' > $dir/1.b.out.txt
```

Part c

```
dir="data1c";
cd $dir;
exts=$(find "./data" -type 'f' -name "*.*"|sed -e 's/.*\.//'|sed -e 's/.*\///'|sort -u)
for ex in ${exts[@]};
do
    mkdir -p $ex
    find "./data" -type 'f' -name \*.$ex | xargs -n 10000 mv -f -t $ex
done;

mkdir -p "Nil"
if [ $(find "./data" -type 'f' ! -name "*.*"|wc -l) -ne 0 ];
then
    find "./data" -type 'f' ! -name "*.*" | xargs -n 10000 mv -f -t "Nil";
fi;

find "." -empty -type 'd' -delete
```

Part d

```
dir="data1d/temp"
newdir="files_mod"
mkdir $newdir
for f in $dir/*;
do
    fn=(${f//// })
    fnn=${fn[-1]}
    cat -n $f > $newdir/$fnn
    sed -i "s/^[ ][ ]*//g" $newdir/$fnn
    sed -i "s/\t/,/g" $newdir/$fnn
    sed -i "s/ /,/g" $newdir/$fnn
done;
```

Part e

```
> "valid.txt"
> "invalid.txt"

function log () {
  if [[ $_V -eq 1 ]];
  then
    printf "\n "
    echo "$@"
  fi;
};

log "[PART 1] Introducing Environment Variables"
export REQ_HEADERS="User-Agent,Connection,Keep-Alive,Host,Accept,traceparent"

log "[PART 2] Fetching & Saving WebPage"
curl -s -o "example.html" -G "https://www.example.com/"

log "[PART 3.1] Get & Print IP Address"
curl -s -G "http://ip.jsontest.com/"|jq -r '.ip'
log "[PART 3.2] Get & Print Response Headers"
curl -s -G --head "http://ip.jsontest.com/"
```

```
log "[PART 4] Print Required Headers from JSON Response"
reqheadslist=".\""${REQ_HEADERS//,/"\",.\""}"\""
echo "Required Header Keys:"$REQ_HEADERS
curl -s -G "http://headers.jsontest.com/"|jq "$reqheadslist"

log "[PART 5] Classify JSON Files as VALID or INVALID"
readarray -d '' entries < <(printf '%s\0' JSONData/*|sort -zV)
for entry in "${entries[@]}";
do
  isval=$(curl -s -d "json=`cat $entry`" -X POST "http://validate.jsontest.com/"|jq -r '.validate')
  if [[ $isval == "true" ]];
  then echo ${entry##*/} >> "valid.txt";
  else echo ${entry##*/} >> "invalid.txt";
  fi;
done;
```

Part f

```
cut -d ' ' -f$2 $1 | tr '[:upper:]' '[:lower:]' | sort | uniq -c | sort -bnr | awk '{print $2" "$1}' > 1f_output_$2_column.freq
```

Part g

```
cat /dev/urandom | awk -v OFS=',' '{for(i=0;i<9;i++) printf "%d,", int((NR+1) * 32768 * rand()); print int((NR+1) * 32768 * rand())}' | head -n 150 > $1 ;

count=$(cut -d ',' -f $2 $1|grep -c $3)

if [[ $count -ne 0 ]];
then echo "YES";
else echo "NO";
fi;
```