EJERCICIO 1 DATES

#!/bin/bash

function check\_format {

echo $1 | grep -Ex "[0-9]{1,2}/[0-9]{1,2}/([0-9]{2}|[0-9]{4})" >/dev/null

}

function check\_date {

day=$1

month=$2

year=$3

months\_31="1 3 5 7 8 10 12"

months\_30="4 6 9 11"

if [ $day -eq 0 ] ; then

return 1

fi

if [ $day -le 31 ] && check\_include $month $months\_31 ; then

return 0

fi

if [ $day -le 30 ] && check\_include $month $months\_30 ; then

return 0

fi

if [ $day -le 28 ] && [ $month -eq 2 ]; then

return 0

fi

if [ $day -eq 29 ] && [ $month -eq 2 ] && check\_leap $year; then

return 0

fi

return 1

}

function check\_include {

head=$1

shift

tail=$@

for number in $tail ; do

if [ "$head" -eq "$number" ] ; then

return 0

fi

done

return 1

}

function check\_gt {

date1=$1

date2=$2

test "$date1" -lt "$date2"

}

function filter {

command=$\*

while read x ; do

if $command $x ; then

echo $x

fi

done

}

function map {

command=$\*

while read x ; do

$command $x

done

}

function not {

$\* # ejecuto lo pasado como comando

result=$? # guardo el resultado de la ejecución

if [ $result -eq 0 ] ; then

return 1

else

return 0

fi

}

function get\_params {

for param in $@ ; do

echo $param

done

}

function format\_date {

day=$(echo $1 + 0 | bc)

month=$(echo $2 + 0 | bc)

year=$3

if [ "$year" -lt 100 ]; then

year=20$year

fi

if [ "$day" -lt 10 ]; then

day=0$day

fi

if [ "$month" -lt 10 ]; then

month=0$month

fi

echo $year$month$day

}

function check\_leap {

year=$1

if [ $(echo $year % 4 | bc) -eq 0 ] && [ $(echo $year % 100 | bc) -ne 0 ] ||

[ $(echo $year % 400 | bc) -eq 0 ]; then

return 0

else

return 1

fi

}

function diff\_date {

date1=$1

date2=$2

}

function get\_days\_of\_month {

date=$1

month=${date:4:2}

year=${date:0:4}

months\_31="1 3 5 7 8 10 12"

months\_30="4 6 9 11"

if check\_include $month $months\_31 ; then

echo 31

return

fi

if check\_include $month $months\_30 ; then

echo 30

return

fi

if check\_leap $year ; then

echo 29

return

fi

echo 28

}

get\_params $@ |

filter check\_format |

tr '/' ' ' |

filter check\_date |

map format\_date |

filter check\_gt $(date +%+4Y%m%d) |

map get\_days\_of\_month

EJERCICIO 2

#!/bin/bash

function less\_than\_255 {

read field

if [ "$field" -le 255 ] ; then

return 0

else

return 1

fi

}

function filter {

command=$1

while read x ; do

if $command $x ; then

echo $x

fi

done

}

function map {

command=$1

while read x ; do

$command $x

done

}

function check\_field {

field=$1

echo $field | less\_than\_255

}

function check\_ip {

ip=$1

field\_numbers=$(echo $ip | grep -oE "[0-9]+" | filter check\_field | wc -l)

if [ $field\_numbers -ne 4 ] ; then

return 1

fi

}

function format\_field {

field=$1

echo $field+0 | bc

}

function format\_ip {

ip=$1

echo $(echo $ip | grep -oE "[0-9]+" | map format\_field) | tr ' ' .

}

grep -xE "[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+" | filter check\_ip | map format\_ip

EJERCICIO 3 FECHAS

#!/bin/bash

function check\_format {

echo $1 | grep -Exo "[0-9]{1,2}/[0-9]{1,2}/([0-9]{2}|[0-9]{4})"

}

function map {

command=$1

while read x ; do

$command $x

done

}

function get\_params {

for param in $@ ; do

echo $param

done

}

function check\_date {

day=$1

month=$2

year=$3

if [ $day -gt 0 ] && [ $day -le 31 ] &&

[ $month -gt 0 ] && [ $month -le 12 ]; then

if [ $day -le 28 ]; then

echo $1 $2 $3

elif [ $day -eq 31 ] &&

([ $month -eq 1 ] || [ $month -eq 3 ] || [ $month -eq 5 ] ||

[ $month -eq 7 ] || [ $month -eq 8 ] || [ $month -eq 10 ] ||

[ $month -eq 12 ]); then

echo $1 $2 $3

elif [ $day -eq 30 ] && [ $month -ne 2 ]; then

echo $1 $2 $3

elif [ $day -eq 29 ] && [ $month -eq 2 ]; then

if [ $day -eq 29 ] && is\_leap $year; then

echo $1 $2 $3

fi

fi

fi

}

function filter\_bigger {

sysdate=$(date +%F | tr -d '-')

day=$(echo $1 + 0 | bc)

month=$(echo $2 + 0 | bc)

year=$3

if [ $year -lt 100 ]; then

year=20$year

fi

if [ $day -lt 10 ]; then

day=0$day

fi

if [ $month -lt 10 ]; then

month=0$month

fi

fecha=$year$month$day

if [ $fecha -gt $sysdate ]; then

echo $day/$month/$year

fi

}

function is\_leap {

year=$1

if [ $(echo $year % 4 | bc) -eq 0 ] && [ $(echo $year % 100 | bc) -ne 0 ] ||

[ $(echo $year % 400 | bc) -eq 0 ]; then

EJERCICIO 4

userlengths=$(cat users | cut -f1 -d: | xargs -Iuser bash -c 'echo -n user: ; echo -n user | wc -c')

maxlength=$(echo -e "$userlengths" | cut -f2 -d: | sort -n | tail -1)

echo -e "$userlengths" | grep ":$maxlength$" | cut -f1 -d:

EJERCICIO 5

#!/bin/bash

edad=$1

mayoria\_edad=18

if [ $# -ne 1 ] || echo $edad | grep -xvE "[0-9]+" > /dev/null ; then

>&2 echo "uso: kk edad (entero)"

exit 1

fi

if [ $edad -lt $mayoria\_edad ] ; then

echo Eres menor de edad

else

echo Eres mayor de edad

fi

EJERCICIO 6

#!/bin/bash

function check\_format {

if (echo $1 | grep -xvE "[0-9]{2}:[0-9]{2}:[0-9]{2}" >/dev/null) ||

[ $(get\_hour $1) -gt 23 ] ||

[ $(get\_min $1) -gt 59 ] ||

[ $(get\_sec $1) -gt 59 ] ; then

return 1

else

return 0

fi

}

function get\_hour {

echo ${1%%:\*}

}

function get\_min {

echo $1 | cut -d: -f2

}

function get\_sec {

echo ${1##\*:}

}

function transform\_seconds {

echo $(($(get\_hour $1) \* 3600 +

$(get\_min $1) \* 60 + $(get\_sec $1)))

}

function add\_zero {

if [ $1 -lt 10 ]; then

echo -n 0$1

else

echo -n $1

fi

}

function format\_seconds {

hh=$(($1 / 3600))

mm=$((($1 - hh \* 3600)/60))

ss=$(($1 - mm \* 60 - hh \* 3600))

echo "$(add\_zero $hh):$(add\_zero $mm):$(add\_zero $ss)"

}

times=$(

for time in $@; do

if check\_format $time ; then

echo $time

fi

done | sort -n -t: -k1 -k2 -k3

)

earliest=$(echo -e "$times" | head -1)

latest=$(echo -e "$times" | tail -1)

secondsbetween=$(($(transform\_seconds $latest) - $(transform\_seconds $earliest)))

echo "$latest - $earliest = $(format\_seconds $secondsbetween)"

EJERCICIO 7

#!/bin/bash

function check\_format {

if (echo $\* | grep -xvE "[0-9]{2}:[0-9]{2}:[0-9]{2}" >/dev/null) ||

[ $(get\_hour $1) -gt 23 ] ||

[ $(get\_min $1) -gt 59 ] ||

[ $(get\_sec $1) -gt 59 ] ; then

return 1

else

return 0

fi

}

function get\_hour {

echo ${1%%:\*}

}

function get\_min {

echo $1 | cut -d: -f2

}

function get\_sec {

echo ${1##\*:}

}

function transform\_seconds {

echo $(($(get\_hour $1) \* 3600 +

$(get\_min $1) \* 60 + $(get\_sec $1)))

}

function add\_zero {

if [ $1 -lt 10 ]; then

echo -n 0$1

else

echo -n $1

fi

}

function format\_seconds {

hh=$(($1 / 3600))

mm=$((($1 - hh \* 3600)/60))

ss=$(($1 - mm \* 60 - hh \* 3600))

echo "$(add\_zero $hh):$(add\_zero $mm):$(add\_zero $ss)"

}

function valid\_time {

while read time ; do

if check\_format $time ; then

echo $time

fi

done

}

function first\_last {

read first

[ -z "$first" ] && first=00:00:00

while read next ; do

last=$next

done

[ -z "$last" ] && last=$first

echo $first

echo $last

}

function to\_seconds {

while read time ; do

transform\_seconds $time

done

}

function from\_seconds {

while read time ; do

format\_seconds $time

done

}

function diff\_time {

read first

read last

echo $(($last-first))

}

function get\_params {

if [ $# -gt 0 ] ; then

echo $@ | tr ' ' '\n'

fi

if [ -p /dev/stdin ] ; then # está abierta la entrada estándar

while read time ; do

echo $time

done

fi

}

get\_params $@ | valid\_time | sort | first\_last | to\_seconds | diff\_time | from\_seconds

EJERCICIO 8

#!/bin/bash

function check\_format {

if (echo $\* | grep -xvE "[0-9]{2}:[0-9]{2}:[0-9]{2}" >/dev/null) ||

[ $(get\_hour $1) -gt 23 ] ||

[ $(get\_min $1) -gt 59 ] ||

[ $(get\_sec $1) -gt 59 ] ; then

return 1

else

return 0

fi

}

function get\_hour {

echo ${1%%:\*}

}

function get\_min {

echo $1 | cut -d: -f2

}

function get\_sec {

echo ${1##\*:}

}

function transform\_seconds {

echo $(($(get\_hour $1) \* 3600 +

$(get\_min $1) \* 60 + $(get\_sec $1)))

}

function add\_zero {

if [ $1 -lt 10 ]; then

echo -n 0$1

else

echo -n $1

fi

}

function format\_seconds {

hh=$(($1 / 3600))

mm=$((($1 - hh \* 3600)/60))

ss=$(($1 - mm \* 60 - hh \* 3600))

echo "$(add\_zero $hh):$(add\_zero $mm):$(add\_zero $ss)"

}

function first\_last {

read first

[ -z "$first" ] && first=00:00:00

while read next ; do

last=$next

done

[ -z "$last" ] && last=$first

echo $first

echo $last

}

function filter {

command=$1

while read x ; do

if $command $x ; then

echo $x

fi

done

}

function map {

command=$1

while read x ; do

$command $x

done

}

function diff\_time {

read first

read last

echo $(($last-first))

}

function get\_params {

for param in $@ ; do

echo $param

done

if [ ! -t 0 ] ; then # está abierta la entrada estándar por tubería

cat

fi

}

get\_params $@ |

filter check\_format |

sort |

first\_last |

map transform\_seconds |

diff\_time |

map format\_seconds

EJERCICIO 9

cd ~ # vaya a su directorio de trabajo

# Compruebe que está en su directorio de trabajo

pwd # Muestra la ruta del directorio actual

whoami # Muestra el usuario conectado

echo $USER # Muestra el usuario conectado

echo $LOGNAME # Muestra el usuario conectado

cat /etc/passwd # Muestra la configuración de los usuarios. Buscar la línea del usuario conectado y comprobar que el penúltimo campo coincide con lo mostrado por el comando pwd anterior.

ls ../../bin/x???\* # Con trayectoria relativa, liste los archivos de "/bin" que comiencen por "x" y que tengan 3 o más caracteres

ls ../../bin/[aeiou]\*[aeiou] ../../bin/[aeiou] # Con trayectoria relativa, liste los archivos de "/bin" que comiencen y terminen por vocal

mkdir ejercicio # En su directorio de trabajo cree un directorio llamado ejercicio

cp ../../etc/group ejercicio/grupos # Copie con trayectorio relativa el archivo "/etc/group" en el directorio "ejercicio" con el nombre "grupos"

cd ejercicio # Muévase al directorio ejercicio

mkdir -p {2000..2100}/{01..12} # Cree un directorio para cada año entre el 2000 y el 2100 y que cada año tenga un subdirectorio para cada mes del 01 al 12

touch {2000..2100}/{01..12}/{01..31} # Cree un archivo del 01 al 31 que corresponde a los días de cada mes

rm {2000..2100}/{0{2,4,6,9},11}/31 # Borre los días 31 de los meses que no tengan 31 días

touch {2000,2004,2008,2010,....... completar....,2096}/02/29 # Añada los 29 de febrero

ls \*/\*/31 # Mostrar todos los días 31 de todos los meses de 31 días de todos los años

EJERCICIO 10

#!/bin/bash

function check\_format {

if (echo $\* | grep -xvE "[0-9]{2}:[0-9]{2}:[0-9]{2}" >/dev/null) ||

[ $(get\_hour $1) -gt 23 ] ||

[ $(get\_min $1) -gt 59 ] ||

[ $(get\_sec $1) -gt 59 ] ||

[ $(get\_hour $1) -lt 12 ]; then

return 1

else

return 0

fi

}

function get\_hour {

echo ${1%%:\*}

}

function get\_min {

echo $1 | cut -d: -f2

}

function get\_sec {

echo ${1##\*:}

}

function transform\_seconds {

echo $(get\_hour $1) \* 3600 + $(get\_min $1) \* 60 + $(get\_sec $1) | bc

}

function add\_zero {

if [ $1 -lt 10 ]; then

echo -n 0$1

else

echo -n $1

fi

}

function check\_plural {

number=$1

word=$2

echo -n $number $word

if [ $number -ne 1 ]; then

echo -n "s"

fi

}

function format\_seconds {

days=$(($1 / 86400))

hh=$((($1 - days \* 86400) / 3600))

mm=$((($1 - hh \* 3600 - days \* 86400)/60))

ss=$(($1 - mm \* 60 - hh \* 3600 - days \* 86400))

if [ $days -gt 0 ]; then

echo -n "$(check\_plural $days día), "

fi

echo "$(check\_plural $hh hora), $(check\_plural $mm minuto) y $(check\_plural $ss segundo)"

}

function add {

sum=0

while read x ; do

sum=$(($sum+x))

done

echo $sum

}

function filter {

command=$1

while read x ; do

if $command $x ; then

echo $x

fi

done

}

function map {

command=$1

while read x ; do

$command $x

done

}

function get\_params {

for param in $@ ; do

echo $param

done

if [ ! -t 0 ] ; then # está abierta la entrada estándar por tubería

cat

fi

}

get\_params $@ |

filter check\_format |

map transform\_seconds |

add |

map format\_seconds

EJERCICIO 11

function pluralize {

number=$1

word=$2

echo -n $number $word

if [ $number -ne 1 ]; then

echo -n "s"

fi

}

function format\_seconds {

days=$(($1 / 86400))

hh=$((($1 - days \* 86400) / 3600))

mm=$((($1 - hh \* 3600 - days \* 86400)/60))

ss=$(($1 - mm \* 60 - hh \* 3600 - days \* 86400))

if [ $days -gt 0 ]; then

echo -n "$(pluralize $days día), "

fi

echo "$(pluralize $hh hora), $(pluralize $mm minuto) y $(pluralize $ss segundo)"

}

function add {

sum=0

while read x ; do

sum=$(($sum+x))

done

echo $sum

}

function filter {

command=$\*

while read x ; do

if $command $x ; then

echo $x

fi

done

}

function map {

command=$\*

while read x ; do

$command $x

done

}

function get\_params {

for param in $@ ; do

echo $param

done

if [ ! -t 0 ] ; then # está abierta la entrada estándar por tubería

cat

fi

}

get\_params $@ |

filter check\_ge 12:00:00 |

map transform\_seconds |

add |

map format\_seconds