lab 2: timeform

By: Keenan Stewart

CS 340-01

# Code

# #!/bin/bash

# #################################

# # civilian to military time: -n #

# #################################

# function civ2mil {

# #################

# #local variables#

# #################

# chour=${1:0:2}

# cmin=${1:3:5}

# postfix=$2

# mtime=0

# ##################

# #setting the time#

# ##################

# if [[ ($cmin < 0) || ($cmin > 59) ]]; then

# echo "Error!! Please check input and try again."

# exit

# fi

# if [[ (($postfix == "am") || ($postfix == "AM")) && ($chour < 12 ) ]]; then

# mtime="$chour""$cmin"

# elif [[ (($postfix == "am") || ($postfix == "AM")) && ($chour == 12) ]]; then

# mtime="00""$cmin"

# elif [[ (($postfix == "pm") || ($postfix == "PM")) && ($chour < 12) ]]; then

# chour=$(($chour + 12))

# mtime="$chour""$cmin"

# elif [[ (($postfix == "pm") || ($postfix == "PM")) && ($chour == 12 ) ]]; then

# mtime="$chour""$cmin"

# else

# echo "Error!! Please check you input and try again."

# exit

# fi

# ############################

# #printing the military time#

# 35 ############################

# echo "civilian to military: $mtime"

# }

# #################################

# # military to civilian time: -m #

# #################################

# function mil2civ {

# #################

# #local variables#

# #################

# mhour=${1:0:2}

# mmin=${1:2}

# ctime=0

# postfix=0

# ##################

# #setting the time#

# ##################

# if [[ ($mmin < 0) || ($mmin > 59) ]]; then

# echo "Error!! Please check input and try again."

# exit

# fi

# if [[ $mhour == "00" ]]; then

# ctime="12:$mmin"

# postfix="AM"

# elif [[ ($mhour > 0) && ($mhour < 12) ]]; then

# ctime="$mhour:$mmin"

# postfix="AM"

# elif [[ $mhour == 12 ]]; then

# ctime="$mhour:$mmin"

# postfix="PM"

# elif [[ ($mhour > 12) && ($mhour < 24) ]]; then

# mhour=$(($mhour - 12))

# ctime="$mhour:$mmin"

# postfix="PM"

# else

# echo "Error!! Please check you inputs and try again."

# exit

# fi

# ############################

# #printing the civilian time#

# ############################

# echo "military to civilian: $ctime $postfix "

# }

# ###############################

# # one timezone to another: -z #

# ###############################

# function timezone {

# mhour=${1:0:2}

# mmin=${1:2}

# zone1=$2

# zone2=$3

# utc=0

# mtime=0

# ########################

# #turns time to UTC time#

# ########################

# case $zone1 in

# AST) utc=$(($mhour + 4)) ;;

# EST) utc=$(($mhour + 5)) ;;

# CST) utc=$(($mhour + 6)) ;;

# MST) utc=$(($mhour + 7)) ;;

# PST) utc=$(($mhour + 8)) ;;

# AKST) utc=$(($mhour + 9)) ;;

# HST) utc=$(($mhour + 10)) ;;

# SST) utc=$(($mhour + 11)) ;;

# ChST) utc=$(($mhour - 10)) ;;

# \*)

# echo "Error!! Unknown time zone. Please check input."

# exit

# ;;

# esac

# ###############################################

# #changes the UTC time to the desired time zone#

# ###############################################

# case $zone2 in

# AST) mhour=$(($utc - 4)) ;;

# EST) mhour=$(($utc - 5)) ;;

# CST) mhour=$(($utc - 6)) ;;

# MST) mhour=$(($utc - 7)) ;;

# PST) mhour=$(($utc - 8)) ;;

# AKST) mhour=$(($utc - 9)) ;;

# HST) mhour=$(($utc - 10)) ;;

# SST) mhour=$(($utc - 11)) ;;

# ChST) mhour=$(($utc + 10)) ;;

# \*)

# echo "Error!! Unknown time zone. Please check input."

# exit

# ;;

# esac

# ######################################

# #insures that hours is between 0 - 23#

# #insures that minutes is between 0-59#

# ######################################

# if [[ $mhour -gt 23 ]]; then

# mhour=$(($mhour - 24))

# fi

# if [[ ($mmin < 0) || ($mmin > 59) ]]; then

# echo "Error!! Please check your input and try again."

# exit

# fi

# mtime="$mhour""$mmin"

# #######################

# #prints out the answer#

# #######################

# echo "The time from $zone1 to $zone2 is: $mtime $zone2"

# }

# ##########################################

# # system uptime since last recycling: -u #

# ##########################################

# function sysUptime {

# echo "$(uptime)"

# }

# ###############################################

# # this handles the options and function calls #

# ###############################################

# while getopts :hm:n:uz: opt; do

# case "$opt" in

# h)

# echo "This command does four things:

# 1) It converts military time to civilian time using the -m option and the time.

# ex. timeform -m 1600 or timeform -m 0400

# 2) It converts civilian time to military time using the -n option and the time.

# ex. timeform -n 04:00 PM or timeform -n 04:00 pm

# 3) It converts one timezone to another using the -z option.

# ex. timeform -z 1600 EST CST

# 4) It prints out the System Uptime since the last recycle time using the -u option.

# ex. timeform -u"

# ;;

# m)

# mil2civ $2

# ;;

# n)

# civ2mil $2 $3

# ;;

# u)

# sysUptime

# ;;

# z)

# timezone $2 $3 $4

# ;;

# \*)

# echo "ERROR!! Please use a valid option: -m, -n, -u, or -z."

# fortune

# ;;

# esac

# done

# exit 1

# Program Description

# Civilian to Military Time

1. When this function is called it gets passed two parameters.

1. I need to separate the given hours, minutes, and post-fix from each other.
2. I need to use if statements to check if the post-fix is AM or PM.
   1. If AM then I need to just put the hours and minutes back together.
   2. If PM then I need to add 12 to the hours and then put the hours and minutes back together.
3. Then I need to print out the result.

# Military to Civilian Time

1. When this function is called it will pass a parameter
2. I need to separate the given hours and minutes
3. I need to use if statement to check if the time is either between 0-12 or 12-23
   * 1. If hour is equal to 0 then set hour to 12, then put the hours and minutes together and post-fix to AM.
     2. If hour is between 0-12 set post-fix to AM and put hours and minutes together.
     3. If hour is equal to
   1. It will print out the result.

# Time Zone

1. When this function is called it passes three parameters.

2. Then it divides the first parameter into hours and minutes, then sets the second parameter to the time zone you want to convert from and the third parameter as the time zone to convert to.

3. It then uses a case statement to convert from your time zone to utc.

4. It then uses a case statement to convert from utc to the desired time zone.

5. I then check everything and prints out the result.

# System Uptime

1. . When called it prints out the system uptime.

# Main Body

1. When the timeform command is called it first uses get opt and a case statement to determine the option that is chosen.

1. . Then calls the function associated with that option.
2. . Then exits

# Data Dictionary

| Name | Scope | Description |
| --- | --- | --- |
| $1 | Local (civ2mil) | Is the parameter that holds the time |
| $2 | Local (civ2mil) | Is the parameter that holds the AM or PM |
| Chour | Local (civ2mil) | Holds the hours of the time |
| Cmin | Local (civ2mil) | Holds the minutes of the time |
| Postfix | Local (civ2mil) | Holds the AM/PM part |
| Mtime | Local (civ2mil) | Hold the converted time |
| $1 | Local (mil2civ) | Is the parameter that holds the time |
| Mhour | Local (mil2civ) | Holds the hour part of the time |
| Mmin | Local (mil2civ) | Holds the minutes part of the time |
| Ctime | Local (mil2civ) | Holds the converted time |
| Postfix | Local (mil2civ) | Holds the AM/PM part |
| $1 | Local (timezone) | Is the parameter that holds the time |
| $2 | Local (timezone) | Is the parameter that holds the first time zone |
| $3 | Local (timezone) | Is the parameter that holds the second time zone |
| mhour | Local (timezone) | Holds the hours part of the time, will hold the starting hours and ending hours |
| mmin | Local (timezone) | Holds the minutes part of the time |
| zone1 | Local (timezone) | Holds the first time zone to be converted from |
| zone2 | Local (timezone) | Holds the second time zone to be converted to |
| utc | Local (timezone) | Holds the hours after being converted from the first time zone and will be converted to the second time zone |
| mtime | Local (timezone) | Holds converted time |
| $2 | Global | Parameter for the command |
| $3 | Global | Parameter for the command |
| $4 | Global | Parameter for the command |

# Structure Chart