* This file codes bouts as 'sleep' time (e.g. sleep, prolonged removals etc) & codes invalid days *
* It uses ActivPal event files *
* Two output files are generated: "sleep_algorithm" contains bout level data; "sleep times_algorithm" contains times for each episode of sleep/prolonged removal. Both files *
* are saved as .dta and .csv files *
* Author: Danielle Bodicoat (07/10/2015) *
* With thanks to Charlotte Edwardson, Kishan Bakrania, and Lis Winkler for valuable input *
* Disclaimer: This code is provided with the hope that it will be helpful, but without any warranty or implied warranty of fitness for a particular purpose *
* NOTE- the code contains references to some variables not in regular activPAL events files and some minor differences in variable names, as modified files had been used for validation purposes. Only minor changes would be needed for use with files output directly from the activPAL software. *
capture clear all
set more off

** THIS SECTION WILL NEED EDITING FOR EACH SPECIFIC USE **
** USERS DO NOT NEED TO EDIT ANYTHING ELSE IN THIS FILE **

** EDIT: set directory where event files are stored **
** ActivPal event files should be stored as separate csv files in the same folder **
** STATA expects csv files to have a .csv extension in lower case (Code may not work if other extensions are used) **
** THIS CODE WILL OVERWRITE THE EVENT FILES IN THIS FOLDER SO MAKE SURE A COPY IS MADE **
** To start with the folder should contain nothing but the ActivPal event files so if you re-run this code you will need to start by putting the original ActivPal event files into a new folder **
cd "Z:\Desktop Files\temp"

** EDIT: set parameters for defining valid days - Change last value in each row (i.e. the numerical value)**
* Maximum percentage in one activity for a valid day (e.g. 95%)

local prop=95
* Minimum no of steps for a valid day (e.g. 500)
local min_steps=500

* Minimum hours of data for a valid day in seconds (e.g. 36000 seconds = 10 hours)
local min_hrs=36000

** EDIT: tell STATA the location of the unique subject ID in the .csv file name

* Start is the location of the first digit of the subject ID in the filename
* Number is the number of digits in the subject ID

* E.g. for csv files named "NNNLL-AP1133030 16May13 10-00am for 7d 13h 0m Events.csv" where the subject ID is NNNLL then start = 1 and number = 5
* E.g. for csv files named "EVENTS_PLUS_SR_NNNN.csv" where the subject ID is NNNN then start = 16 and number = 4

local start=1
local number=5

**EDIT: tell STATA the name of key variables (change last name in each row)

*Variable where activity code is stored (0=sedentary, 1=standing, 2=stepping)
*E.g. if your activity code variable is called activitycode then change the following line to local activity="activitycode"
local activity="activitycode"

*Variable where length of bout is stored in seconds
local length="intervals"

*Variable where the date and time when the bout started is stored
*E.g. if your date & time field is called time then change the following line to local datetime="time"
local datetime="time"

** EDIT: set parameters for defining sleep/prolonged removals - Change last value in each row (i.e. the numerical value)**

* Minimum bout length for a second long sedentary/standing bout in initial bout identification in seconds (e.g. 18000 seconds = 5 hours)
local min_second=18000

* Check window length in minutes (e.g. 15 minutes)
local check=15

* Minimum length of long sedentary bout in seconds (e.g. 7200 seconds = 2 hours)
local min_long=7200
* Maximum number of steps (e.g. 20)
local max_steps=20
* Minimum length of short sedentary bout in seconds (e.g. 1800 seconds = 30 mins)
local min_short=1800

******************************************************************************
*********
** USERS DO NOT NEED TO EDIT ANY OF THE CODE PAST THIS POINT                **
** NB. Edits can be made if the user wishes to change the functionality of the code but this is only **
** recommended for users who understand STATA coding                              **
******************************************************************************

local i=0
local files : dir . files "*.csv"

foreach f of local files {

***prepare data
drop _all
insheet using "f", double

rename `activity' activitycode
rename `length' intervals
rename `datetime' datetime_tmp

gen steps=cumulativestepcount-cumulativestepcount[_n-1] if _n~1
replace steps=cumulativestepcount if _n=1

gen event=1
replace event=cond(_n==1, 1,
///
cond(activitycode~=activitycode[_n-1], event[_n-1]+1,
///

cond(activitycode==activitycode[_n-1], event[_n-1], .))

collapse (sum) intervals steps (first) activitycode datetime_tmp, by(event)

gen counter=_n

gen id=substr("f", start, number)

foreach v of varlist datetime_tmp {
    capture confirm string variable `v`
    if !_rc {
        gen double datetime=clock(datetime_tmp,"DMYhms")
    } else {
        gen double datetime=round(((datetime_tmp+td(30dec1899))*86400)*1000
    }
}

format datetime %tc

gen date=dofc(datetime)

format datetime %d

gen hour=hh(datetime)

*gen noon til noon variable

sort counter

gen day=1 if _n==1
replace day=cond(hour>=12 & hour[_n-1]<12, day[_n-1]+1, ///
    cond((dofc(datetime)-dofc(datetime[_n-1]))>1,day[_n-1]+1,day[_n-1])) if day=. drop hour event

**** identify sleep time ****

**** NB on 1st and last day noon to noon might not be 24 hours so might not have been to sleep in that time

**** i.e. tagging longest sed/stand bout as sleep might be incorrect
**** to 'fix' this... longest bout needs to be 2+ hours to be classed as sleep

*take longest bout of sedentary/standing as sleep time

gen copyv = activitycode
recode copyv (1=0)
gsort day copyv -intervals
by day: gen sleep=1 if _n==1 & copyv==0 & intervals>=`min_long'

*if a second long sedentary/standing bout of over 5 hours then assume also sleep
replace sleep=1 if intervals>=`min_second' & copyv==0

*check for long bouts of sedentary after 'end' and before 'start' of sleep

local old 0
local d 1
while `d'~=0{
sort counter
gen double sleep_end=datetime+(check'*60*1000)+(intervals*1000) if sleep==1
format sleep_end %tc
replace sleep_end=sleep_end[_n-1] if sleep_end==. & datetime<sleep_end[_n-1] & sleep_end[_n-1]==.
replace sleep=1 if (sleep_end==. & sleep_end[_n+1]==.) & (copyv==0 & intervals>=`min_long')

bysort sleep_end: egen total_end=sum(step) if sleep_end==.

sort counter
replace sleep=1 if (sleep_end==. & sleep_end[_n+1]==.) & total_end<=`max_steps' & ((copyv==0 & intervals>=`min_short')|(copyv[_n+1]==0 & intervals[_n+1]>=`min_short'))
gsort -counter
replace sleep=1 if sleep[_n-1]==1 & sleep_end==.

gen double sleep_start=datetime-(check'*60*1000) if sleep==1 & sleep[_n+1]==.
format sleep_start %tc
replace sleep_start=sleep_start[_n-1] if sleep_start==. & (datetime+(intervals*1000))>sleep_start[_n-1] & sleep_start[_n-1]==.
replace sleep=1 if (sleep_start~= . & sleep_start[_n+1]== .) & (copyv==0 & intervals>=`min_long')

bysort sleep_start: egen total_start=sum(step) if sleep_start~=.

gsort -counter

replace sleep=1 if (sleep_start~= . & sleep_start[_n+1]== .) & total_start<=`max_steps' & ((copyv==0 & intervals>=`min_short') | (copyv[_n+1]==0 & intervals[_n+1]>=`min_short'))

sort counter

replace sleep=1 if sleep[_n-1]==1 & sleep_start~=

replace sleep=1 if sleep[_n-1]==1 & activity[_n-1]==0 & activity==1 & activity[_n+1]==0
replace sleep=1 if sleep[_n-1]==1 & activity[_n-2]==0 & activity[_n-1]==1 & activity==0

gsort -counter

replace sleep=1 if sleep[_n-1]==1 & activity[_n-1]==0 & activity==1 & activity[_n+1]==0
replace sleep=1 if sleep[_n-1]==1 & activity[_n-2]==0 & activity[_n-1]==1 & activity==0

drop sleep_end sleep_start total_end total_start

count if sleep==1
local new `r(N)'
local d=`new'-`old'
di "Last iteration count=`old' This iteration count=`new' Difference=`d"
local old `new'

}

**** identify invalid days ****

gen sittings = intervals if activitycode ==0 & sleep~1

gen standings = intervals if activitycode ==1 & sleep~1

gen steppings = intervals if activitycode==2 & sleep~1

gen wwears = intervals if sleep~1
gen stepsvalid = steps if sleep ~= 1  
gen newintervals = intervals if sleep ~= 1

bysort date: egen total_sed=sum(sittings)  
bysort date: egen total_stand=sum(standings)  
bysort date: egen total_step=sum(steppings)  
bysort date: egen total_wwear =sum(wwears)  
bysort date: egen stepsvalid2 =sum(stepsvalid)  
bysort date: egen total =sum(newintervals)

sort counter

*invalid day if >='prop' in one behaviour

gen prop_sed=(total_sed/total)*100  
gen prop_stand=(total_stand/total)*100  
gen prop_step=(total_step/total)*100

gen invalid_day=1 if (prop_sed>=`prop' & prop_sed~=.)|(prop_stand>=`prop' & prop_stand~=.)|(prop_step>=`prop' & prop_step~=.)

gen str30 invalid_reason="N/A"
replace invalid_reason="High proportion in one behaviour" if invalid_day==1

*invalid day if <`min_steps' steps per day
replace invalid_day=1 if stepsvalid2<`min_steps'
replace invalid_reason="Too few steps" if invalid_day==1 & invalid_reason=="N/A"

*invalid day if <`min_hrs' hours of wear/wake
replace invalid_day=1 if total_wwear<`min_hrs' | total_wwear == .
replace invalid_reason="Too little wear" if invalid_day==1 & invalid_reason=="N/A"

drop sittings standings steppings wwear stepsvalid* total* prop_*
**** export data ****
replace sleep=0 if sleep==.
replace invalid_day=0 if invalid_day==.
outsheet using "f", comma replace

local i=1
}**** Merge all files into one big file

local i=0
cap erase mybigfile.dta
local files : dir . files "*.csv"

foreach f of local files {
drop _all
insheet using "f", double
if `i'>0 append using mybigfile, force

save mybigfile, replace
local i=1
}

sort id counter
rename counter bout_id

save sleep_algorithm.dta, replace
outsheet using sleep_algorithm.csv, replace comma

**** generate sleep and wake times ****

gen double tmp=clock(datetime,"DMYhms")
format tmp %tc
drop datetime date
rename tmp datetime
gen date=dofc(datetime)
format date %d

gen double sleeptime=datetime if sleep==1 & (sleep[_n-1]==. | sleep[_n-1]==0)
gen double waketime=datetime+(intervals*1000) if sleep==1 & (sleep[_n+1]==0 | sleep[_n+1]==.)
format sleeptime waketime %tc

replace sleeptime = sleeptime[_n-1] if sleep==1 & sleeptime[_n-1]==.
replace sleeptime=. if waketime==.
keep if sleeptime==.
keep id sleeptime waketime invalid*

save "sleep times_algorithm.dta", replace

outsheet using "sleep times_algorithm.csv", replace comma