```
#!/usr/bin/python

import argparse
import time

parser = argparse.ArgumentParser(description='Decide when to wake up')
parser.add_argument('--clean','-c',action='store_true',dest='timeonly',
                    default='False',help='only display wake time')
parser.add_argument('--wake','-w',action='store_true',dest='wake',
                    default='False',help='wake at specified time')
parser.add_argument('--hour','-y',action='store',dest='hour',
                    default=-1,help='wake/sleep at this hour')
parser.add_argument('--minute','-m',action='store',dest='minute',
                    default=-1,help='wake/sleep at this minute')
parser.add_argument('--pm',action='store_true',dest='pm',
                    default='False',help='interpret time as pm')

args = parser.parse_args()

def sleepAt(sleepTime):
    # takes 14 minutes to sleep
    # and we need 5 sleep cycles of 90 minutes each
    firstWakeTime = sleepTime + (60*14) + 4*(90*60)
    wakeTime = firstWakeTime + (90*60)
    secondWakeTime = wakeTime + (90*60)

    if args.timeonly == 'False':
        print "\t  ",time.strftime("%I:%M %p", time.localtime(secondWakeTime))
        print "Wake up at",time.strftime("%I:%M %p", time.localtime(wakeTime))
        print "\t  ",time.strftime("%I:%M %p", time.localtime(firstWakeTime))
    else:
        print time.strftime("%I:%M %p", time.localtime(wakeTime))

def wakeAt(wakeTime):
    # takes 14 minutes to sleep
    # and we need 5 sleep cycles of 90 minutes each
    firstSleepTime = wakeTime - (60*14) - 4*(60*90)
    sleepTime = firstSleepTime - (90*60)
    secondSleepTime = sleepTime - (90*60)

    if args.timeonly == 'False':
        print "\t      ",time.strftime("%I:%M %p",
```



```
time.localtime(secondSleepTime))
        print "Go to sleep at",time.strftime("%I:%M %p",
time.localtime(sleepTime))
        print "\t      ",time.strftime("%I:%M %p",
time.localtime(firstSleepTime))
    else:
        print time.strftime("%I:%M %p",time.localtime(sleepTime))

def main():
    now = time.localtime()
    targetDay = time.localtime().tm_mday
    targetHour = time.localtime().tm_hour
    targetMinute = time.localtime().tm_min

    if args.hour >= 0:
        if args.pm == "True":
            hour = int(args.hour) + 12
        else:
            hour = int(args.hour)
        now = time.gmtime()
        if (now.tm_hour + hour) >= 24:
           targetTime.tm_mday = targetTime.tm_mday + 1
        targetHour = hour

    if args.minute >= 0:
        targetMinute = int(args.minute)

    targetTime = (now.tm_year,now.tm_mon, targetDay,
                  targetHour, targetMinute, now.tm_sec,
                  now.tm_wday, now.tm_yday, now.tm_isdst)

    if str(args.wake) == 'True':
        wakeAt(time.mktime(targetTime))
    else:
        sleepAt(time.mktime(targetTime))

if __name__ == "__main__":
    main()
```