import os import sys from termcolor import colored import json import logging from datetime import datetime

import am_config as amc from ampyutils import amutils

author = 'amuls'

def rnxobs_header_info(dTmpRnx: dict, dGNSSs: dict, logger: logging.Logger) -> dict: """ rnxobs_header_info extracts the basic hedaer info from the rinex file and stores it in a JSON structure """ cFuncName = colored(os.path.basename(**file**), 'yellow') + '-' + colored(sys._getframe().f_code.co_name, 'green')

```
# create the CLI command for extracting header information into a JSON structure
args4GFZRNX = [amc.dRTK['bin']['GFZRNX'], '-finp', dTmpRnx['obs'], '-meta', 'bas
logger.info('{func:s}: extracting RINEX observation header'.format(func=cFuncNam
amutils.run_subprocess(sub_proc=args4GFZRNX, logger=logger)

with open(dTmpRnx['obs'] + '.json', 'r') as f:
    dObsHdr = json.load(f)

# collect time info
dTimes = {}
dTimes['DT'] = datetime.strptime(dObsHdr['data']['epoch']['first'][:-1], '%Y %m %
dTimes['date'] = dTimes['DT'].date()
dTimes['doy'] = dTimes['DT'].timetuple().tm_yday
dTimes['year'] = dTimes['DT'].timetuple().tm_year
dTimes['yy'] = dTimes['year'] % 100

# collect info per satellite system
dSatSysts = {}
for _, satsys in enumerate(dObsHdr['file']['satsys']):
    dSatSyst = {}

    dSatSyst['sysfrq'] = dObsHdr['file']['sysfrq'][satsys]
    dSatSyst['systyp'] = dObsHdr['file']['systyp'][satsys]
    dSatSyst['sysobs'] = dObsHdr['file']['sysobs'][satsys]

    dSatSysts[dGNSSs.get(satsys)] = dSatSyst

# store the usefull info
amc.dRTK['rnx'] = {}
amc.dRTK['rnx']['times'] = dTimes
amc.dRTK['rnx']['satsysts'] = dSatSysts

# report information to user
logger.info('{func:s}: RINEX observation basic information'.format(func=cFuncNam

logger.info('{func:s}:     times:'.format(func=cFuncName))
logger.info('{func:s}:           first: {first!s}'.format(first=dObsHdr['data']['
logger.info('{func:s}:            last: {last!s}'.format(last=dObsHdr['data']['ep
logger.info('{func:s}:        interval: {interval:.1f}'.format(interval=float(dOb
logger.info('{func:s}:          DOY/YY: {DOY:03d}/{YY:02d}'.format(DOY=dTimes['do

for _, satsys in enumerate(dObsHdr['file']['satsys']):
```

```
    logger.info('{func:s}:     satellite system: {satsys:s} ({gnss:s})'.format(sa
    logger.info('{func:s}:         frequencies: {freqs!s}'.format(freqs=dObsHdr['
    logger.info('{func:s}:       system types: {systypes!s}'.format(systypes=dOb
    logger.info('{func:s}:        observables: {obs!s}'.format(obs=dObsHdr['file

logger.info('{func:s}: dObsHdr =\n{json!s}'.format(func=cFuncName, json=json.dum

return dObsHdr
```

def rnxobs_statistics(dObsHdr: dict, dTmpRnx: dict, dGNSSs: dict, logger: logging.Logger): “”" rnxobs_statistics gets the statistics of the observations in the RINEX observation file “”" cFuncName = colored(os.path.basename(**file**), ‘yellow’) + ‘-’ + colored(sys._getframe().f_code.co_name, ‘green’)

```
# create the CLI command for getting observation statistics in a temporary file
args4GFZRNX = [amc.dRTK['bin']['GFZRNX'], '-finp', dTmpRnx['obs'], '-stk_obs', '-
logger.info('{func:s}: extracting RINEX observation statistics'.format(func=cFunc
amutils.run_subprocess(sub_proc=args4GFZRNX, logger=logger)

# read in the obsstat file per satellite system
for _, satsys in enumerate(dObsHdr['file']['satsys']):
    pass
```