安装pynmea2

pip install pynmea2

导入：import pynmea2

解析字符串中 NMEA 0183 协议的数据，可以使用 pynmea2.parse(data, check=False) 方法，其中的 check 参数指定是否对消息中的检校字段进行检查。

示例代码演示解析 GGA 数据

>>> line = '$GPGGA,092750.000,5321.6802,N,00630.3372,W,1,8,1.03,61.7,M,55.2,M,,\*76'

>>> record = pynmea2.parse(line)

>>> record

<GGA(timestamp=datetime.time(9, 27, 50), lat='5321.6802', lat\_dir='N', lon='00630.3372', lon\_dir='W', gps\_qual=1, num\_sats='8', horizontal\_dil='1.03', altitude=61.7, altitude\_units='M', geo\_sep='55.2', geo\_sep\_units='M', age\_gps\_data='', ref\_station\_id='')>

解析完成后，可通过属性来访问各个字段

>>> print('GPS Quality Indicator:', record.gps\_qual)

GPS Quality Indicator: 1

>>> print('Horizontal Dilution of Precision:', record.horizontal\_dil)

Horizontal Dilution of Precision: 1.03

>>> print('Latitude:', record.lat)

Latitude: 5321.6802

>>> print('Latitude Direction:', record.lat\_dir)

Latitude Direction: N

>>> print('Number of Satellites:', record.num\_sats)

Number of Satellites: 8

>>> record.timestamp

datetime.time(9, 27, 50)

除了按照协议约定格式对数据进行解析之外，pynmea2.parse() 函数还做了一些必要的数据转换工作，将经纬度坐标转换为 Python 中的 float 类型：

>>> print('Latitude:', record.latitude)

Latitude: 53.361336666666666

>>> print('Longitude:', record.longitude)

Longitude: -6.50562

此外，解析方法还为输出结果添加了额外的属性：latitude\_minutes，latitude\_seconds，longitude\_minutes 和 longitude\_seconds，它们存储了大地坐标的分、秒各部分。因此你可以方便地对坐标进行格式化输出：

>>> print('Latitude: {:02d}°{:02d}′{:07.4f}″'.format(int(record.latitude), int(record.latitude\_minutes), record.latitude\_seconds))

Latitude: 53°21′40.8120″

>>> print('Longitude: {:02d}°{:02d}′{:07.4f}″'.format(int(record.longitude), int(record.longitude\_minutes), record.longitude\_seconds))

Longitude: -6°30′20.2320″

>>> print('Altitude: {:.3f}'.format(record.altitude))

Altitude: 61.700

pynmea2 还能够直接处理 NMEA 0183 协议的数据流，你可以使用 pynmea2.NMEAStreamReader 类来解析数据流：

>>> streamreader = pynmea2.NMEAStreamReader(input\_stream)

>>> while 1:

... for record in streamreader.next():

... print(repr(record))

...

<GSV(num\_messages='3', msg\_num='1', num\_sv\_in\_view='11', sv\_prn\_num\_1='10', elevation\_deg\_1='63', azimuth\_1='137', snr\_1='17', sv\_prn\_num\_2='07', elevation\_deg\_2='61', azimuth\_2='098', snr\_2='15', sv\_prn\_num\_3='05', elevation\_deg\_3='59', azimuth\_3='290', snr\_3='20', sv\_prn\_num\_4='08', elevation\_deg\_4='54', azimuth\_4='157', snr\_4='30')>

<VTG(true\_track=89.68, true\_track\_sym='T', mag\_track=None, mag\_track\_sym='M', spd\_over\_grnd\_kts=Decimal('0.00'), spd\_over\_grnd\_kts\_sym='N', spd\_over\_grnd\_kmph=0.0, spd\_over\_grnd\_kmph\_sym='K')>

<GLL(lat='4250.5589', lat\_dir='S', lon='14718.5084', lon\_dir='E', timestamp=datetime.time(9, 22, 4, 999000), status='A')>

...

pynmea2 在数据解析失败时会抛出 pynmea2.nmea.ParseError 异常，在需要的时候，你可以捕获并处理它。示例：

>>> try:

... gga\_msg = pynmea2.parse('at^sisw=1,1500,0,0')

... except pynmea2.nmea.ParseError:

... print('Warning: a line parsing failed!')

...

Warning: a line parsing failed!