```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import xarray as xr
import datetime
from dateutil.relativedelta import relativedelta

import cartopy.crs as ccrs
import cartopy.feature as cfeature
import matplotlib.ticker as mticker
import matplotlib.style as mplstyle
from cartopy.mpl.ticker import LongitudeFormatter, LatitudeFormatter

import gsw

year_bgn,month_bgn=2021,1  # 解析対象年月 はじめ
year_end,month_end=2024,12 # 解析対象年月 おわり

dt    =datetime.datetime(year_bgn,month_bgn,15) # dt にはじめの年月日を代入
dt_end=datetime.datetime(year_end,month_end,15) # dt_end におわりの年月日を代入

dlat=gsw.distance([0.,0.],[0.,1.]) # 緯度１度の距離を取得（あとで使う）

OHC_time=[] # OHC全量の時間変化を記録するための配列　最初は要素０としておく
time=[] # OHC全量を図示する際の横軸


while True: # 月でループ

    # read MOAA GPV data
    # データを読み込む

    date = ???
    dtyp = ???
    lat = ???
    lon = ???
    prs = ???
    toi = ???
    soi = ???
    ncr.close()

    # 配列形状を fortran 風に変更
    toi = toi.T
    soi = soi.T

    im = int(lon.shape[0]) # 東西格子数
    jm = int(lat.shape[0]) # 南北格子数
    km = int(prs.shape[0]) # 鉛直格子数

    mask = np.ones([im,jm,km]) # 欠損値処理のためのマスク配列　１を代入
    mask[toi.values < -100.]=0. # 欠損値の格子には０を代入
    mask[soi.values < -100.]=0. # 欠損値の格子には０を代入

    # 鉛直方向の格子点間の距離（圧力差）を求める
    prs_u = np.zeros(km)
    for k in range(km-1):
        prs_u[k+1]=0.5*(prs[k]+prs[k+1])

    DZ = np.zeros([im,jm,km])
    for k in range(km-1):
        DZ[:,:,k]=prs_u[k+1]-prs_u[k]
    DZ[:,:,km-1]=50. # 一番下

    # ループを避けるために３次元配列を用意する
    LON_3D = np.zeros([im,jm,km])
    for i in range(im):
```

```
67          LON_3D[i,:,:]=lon.values[i]
68      LAT_3D = np.zeros([im,jm,km])
69      for j in range(jm):
70          LAT_3D[:,j,:]=lat.values[j]
71      PRS_3D = np.zeros([im,jm,km])
72      for k in range(km):
73          PRS_3D[:,:,k]=prs.values[k]
74
75      # 絶対塩分、保存温度、エンタルピー、密度の計算
76      SA = gsw.SA_from_SP(soi.values,PRS_3D,LON_3D,LAT_3D)
77      CT = gsw.CT_from_t(SA,toi.values,PRS_3D)
78      ET = gsw.enthalpy(SA,CT,PRS_3D)
79      RO = gsw.rho(SA,CT,PRS_3D)
80
81      # 欠損値を代入
82      SA[mask == 0.]=-10000.
83      CT[mask == 0.]=-10000.
84      ET[mask == 0.]=-10000.
85      RO[mask == 0.]=-10000.
86
87      # OHC を計算
88      OHC = ??? # lon,lat の２次元配列
89      OHC_tot= ???# 面積をかけて足し合わせ
90      OHC_time=np.append(OHC_time,OHC_tot)
91      time = np.append(time,dt) # 年月を代入
92
93
94      X,Y = np.meshgrid(lon, lat, indexing = 'ij')  # indexing = 'ij' is F style
95
96      # 解析対象年月のOHCの図示
97      fig, figa = ???
98
99      gl = figa.gridlines(crs=ccrs.PlateCarree())
100     gl.xlocator = mticker.FixedLocator(np.arange(-180,180.1,30))
101     gl.ylocator = mticker.FixedLocator(np.arange(-75,75.1,15))
102     #figa.set_xticks(np.arange( 0,360.1,30),crs=ccrs.PlateCarree())
103     #figa.set_yticks(np.arange(-80,80.1,10),crs=ccrs.PlateCarree())
104
105     data=???
106     cf=figa.contourf(X,Y,data, ???
107
108     fig.show()
109     fig.savefig('../JPG/OHC'+dt.strftime('%Y%m')+'.jpg')
110     plt.close()
111
112     if dt == dt_end : # 解析終了年月だったら終了
113         break
114     dt=dt+relativedelta(months=1) # 解析対象年月を１月先に進める
115
116 # OHC の時間変化を図示
117 figa.plot(time,OHC_time)
118 figa.xaxis.set_major_formatter(mdates.DateFormatter("%Y/%m"))
119 fig.show()
120 fig.savefig('../JPG/OHCtime.jpg')
121
```