プログラミング基礎II 第2回レポート課題
モジュールとパッケージ・クラス

レポート提出期限：2026年1月27日
提出先：学務情報システム

課題1：モジュール・パッケージ（ソーティング）

下記の入出力例のように，複数の実数値を読み込んで**降順に**並べ替えて表示するプログラムを作成せよ．ただし，以下の条件を満たすこと．

1. 以下の3つのソートアルゴリズムをモジュールとして実装すること．ソートのアルゴリズムは講義HPのスライドを参考に作成すること．
   ・交換ソート（bubble sort）: `bubble_sort.py`
   ・選択ソート (selection sort) : `selection_sort.py`
   ・挿入ソート（insertion sort) : `insertion_sort.py`

2. 各モジュールには，リスト型のdataを引数として受け取り，降順にソートした結果を戻り値として返却する以下のユーザ定義関数を作成すること．

   `def descending_sort (data)`

3. 以下の入出力例が再現されるようにプログラム (`sort_test.py`) を作成せよ．ただし，各モジュールを `prog_sort` フォルダにまとめてパッケージとして扱うこと．

   prog_sort パッケージ

   ```
   prog_sort/
     |-- __init__.py
     |-- bubble_sort.py
     |-- selection_sort.py
     `-- insertion_sort.py
   ```

4. 提出物

   - パッケージ内の `bubble_sort.py`，`selection_sort.py`，`insertion_sort.py` 及び `sort_test.py` のソースコード（py ファイル）
     ※ ファイルは複数になってもよい．複数ファイルは tar でまとめてもよい．
   - プログラムの内容・動作の解説及び実行結果の画面キャプチャを含む文書（PDF ファイル：`2nd_rep_kadai1.pdf`）

入出力例

```
10個の実数を入力して下さい.
data[0]:-12.34 ↵
data[1]:0 ↵
data[2]:-3.0 ↵
data[3]:1.2 ↵
data[4]:4 ↵
data[5]:5.89 ↵
data[6]:-13.01 ↵
data[7]:0.234 ↵
data[8]:-0.21 ↵
data[9]:1 ↵

ソート方法選択[1:交換ソート, 2:選択ソート, 3:挿入ソート]:1
降順にソートしました.
data[0]:5.89
data[1]:4.0
data[2]:1.2
data[3]:1.0
data[4]:0.234
data[5]:0.0
data[6]:-0.21
data[7]:-3.0
data[8]:-12.34
data[9]:-13.01
```

課題2：クラス

ソースコード 1 に実数型の実部と虚部をメンバとする複素数クラスの記述例を提示する．ここで，実部と虚部のアクセサ（ゲッタ，セッタ），複素数の文字列化と画面表示に関わるメソッドが実装されている．その動作を入出力例に示す．以下の指示に従ってプログラムを作成すること．

ソースコード 1: Cpx.py

```
import math

class Cpx:
    def __init__(self, re=0., im=0.):
        self._re = float(re)
        self._im = float(im)

    @property     # ゲッタ
    def re(self):
        return self._re
    @re.setter    # セッタ
    def re(self, value):
        self._re = value

    @property     # ゲッタ
```

```
    def im(self):
        return self._im
    @im.setter    # セッタ
    def im(self, value):
        self._im = value

    def __repr__(self):
        return self.__str__()

    def __str__(self):
        str_  = ''  if self._re == 0 else str(self._re)
        str_ += ''  if self._im <  0 else '+'
        str_ += str(self._im) + 'i'
        return str_
```

入出力例

```
a = Cpx(1,2)↵

b = Cpx(3,-3)↵

print(a.re)↵
1.0

print(a.im)↵
2.0

b.real = -10↵

b.imag = 0.1↵

print(b)↵
-10.0-0.1i
```

1. このクラス型を以下のように拡張したプログラムを作成すること（ファイル名：Cpx2.py）.

   (a) 複素数の複素共役，絶対値，偏角，極座標（絶対値，偏角）を返すメソッドの追加

   ```
   def conj(self) -> Cpx
   def abs(self) -> float
   def phase(self) -> float
   def polar(self) -> tuple
   ```

   (b) 複素数同士の ==，+，-，*，/演算を行うためのメソッドの追加

   ```
   def __eq__(self, other)
   def __add__(self, other)
   def __sub__(self, other)
   def __mul__(self, other)
   def __truediv__(self, other)
   ```

2. 提出物

- Cpx2.py のソースコード（py ファイル）
  ※ ファイルは複数になってもよい．複数ファイルは tar でまとめてもよい．
- プログラムの内容・動作の解説及び実行結果の画面キャプチャを含む文書（PDF ファイル：2nd_rep_kadai2.pdf）

入出力例

```
u=Cpx(1.23, -6.2) ↵

v=Cpx(4,5) ↵

u ↵
1.23-6.2i

v ↵
4.0+5.0i

u.conj() ↵
1.23+6.2i

u.polar() ↵
(6.320830641616654, -1.374952115374697)

v.polar() ↵
(6.4031242374328485, 0.8960553845713439)

u==v ↵
False

u+v ↵
5.23-1.2000000000000002i

u-v ↵
-2.77-11.2i

u*v ↵
35.92-18.65i

u/v ↵
-0.6360975609756098-0.7548780487804879i
```