ディジタルメディア処理 2025

締め切り：
scombz参照

雛形:
こちら (http://takashiijiri.com/classes/dm/) に雛形があるので、必ず利用してください
雛形のファイル名は変更しないでください（課題XXのファイル名はexerXX.pyのまま）

解答に関する注意:

- 各課題では、入出力の詳細な仕様が指定されます。従ってください。
- 採点自動化のため、フォルダ名・ファイル名が間違っているもの、入出力の仕様を満たさないコードは、評価できず0点扱いとなります。
- この課題では計算速度を重視しませんが、評価用入力データに対して極端に長い時間のかかるもの（20秒以上）は、自動採点の都合上0点とします。
- 各課題について、入出力例を用意するので作成したプログラムのテストに利用してください。ただし、評価には配布したものとは別のデータを使います。
- この課題の解答となるコードを、この課題の解答と分かる形でWeb上（GitHub、SNS、個人web page)に公開する事は避けてください。
- コードのコピーが発見された場合には、コピー元・コピー先の両名ともカンニングと同様の処置をとります。※Pythonはコードがどうしても似てしまうことは理解しています。カンニングの疑いをかけられないかと不安になったり、あえて”へんな書き方”をする必要はないです。
- 知恵袋やteratailなどのナリッジコミュニティサイトにて、問題文を提示し、解答を得る事は行なわないでください。上記のような活動を発見した場合は、しかるべき処置をとります。
- 本課題は自身で考えてコードを書いてもらうことを目的とするため、この課題の回答のためにchatGPTやCopilot等のLLMは利用しないでください。

準備

Step1 : http://takashiijiri.com/classes/dm/より、下記のファイルを取得してください。

- python_install.pdf
- dm_template.zip

Step2 : python_install.pdf に従い Pythonをインストールしてください。

Step3 : template.zipを解凍してください（フォルダをダブルクリック > 全て展開 を選択）

Step4 : コマンドプロンプトを開き、解凍してできたtemplate フォルダのトップをカレントにしてください。（フォルダを開いて アドレスバーに「cmd」と入れてもOK）

Step5 : 下記のコマンドを実行し、テンプレートを初期化してください。

```
>python dm_starter.py
```

Step6: 初期化キーの提出 : 実行に成功すると、

```
>python dm_starter.py
initialize successXXXXXXXXXX.XXXXXXXXX
```

と出力されます。この出力された一行『`initialize successXXXXXXXXXX.XXXXXXXXX`』をscombzの『テスト : 課題準備』より提出してください。「XXXXXXXXXX.XXXXXXXXX」には、数値が入ります。この数値を不正行為防止に利用するため、ここで初期化した雛形を必ず利用して解答して下さい（ファイル紛失などの際はご連絡ください）。提出がない場合は提出したファイルを採点対象としないか、または、減点対象とします。

提出方法:

1. 初期化した dm_template フォルダ内の dayN というフォルダに解答を記述してください
   ※1日目なら day1というフォルダです
2. 作業した dayNというフォルダをzip圧縮してください 「右クリック > 送る > 圧縮（zip）」
3. 出来上がった zipファイルをscombzより提出してください

※ 提出ファイル名は dayN.zip となるはずです。

※ 発展課題は、ファイル提出だけでなく教員（またはTA）に確認を受けてください。

目次

day1

- 01. hello world ★
- 02. hogefuga ★
- 03. ファイル入力と配列 ★
- 04. 画像のグレースケール化 ★★
- 05. 迷路（発展） ★★★★

day2

- 06. 画像の二値化 ★★
- 07. 色の変換 ★★
- 08. モザイク画像の作成 ★★★
- 09. ガウシアンフィルタ ★★
- 10. 素数分布画像（発展） ★★★

day3

- 11. ソーベルフィルタ ★★
- 12. 勾配強度画像 ★★★
- 13. Medianフィルタ ★★
- 14. ランレングス符号化 ★★★
- 15. ハフマン符号化（発展） ★★★★

day4

- 16. Bilateralフィルタ ★
- 17. ガンマ変換 ★★
- 18. ハーフトーン 1 ★★
- 19. ハーフトーン 2 ★★★
- 20. ハーフトーン 3（発展） ★★★

day5

- 21. 離散フーリエ変換 ★★
- 22. 逆離散フーリエ変換 ★★
- 23. 2次元フーリエ変換 ★★★★
- 24. Deconvolution（発展） ★★★★★

易 ★
普通 ★★
やや難 ★★★
難 ★★★★
※ 雛形に多くのヒントがあります。まずは雛形にあるコードを読んでください。

課題01 hello, world

2つの整数を受け取り、その積の回数だけ『hello, world』と標準出力に表示するプログラムを作成せよ。実行コマンドは以下の通り。

```
python exer01.py  a  b
```

仕様

- 非負整数値 aとbをコマンドライン引数より受け取る
- a x b 回『hello, world』と出力する

※ すべての課題にひな形 exerXX.pyが用意してあるので参考にしてください。

※ 毎年数名スペルミスをする方がいます。カンマは半角で、その後に半角スペースが一つです．ご注意を。

入出力例 ―----------

```
> python exer01.py 2 3
hello, world
hello, world
hello, world
hello, world
hello, world
hello, world
```

a=2, b=3なら、6回『hello, world』と出力します。

※ すべての課題に対して，正解プログラムが出力する例を提供します

※ 正解画像ファイルなどは雛形のフォルダ内に入れておきます

※ 自身で作成したプログラムの出力と見比べることでデバッグに利用してください

※ 正解ファイルを上書きするとデバッグが難しくなるので注意

課題02 hogefuga :

非負整数値Nをコマンドラインから受け取り、下記のルールに従い1からNまでの整数を順番に標準出力に表示するプログラムを作成せよ。実行コマンドは以下の通り。

```
python exer02.py N
```

仕様 :

- 入力Nはコマンドライン引数として受け取る
- 1からNまでの整数を順番に標準出力に表示する。ただし、以下のルールに従う。
  - 表示する数字が『3の倍数』かつ『5の倍数でない』時には，数字ではなく ”hoge” と表示する
  - 表示する数字が『5の倍数』かつ『3の倍数でない』時には，数字ではなく”fuga” と表示する
  - 表示する数字が『3の倍数』かつ『5の倍数』の時には，数字ではなく”hogefuga”と表示する

入出力例 ―----------

N=16なら下記のような出力になります。

```
> python exer02.py 16
1
2
hoge
4
fuga
hoge
7
8
hoge
fuga
11
hoge
13
14
hogefuga
16
```

※ hogeとfugaのスペルが違う方が例年数名いるので注意してください（本質的な間違いではないのですが仕様は仕様なので。。）。

課題03 ファイル入力と配列

数値データをファイルから読み込み、その最大値と平均値、および、１番目・２番目・３番目に小さい値をファイルに出力するプログラムを作成せよ。実行コマンドは以下の通り 。

```
python exer03.py file_in.txt file_out.txt
```

仕様

- 入出力ファイル名（file_in.txt、file_out.txt）は，コマンドライン引数として取得
- 入力ファイルには１行に１つの実数値が記載されている
- 出力ファイルの 1行目に，最大値，平均値 を記載する
- 出力ファイルの 2行目に，1番目・2番目・3番目に小さい値 を記載する
- 数値の間には半角スペースを一つだけ書くこととする

※ ファイル入出力については雛形を参照

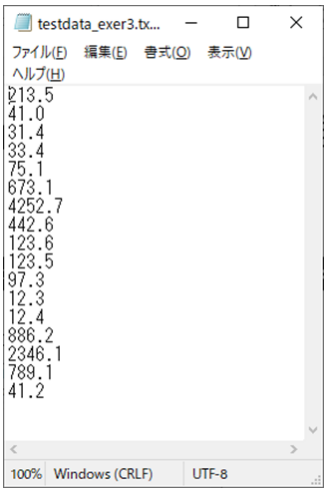
入出力例 ―----------

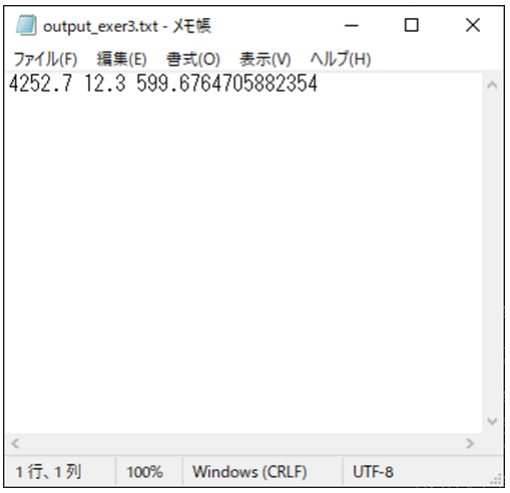
day1/data フォルダ内に入出力例があります。下記のコマンドにより入力データ『data/testdata.txt』の処理を行うと、出力データ『./output_exer3.txt』が得られます。

```
python exer03.py data/testdata.txt ./output_exer3.txt
```

※ dataフォルダの data/testdata.txt を読み込み、./output_exer3.txt に書き出すコマンド

※正解データへの上書き防止のため、カレントへ出力するコマンドを提示します。





入力例　　出力例

※ 実行環境・利用するライブラリに依存して多少の誤差が出る場合がありますが、小さな誤差であれば正解扱いとするのでピッタリ同じになる必要はありません。

課題04. 画像のグレースケール化

画像データを読み込み、グレースケール化して保存するプログラムを作成せよ。実行コマンドは以下の通り。

```
> python exer04.py file_in.png file_out.png
```

仕様:

- 入出力ファイル名（file_in.png, file_out.png）は，コマンドライン引数として取得
- グレースケール値は、赤・緑・青チャンネルの平均を整数キャストしたものとする

```
グレースケール値　= (int)(赤 + 緑 + 青)/3
```

入出力例 ―----------

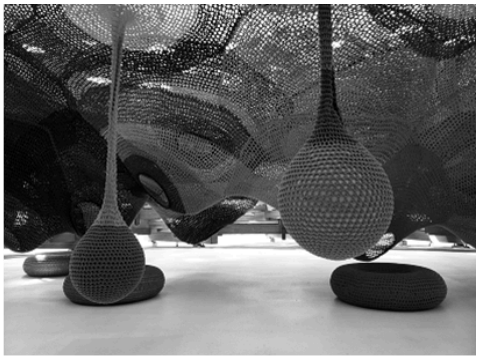
入力・出力画像はdataフォルダ内にあります。以下のコマンドを実行すると、dataフォルダ内のdata/img.png がグレースケール化されたファイル ./img_gray.png が出力されます。

```
> python exer04.py data/img.png ./img_gray.png
```

※正解データの上書き防止のため、カレントに出力するコマンドを提示しています。

※ dataフォルダ内に正解データがあるので自身で出力結果を確認してください。

img.png
入力



img_gray.png
出力

課題05: 迷路 (発展)

迷路を解き、スタートからゴールまでの経路が有る場合は最小歩数を表示し、経路が無い場合は-1を表示するプログラムを作成せよ。実行コマンドは以下の通り。

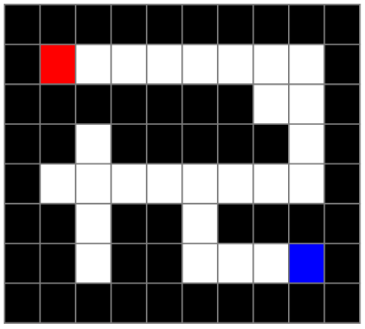
```
python exer05.py meiro.bmp
```

仕様

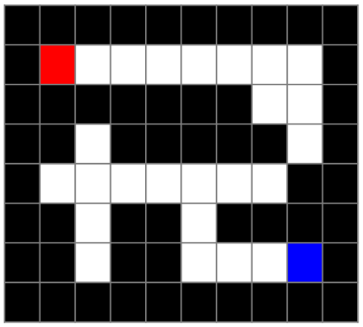
- 迷路の仕様は以下の通り
  - 迷路は画像で与えられる
  - 黒画素 (r,g,b) = (0, 0, 0)は壁で通れない
  - 白画素 (r,g,b) = (255,255,255)は通れる通路
  - 赤画素 (r,g,b) = (255, 0, 0)はスタート
  - 青画素 (r,g,b) = (0, 0,255)はゴール
  - ある白画素から一歩で，上下左右の白画素に移動できる
- 画像ファイル名はコマンドライン引数より取得する
- 計算結果は標準出力に表示する

※ 再帰関数は利用しないでください。コールスタックの深さに限界があるので再帰関数を利用して実装すると大きな画像に対しては動かない可能性があります

※ dataディレクトリ内に迷路画像サンプルがあります



正解は18



正解は－1

※ 発展課題は、講義中に教員 or TAに見せ、チェックを受けてください。