ソフトウェア演習Ⅲ〔課題２：NumPy 配列と関数例〕青野雅樹

以下の問題に対する Python プログラムを作成し，プログラム（kadai2.py）と実行結果（kadai2.txt）をつけ、全体を ZIP にまとめて Moodle にアップロードせよ。締め切りは 10 月 25 日（火）の夜までとする。画像の NumPy ファイルは ZIP に入れなくてよい。

(1) COCO と呼ばれるカラー画像データセットの一部を NumPy 配列として用意している。場所は、https://www.kde.cs.tut.ac.jp/~aono/data/Npy/ 以下にある。ファイル名は XY.npy で、X と Y は数字になっている。学籍番号の末尾と Y が一致するものが４ファイルあるので（ローカルに）ダウンロードせよ。

(2) 関数 **average_RGB** を作成せよ。以下の引数を有する。第一引数は画像を表す NumPy 配列(ndarray)、第二、第三引数は、画像の開始位置（左上 X 座標(x0)と Y 座標(y0)）、第四、第五引数は、画像の縦幅(h)と横幅(w)。NumPy 配列として与えられた第一引数のカラー画像の、開始位置(x0,y0)から縦幅 h, 横幅 w の範囲の(部分)画像の(R,G,B)値の平均値を計算し、これを NumPy 配列（3 次元の(R,G,B)値の NumPy 配列））にして返す。

(3) 次頁の図に示すように、画像全体、ならびに 5 つの部分領域に対して(2)で作成した **average_RGB** 関数を適用し、得られた平均の(R,G,B)値を並べて、NumPy 配列を作成する。具体的には、全体の(R,G,B)の平均値を avg_all と表し、5 つの部分領域から得られる平均値をそれぞれ、avg_LT, avg_RT, avg_LB, avg_RB, avg_CT と表記することとする。アンダースコア以降の略号の意味は、図を参照のこと。このとき、全体の平均の(R,G,B)値と 5 つの領域の平均の(R,G,B)値の合計 6 つの(R,G,B)値から、NumPy 配列を np.array([avg_all, avg_LT, avg_RT, avg_LB, avg_RB, avg_CT])として作成せよ。

(4) 次に、(3)で作成した NumPy 配列を引数とし、全体(avg_all)と、平均色がもっとも似ていた 1～5 の領域のインデックスと、もっとも似ていなかった領域のインデックスをタップルで返す**関数 FindMinMax** を作成せよ。似ているかどうかの判断は(R,G,B)空間を 3 次元空間と解釈し、3 次元空間でのユークリッド距離で判断せよ。すなわち、距離が近ければ似ていると判断せよ。

(5) main 部分（if __name__ == "__main__":）から、ファイル名を指定して、NumPy ファイルを読み込み、まずファイル名とシェープ（サイズ）を書き出せ。次に、(3)（average_RGB を 6 回呼出す操作）と(4)(FindMinMax)を順次実行し、全体の平均色をプリントしたあと、全体の平均色ともっとも似ていた領域（の名前と平均色）ともっとも似ていなかった領域（の名前と平均色）をプリントせよ。ファイルがないときの例外処理（「指定したファイルがないことを表示し、終了」）を忘れずに。

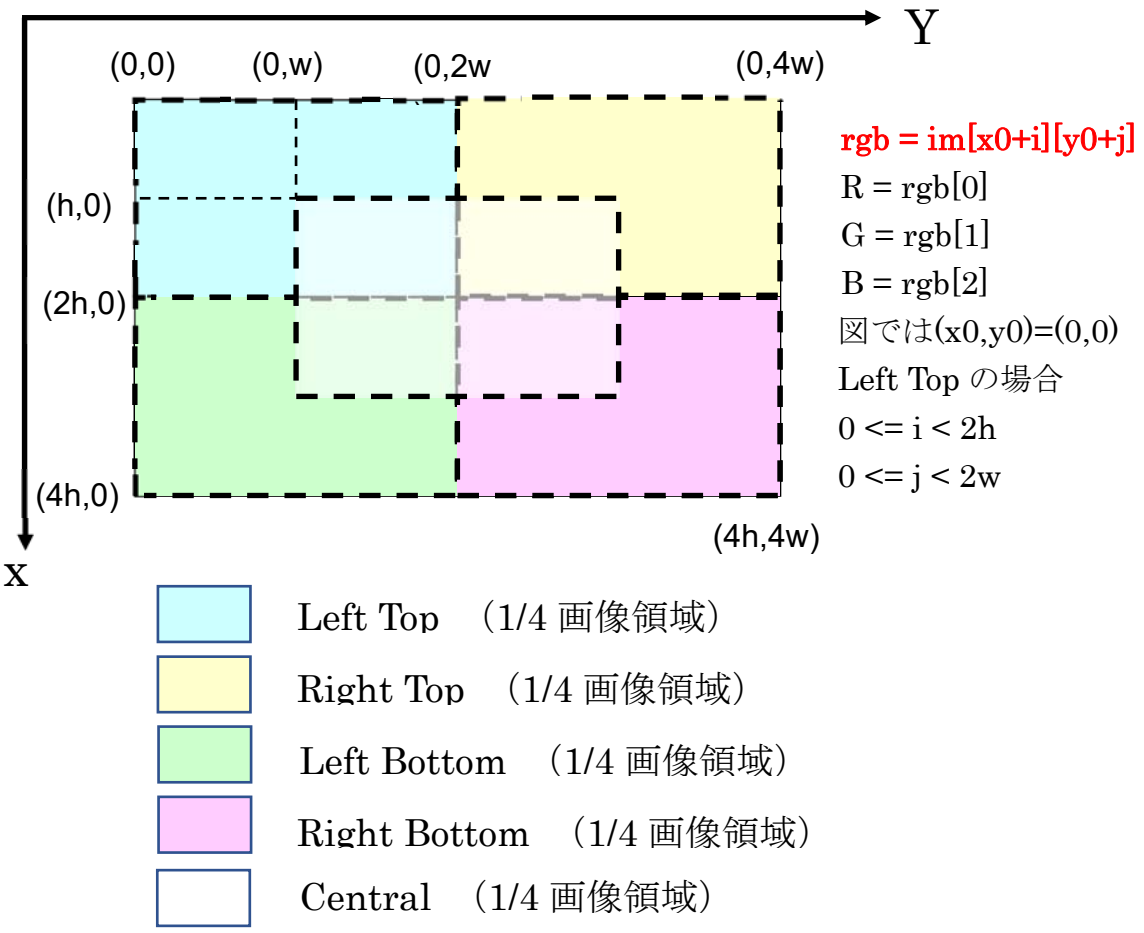
(6) (5)の操作をダウンロードした 4 つのファイルから適当に選んだ 2 つのファイルに対して実行せよ。（> python kadai2.py [ファイル名] を 2 回実行せよ）

注：COCO は https://cocodataset.org/#home にあります。

コメントとヒント： ５つの部分領域は以下の図の通りです。ただし、全体の画像サイズを(4h, 4w)と仮定しています。**画素をアクセスする場合、Y->X の順に並んでいます。NumPy 画像ファイルのシェープは、縦方向、横方向の順になっていますので注意してください。**

(4)の FindMinMax では、課題に書いているように、(R,G,B)値ごとに、**ユークリッド距離**で似ているかどうかを判断してください。(5)の領域名は図に示すような 5 種類の名前（例 Left Top 領域）を使ってください。図では、画像サイズを(4h,4w) (h = height, w = width)としていますが、実際用意した画像は、サイズは異なるものがありますが、横方向の画素数も縦方向の画素数も 4 の倍数となっています。

ユークリッド距離計算の関数(たとえば def Euclid(a,b):)では、2 引数（いずれも NumPy 配列）をもたせますが、2 つの引数のタイプ、ならびにシェープ（(R,G,B)値のため 3 次元のはず）の一致を最初に確認するようにしてください。引数のタイプが NumPy の ndarray であることは、以下のような isinstance 関数で調べられます。




```
if (not isinstance(a, np.ndarray)):# import numpy as np を前提
```

第一引数（ここでは a）が NumPy 配列でないときの処理

出力例を以下に示します。（ただし、以下は一回実行した例です。課題では 2 回実行するので、２回目は > python kadai2.py ZY.npy >> kadai1.txt のように末尾に append するように実行ください。あるいは、2 回、別々に実行した出力を適宜、足し合わせて編集してください）。(R,G,B)のプリントでは、以下の出力例では、rgb に色があるとして

print(f"その平均色 = ({rgb[0]:.3f},{rgb[1]:.3f},{rgb[2]:.3f}")としています。小数点以下は、上例のように 2～3 桁程度のプリントで結構です。

【出力例】

```
**********************************************
青野雅樹, 01162069
日付: 2022-10-15 16:01:33.359859
内容：NumPy 配列と関数（画像 NumPy ファイルの処理）
**********************************************
ファイル名：Npy/sample.npy
サイズ：高さ= 372, 幅 = 480
全体の平均色 = (163.01, 150.80, 131.77
平均色がもっとも全体に近い領域は、Right Top 領域
その平均色 = (152.129,149.803,124.604
平均色がもっとも全体と異なる領域は、Left Top 領域
その平均色 = (139.181,120.231,110.914
```

今回は、画像を表示する必要はありませんが、どのような画像か見てみたい人は、たとえば以下のようなコードで表示可能です。（Matplotlib のいろいろな描画例は第六回あたりの授業の資料で紹介する予定です）

```
import numpy as np
im = np.load("XY.npy") # 画像の NumPy ファイル XY.npy をロード

from matplotlib import pyplot as plt # Matplotlib パッケージを利用
plt.imshow(im)
plt.show()
```

あくまで参考ですが、`sample.npy` を画像表示した例と、５つの４分割画像領域の平均色を表示した例は以下のようです。