

COMP10001 Foundations of Computing

Semester 2, 2016

Tutorial Questions: Week 11

1. Implement a function `powerset(items)`, which generates a list of all the subsets of `items` (each of which is, in turn, a list). For example:

```
>>> powerset([1,2,3])  
[[], [1], [2], [3], [1,2], [2,3], [1,3], [1,2,3]]
```

You may use `itertools.combinations`, but not `itertools.power`, in your solution.

2. The following code is meant to output a dictionary of player IDs, and the list of cards played by a given player:

```
from collections import defaultdict  
players = iter(range(4))  
cards = [('3C',2), ('5D',3), ('8C',1), ('2H',2)]  
holds = defaultdict(list)  
for card in cards:  
    for player in players:  
        if card[1] == player:  
            holds[player].append(card[0])  
holds = dict(holds)
```

On completion of the code, however, the content of `holds` is:

```
{2: ['3C']}
```

and not the expected:

```
{1: ['8C'], 2: ['3C', '2H'], 3: ['5D']}
```

What is the issue, why does the problem occur, and how can it be fixed?

3. Timbuktu Public Library has recently decided to digitize its archive of local newspapers from the last 50 years. The pages of the newspapers will be scanned and stored as digital color images. Suppose that each newspaper page will be scanned as a *3000 pixels wide* and *6000 pixels high* RGB image. The library currently owns 15 TB (Terabytes) of storage (you may assume that one Terabyte is equal to 1,000,000,000,000 bytes). How many pages of newspaper can be hosted using the library's existing storage?

Note: Assume uncompressed RGB format is used, with one byte for each of the R, G, and B components.

4. The following function reflects an image in the vertical axis:

```

import Image

def open_img(fname):
    img = Image.open(fname)
    pix = img.load()
    mode = img.mode
    dimensions = img.size # width,height
    return dimensions,pix,mode

def vertical(infile,outfile):
    dim,pix_in,mode = open_img(infile)
    img_out = Image.new(mode, dim)
    pix_out = img_out.load()
    for x in range(dim[0]):
        for y in range(dim[1]):
            pix_out[x,y] = pix_in[dim[0]-x-1,y]
    img_out.save(outfile)

```

Based on this, write the following functions (potentially reusing `open_img` from the code provided):

- (a) `rotate180`, that takes an input file `infile` and an output file `outfile` as arguments, and generates a new image in `outfile` based on rotating `infile` by 180 degrees
- (b) `rotate90`, that takes an input file `infile` and an output file `outfile` as arguments, and generates a new image in `outfile` based on rotating `infile` by 90 degrees clockwise
- (c) `rescale`, that takes an input file `infile` and an output file `outfile` as arguments, and generates a new image in `outfile` based on scaling `infile` to half of its original size