CSC1015F Assignment 7: Arrays

Assignment Instructions

This assignment involves constructing Python programs that manipulate lists, lists of lists, and strings.

Question 1 [30 marks]

Write a program called `acronym.py` that may be used to obtain an acronym for a given sentence.

"acronym : a word formed from the first letters of each one of the words in a phrase." (Merriam-Webster)

By way of an example, "Information and Communication Technology Services", "ICTS".

It is common for an acronym to be formed only from the significant parts of a sentence. Words such as 'and', 'the' and 'for' are ignored. Your program should begin by prompting the user to enter a list of words to be ignored. It should then ask the user to enter the sentence for which the acronym is to be generated.

Sample I/O:

```
Enter words to be ignored (separated by commas):
the, for, and
Enter a title to generate its acronym:
The Centre for Theoretical Physics and Astrophysics
The acronym is: CTPA
```

HINT: `'one, two'.split(', ')` is `['one', 'two']`.

Question 2 [40 marks]

Write a module of utility functions called `util.py` for manipulating 2-dimensional arrays of size 4x4. (These functions will be used in Question 3.)

The functions you need to write are as follows:

```
def create_grid(grid):
    """create a 4x4 array of zeroes within grid"""

def print_grid (grid):
    """print out a 4x4 grid in 5-width columns within a box"""

def check_lost (grid):
    """return True if there are no 0 values and there are no
adjacent values that are equal; otherwise False"""

def check_won (grid):
    """return True if a value>=32 is found in the grid; otherwise
False"""

def copy_grid (grid):
    """return a copy of the given grid"""
```

CONTINUED

```
def grid_equal (grid1, grid2):
    """check if 2 grids are equal - return boolean value"""
```

Note: these functions are described using docstrings. See Hussein's lecture notes on the topic: csc1015f_2015_functions.pdf, slide 14.

Use the `testutil.py` test program to test your functions. This program takes a single integer input value and runs the corresponding test on your module. This is a variant of unit testing, where test cases are written in the form of a program that tests your program. You will learn more about unit testing in future CS courses.

Sample I/O:

```
2
+--------------------+
|2          2        |
|     4          8   |
|     16         128 |
|2    2    2    2    |
+--------------------+
```

Sample I/O:

```
7
True
```

Sample I/O:

```
9
True
```

Sample I/O:

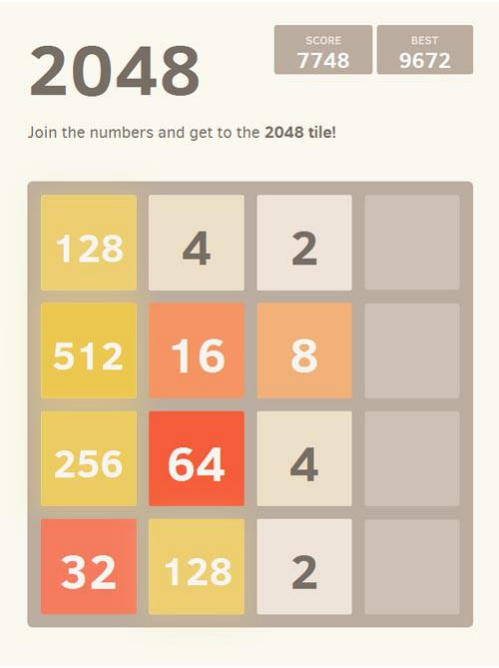
```
17
4 4
16 16
2 2
64 4
64 16
64 2
```

Sample I/O:

```
18
True
```

CONTINUED

Question 3 [30 marks]



2048 is a puzzle game where the goal is to repeatedly merge adjacent numbers in a grid until the number 2048 is found. Your task in this question is to complete the code for a 2048 program, using the utility module (`util.py`) from Question 2 and a supplied main program (`2048.py`).

The heart of the game is the set of merging functions that merge adjacent equal values and eliminate gaps - you are required ONLY to write these functions in a module named `push.py`:

```
def push_up (grid):
    """merge grid values upwards"""

def push_down (grid):
    """merge grid values downwards"""

def push_left (grid):
    """merge grid values left"""

def push_right (grid):
    """merge grid values right"""
```

The original game can be played at: http://gabrielecirulli.github.io/2048/

Note:

- The `check_won`() function from `util.py` assumes you have won when you reach 32 - this is simply to make testing easier.
- The random number generator has been set to generate the same values each time for testing purposes.

Sample I/O:

```
+--------------------+
|                    |
|                    |
|          2         |
|2                   |
+--------------------+
Enter a direction:
l
+--------------------+
|                    |
|                    |
|2                   |
|2         2         |
+--------------------+
Enter a direction:
u
+--------------------+
|4         2         |
|                    |
|                    |
|     4              |
+--------------------+
Enter a direction:
d
+--------------------+
|     2              |
|                    |
|                    |
|4    4    2         |
+--------------------+
Enter a direction:
r
+--------------------+
|               2    |
|                    |
|     2              |
|          8    2    |
+--------------------+
Enter a direction:
x
```

Submission

Create and submit a Zip file called '`ABCXYZ123.zip`' (where ABCXYZ123 is YOUR student number) containing `acronym.py`, `util.py`, and `push.py`.

END