CM1103 Lab worksheet week 6

Lists

1. Write Python statements that achieve the following (in order):
   (a) Create a list containing the strings `"football"`, `"rugby"`, `"hockey"` and `"tennis"`.
   (b) Print the first and last elements of the list
   (c) Add the element `"cycling"` to the end of the list.
   (d) Print how many elements the list has.
   (e) Print the first letter of each element of the list
   (f) Remove the element `"football"`.
   (g) Create a new list containing only the middle 2 elements of the current list.
2. Start `idle` and enter the commands:

```
x = [1,2,3,4]
x.pop(3)
x.remove(3)
```

   What is the difference between the functions `pop` and `remove`? Why does only one produce output? Provide code that creates a list containing the letters `a,b,c,d,e`, then deletes the 4th element of the list (remembering that we start indexing lists at element 0!), and finally deletes the element `'a'`.

Iteration

3. (a) Provide code that uses a pair of nested loops to output a 10x10 square of asterisks to the screen. [Hint: the statement `print "*"` will output an asterisk followed by a newline, the statement `print "*",` will output an asterisk with no newline, and the statement `print` will output a single newline.]
   (b) Rewrite your code as a function that outputs an `m` $\times$ `n` rectangle of asterisks, where `m` and `n` are passed as arguments.
4. Suppose you have a number of pallets to be loaded onto a lorry with a weight limit of 3,000 Kg. Assuming the weights of each pallets (in Kg) are stored in a list (e.g. `weights = [ 750, 387, 291, 712, 100, 622, 109, 750, 282 ]`), write a function that uses a `while` loop to consider each pallet in turn. If it can be added to the lorry without overloading, then print it's weight to the screen and continue to the next pallet. If it would overload the lorry, then stop loading and print out the total weight added so far.

Getting input

5. Write a function that prompts the user to enter length and width values, and displays an appropriately sized rectangle of asterisks (using your answer to 3b).

Sets

6. Write Python code that:
   (a) Creates the **sets** $A = \{1, 2, 3, 4\}$ and $B = \{3, 4, 5, 6\}$.
   (b) Calculates $A \cup B$.
   (c) Calculates $A \cap B$.
   (d) Calculates $A - B$.
   (e) Calculates $(A - B) \cup (B - A)$.
   (f) Calculates $A \cap A$.

Assessed

This question in this section is one of five parts of the coursework *'In-lab' assessment of implementation skills* for CM1103. The coursework is worth 25% of the module in total, with each part worth 5% of the module.

7. Download the file `Rot13.py`, and modify the contents so that the function performs *rot13* on a String argument. *Rot13* is an encryption technique that simply replaces each letter by its partner 13 characters further along the alphabet (see http://en.wikipedia.org/wiki/ROT13).

After completion **and before your lab in week 7**, you will need to go to `http://www.cs.cf.ac.uk/workbook/` to upload the contents of both of your solutions, **and** the output you get from running the doctest commands:

```
python -m doctest -v Rot13.py
```

Save the resulting pdf to disk, and bring a printout of the pdf to your lab session in week 7 to be signed by your lab tutor. Keep the pdf and your signed hard copy safe for your module workbook – you will need to submit this later in the module.

Advanced

8. Exercise 5.1 and 5.2 from Think Python.
9. Find out the difference between the `input` and `raw_input` functions in Python.
10. Write an function `printStarCircle` that outputs a circle of radius 10 made up of asterisks. [Hint: Use nested loops to consider each integer coordinate i, j with —$10 < i, j < 10$. For each pair, output an asterisk if $i^2 + j^2 < 10^2$, otherwise output a space.]
11. Consider question 4. Rewrite your answer so that it tries to add each pallet in order, but skips any pallet that would overload the lorry instead of stopping. How does this change the loop you should use. Also try rewriting your answer so that the function returns a list of the pallet weights that could

not be added. Finally, does this algorithm guarantee an optimal solution to the problem (i.e. does it always load as many pallets as possible, or the maximum weight possible)? Can you think of a list of weights that would give a particularly bad answer?

Competition: *Christmas lights*

I have a string of 50 Christmas lights in a single line, and when they are first plugged in, all of the lights are off. The lights are controlled by a single button, where every time the button is pressed, some of the lights flip their state (i.e. if they are off, they change to on; if they are on, they change to off). The selection of lights which change depends on how many times the button has been pressed so far:

- On the 1st press, the 1st, 2nd, 3rd, 4th, …, 50th lights change state;
- On the 2nd press, the 2nd, 4th, 6th, …, lights change state;
- On the 3rd press, the 3rd, 6th, 9th, …, lights change state;
- …
- On the ith press, the lights corresponding to all multiples of i change state.

If I plug the lights in and press the button **50 times**, which lights will be on? Write some Python code to find the answer.

Email S.M.Allen@cs.cf.ac.uk with your answer and the code you wrote to solve the problem, with the subject line "CM1103 week 6 competition" by midnight on Wednesday in Week 7 – a winner will be picked at random from the correct answers **but** you'll get one extra entry for particularly noteworthy or interesting code, and one extra entry if you can explain why the pattern in the answer occurs.

[Hint: After 3 presses, your lights that are on will be:

```
*--***--***--***--***--***--***--***--***--***-
```

]