CM1103 Lab worksheet week 6

Lists *[Think Python Version 2 - Sections 10.1–10.6, 10.8]*

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 `REPL` in Sublime Text 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'`.

Lambda expressions

The `lambda` keyword is used to define small, anonymous functions. For example, the function:

```
def square(n):
   return n * n
```

could be defined as a lambda expression:

```
square = lambda n : n * n
```

3. Verify that both versions result in the same behaviour.

Lambda expressions allow for more concise and readable code in some cases. For example, the built in `sorted` function takes an optional argument `key` that allows us to specify how elements should be processed before they are compared.

4. What does the following code do when typed into REPL?

```
a = [2, -6, -5, 3, 9]
sorted(a)
sorted(a, key=lambda x : -x)
sorted(a, key=lambda x : abs(x))
sorted(a, key=lambda x : x % 2)
sorted(a, key=lambda x : (x % 2, x))
```

5. Given the list:

```
a = ["tim", "bob", "trevor", "susan", "anna"]
```

use the `sorted` function with an appropriate lambda expression to sort `a` by:

(a) The first letter of each name;

(b) The second letter of each name;

(c) The last letter of each name;

(d) The length of each name;

(e) The length of each name, with ties decided by alphabetical order;

(f) All names that don't start with a vowel are listed first, otherwise in alphabetical order.

Iteration *[Think Python Version 2 - Section 7.3]*

6. (a) Provide code that uses a pair of nested loops to output a 10x10 square of asterisks to the screen. [*Hint*: the function call `print("*")` will output an asterisk followed by a newline, the function call `print("*", end="")` will output an asterisk with no newline, and the function call `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.

7. 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 *[Think Python Version 2 - Section 5.11]*

Note that the functions used to get input changed significantly between Python 2 and 3.

8. 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 6b).

Sets

9. 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 $A \cap A$.

Formative assessment

The question in this section will help to prepare you for the coursework that will be set in week 9.

Download the file `Week6Template.py`, and complete the function as indicated below. Try to ensure your code passes all of the doctests.

10. Complete the function `sumHighest(a, n)` so that it calculates and returns the sum of the `n` highest values in the list `a`. If `n` is less than 0, the answer should be 0; if `n` is greater than the number of elements in `a`, all should be included in the sum.

    For full marks, your function should contain only 2 lines (1 for the definition and 1 for the body), by using *slices* and *built-in functions*.

Advanced

11. Read section 5.9 from Think Python Version 2 and then complete the two exercises at the end of the section.

12. 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.]

13. Consider question 7. 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?

Challenge: ***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 AllenSM@cardiff.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:

*−−−***−−−***−−−***−−−***−−−***−−−***−−−***−−−***−

]