Laboratory 08 — Complex Strings and Conditionals

Topics covered:

- Strings as Sequences
- Complex Conditionals

PREPARATION

Lab attendance is compulsory. You will receive 1 mark for being present at the start of the lab and staying at least until the tutor has finished introducing the lab and has signed your attendance sheet.

EXERCISES

The following exercises must be completed during your allocated laboratory time. You must show your work to the laboratory tutor who will sign off when the work is completed correctly.

EXERCISE 8.1

[3 marks] Write a function named **`case_insensitive_substring()`** which accepts two strings as parameters and returns True if the first string is a substring of the second string and False otherwise. This should work even if the cases do not match.

Arguments: 2 strings

Returns: True or False

Other knowledge: You can use/modify the `change_case()` function from Lecture 21, Slide 15 and `substring()` from Lecture 19, Slide 12. Do not use the `lower()` or `upper()` methods for this exercise.

Sample Tests:

```
>>> case_insensitive_substring('ab', 'ab fab')
True
>>> case_insensitive_substring('Ab', 'ab fab')
True
>>> case_insensitive_substring('ba', 'ab fab')
False
```

EXERCISE 8.2

[3 marks] Write a function named **`advanced_poetry()`** which reads in 4 files (adjective.txt, noun.txt, adverb.txt and verb.txt) and returns a string of poetry of the form “Adjective Noun Adverb Verb Adjective Noun”.

Arguments: none

Returns: a string of poetry with 6 randomly selected words

Other knowledge: Use the `poetry()` function from Lecture 21, Slide 7 as a starting point. You will need to read and use the provided files: adjective.txt, noun.txt, adverb.txt, and verb.txt. File reading was discussed in Lecture 10. You can create “helper” functions to perform repreated tasks and/or to make your code easier to read.

Sample Tests:

```
>>> advanced_poetry()
'beautiful friend quietly loves beautiful friend'
>>> advanced_poetry()
'beautiful cat quickly hates lovely dog'
```

EXERCISE 8.3

[3 marks] Write a function named **`case_insensitive_unique_list()`** which takes a list of strings as a parameter and returns a list of unique values from the parameter list.

Arguments: a list of strings

Returns: a list containing a unique list of strings

Other knowledge: You can use/modify `change_case()` from Lecture 21, slide 15 and `unique_list()` from Lab 5.

Sample Tests:

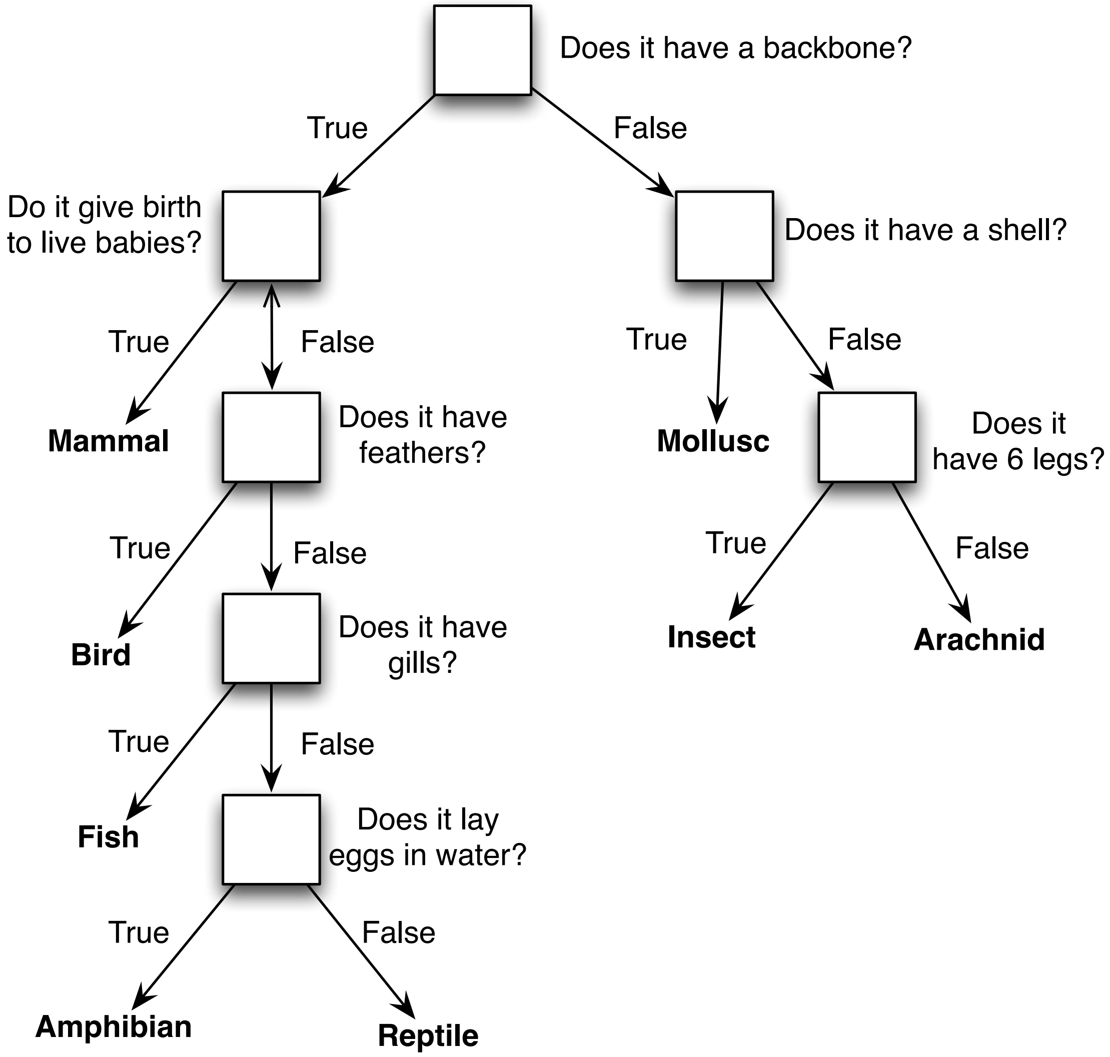
```
>>> case_insensitive_unique_list(['cat', 'dog', 'bug', 'dog', 'ant', 'dog', 'bug'])
['cat', 'dog', 'bug', 'ant']

>>> case_insensitive_unique_list(['Welcome', 'to', 'COMPSCI101', 'To'])
['Welcome', 'to', 'COMPSCI101']
```

The following exercises must be completed prior to the start of your Lab 9 session. Include all the exercises in a single module (file), named "Lab08_Homework.py". Your file must include a docstring at the top of the file containing your name, UPI and ID number. You must submit the file containing your exercises using the Assignment Dropbox before the start of Lab 09.

EXERCISE 8.4

Write a function named **`which_animal()`** which given a list of 7 answers to True/False questions, returns the class of an animal. The order of the 7 questions is:
Does it have a backbone?
Does it have a shell?
Does it have 6 legs?
Does it give birth to live babies?
Does it have feathers?
Does it have gills?
Does it lay eggs in the water?



Arguments: list of booleans

Returns: a string specifying the type of animal.

Sample Tests:

```
>>> which_animal([True,True,True,True,True,True,True])
'Mammal'

>>> which_animal([False,False,False, False, False, False, False])
'Arachnid'
```

EXERCISE 8.5

Write a function named **`movie_price()`** which returns the price of a movie ticket given the day of the week and the hour when the movie begins (given on a 24 hour clock). The prices are:

- 10.75 all day Tuesday
- 5.75 all day Wednesday
- 12.75 before 5pm Monday, Thursday, and Friday
- 15.75 weekends and from 5pm Monday, Thursday, and Friday

Arguments: an input filename

Returns: the average score (float)

Sample Tests:

```
>>> movie_price('Tuesday',4)
10.75

>>> movie_price('Saturday',15)
15.75

>>> move_price('Friday', 17)
15.75
```

ADVANCED EXERCISES (OPTIONAL)

EXERCISE 8.6

[0 marks] Write a function named **`replace_text_list()`** which takes 2 parameters: the name of the file which contains the string and a list of pairs of words (in a list) where the first word is to be replaced by the second word. It prints out the string after the replacement is done.

Arguments: an input filename, a list of lists of words

Prints: the string after the words are replaced

Other Knowledge: You might want to call the `replace_text()` function from lab 5.

Sample contents of input file:

```
The woods are lovely dark and deep
But I have promises to keep
And miles to go before I sleep
And miles to go before I sleep
```

Sample Tests:

```
>>> replace_text_list("words.txt", [["go", "think"], ["dark","deep"]])
The woods are lovely deep and deep
But I have promises to keep
And miles to think before I sleep
And miles to think before I sleep

>>> replace_text_list("words.txt", [["go", "think"],["dark","deep"],["deep","sleep"]])
The woods are lovely sleep and sleep
But I have promises to keep
And miles to think before I sleep
And miles to think before I sleep
```

EXERCISE 8.7

[0 marks] Write the **`crossword_helper()`** function which takes 2 parameters. A word with wildcards in it like "*arn*val" and a file name containing a dictionary. It returns a list of all words that match the word like ["carnival"].

Arguments: a filename of a file containing a dictionary and a string with wildcard(s).
Other Knowledge: The wildcard "*" can match any letter. Use the file unixdict.txt.
Returns: list of words which match the parameter string

Sample output:

```
>>> crossword_helper("unixdict.txt","si*t")
['sift', 'silt']
```

ASSESSMENT

Name:____

Lab day and time: ____

Check list for laboratory exercises (to be completed by Lab tutor)

On time: □ (1 mark)

Exercise 8.1: □ (3 marks)

Exercise 8.2: □ (3 marks)

Exercise 8.3: □ (3 marks) Teaching Assistant: ________________________

Total mark: __________________/10 Tutor: ________________________

MARKING SCHEME

Marks	Feedback
0.5	Include a docstring at the top of the file containing your name, UPI and ID number
0.5	Include a docstring in each function.
1	Include all the exercises in a single file
1	The `which_animal()` function is defined correctly.
3	Well done! Your function passed test case 1, 2, and 3.
1	The `movie_price()` function is defined correctly.
3	Well done! Your function passed test case 1, 2, and 3.