Version: 2025/04/01 12:49

CSC1015F Assignment 6B (70 Marks)

Strings

Learning Objectives

By the end of this assignment, you should be able to:

- Know the structure of a string and how its component characters are encoded using Unicode.
- Be able to apply indexing to access individual characters in a string.
- Be aware of the different string functions and methods, such as `len()`, `count()`, `upper()`, and `lower()`, and what they are used for.
- Understand string slicing and all the variations in its syntax.
- Be able to apply this knowledge to solve string manipulation problems.

Assignment Instructions

This assignment involves constructing Python programs that use input and output statements, '`if`' and '`if-else`' control flow statements, '`while`' statements, '`for`' statements, and statements that perform `string` manipulation.

String concepts: string, index, `int()`, `str()`, `find()`, `strip()`, slice, concatenation, iteration over.

NOTE Your solutions to this assignment will be evaluated for correctness and for the following qualities:

- Documentation
  - Use of comments at the top of your code to identify program purpose, author and date.
  - Use of comments within your code to explain each non-obvious functional unit of code.
- General style/readability
  - The use of meaningful names for variables and functions.
- Algorithmic qualities
  - Efficiency, simplicity

These criteria will be manually assessed by a tutor and commented upon. In this assignment, up to 10 marks will be deducted for deficiencies.

Question 1 [30 marks]

Write a Python program called `punctuation.py` that takes a paragraph as input and:

- Removes all punctuation marks (.,!?;:) from a paragraph.
- Counts and displays the number of unique words in a paragraph.

You may NOT use built-in functions like `split()`, `replace()`, `set()`, etc. However, you may use `len()`.

Sample IO *(The input from the user is shown in* ***bold*** *font – do not program this)*:

```
Enter a paragraph of any length:
Wait... did you see that!? It was incredible; I couldn't believe my
eyes: a flying cat!
A paragraph without punctuation:
Wait did you see that It was incredible I couldn't believe my eyes a
flying cat
The number of unique words in a paragraph:
16
The unique words are:
['Wait', 'did', 'you', 'see', 'that', 'It', 'was', 'incredible',
'I', "couldn't", 'believe', 'my', 'eyes', 'a', 'flying', 'cat']
```

Sample IO *(The input from the user is shown in* ***bold*** *font – do not program this)*:

```
Enter a paragraph of any length:
I love cooking: pasta, pizza, and burgers; but desserts? Oh, they're
my weakness!
A paragraph without punctuation:
I love cooking pasta pizza and burgers but desserts Oh they're my
weakness
The number of unique words in a paragraph:
13
The unique words are:
['I', 'love', 'cooking', 'pasta', 'pizza', 'and', 'burgers', 'but',
'desserts', 'Oh', "they're", 'my', 'weakness']
```

Question 2 [20 marks]

Students who enter university are often shocked to learn that they need to provide references in all their written work (and even programs). A common mistake is using the wrong format or, even worse, using inconsistent formats in the list of references. This can lead to violent reactions from your lecturer or even the loss of precious marks! Fortunately, this can be remedied with a simple program to reformat references to be consistent.

Write a program called '`references.py`' to reformat references as follows:

- The author names are in title case.
- The title has only the first letter capitalised and,
- The rest remains the same.

Assume that the input reference format is: *a list of authors, space, (year), space, title, comma, other information.*

Hint: You will need to use string manipulation functions such as: `find`, `title`, `capitalize`, `replace`, `[:]` (slicing).

Sample IO *(The input from the user is shown in **bold** font – do not program this)*:

```
Enter the reference:
poulo, lebeko bernard (2013) fine-grained scalability Of digital
library services In The cloud, SAICSIT Conference 2014, ACM, pp23-34,
2014
Reformatted reference:
Poulo, Lebeko Bernard (2013) Fine-grained scalability of digital
library services in the cloud, SAICSIT Conference 2014, ACM, pp23-34,
2014
```

Sample IO *(The input from the user is shown in **bold** font – do not program this)*:

```
Enter the reference:
suleman, h (2009) Utility-based high performance digital library
systems, Proceedings of Second Workshop on Very Large Digital
Libraries (VLDL 2009), 2 October 2009, Corfu, Greece, 1-8, DELOS.
Reformatted reference:
Suleman, H (2009) Utility-based high performance digital library
systems, Proceedings of Second Workshop on Very Large Digital
Libraries (VLDL 2009), 2 October 2009, Corfu, Greece, 1-8, DELOS.
```

Sample IO *(The input from the user is shown in* ***bold*** *font – do not program this)*:

```
Enter the reference:
POULO, lebeko bernard (2013) fine-grained scalability Of digital
library services In The cloud, SAICSIT Conference 2014, ACM, pp23-
34, 2014
Reformatted reference:
Poulo, Lebeko Bernard (2013) Fine-grained scalability of digital
library services in the cloud, SAICSIT Conference 2014, ACM, pp23-
34, 2014
```

Question 3 [20 marks]

Write a Python program called `search_word.py` that asks the user to input a sentence and a word to find. The program should:

- Display whether the word is present in the sentence or not.
- If found, show the index where the word first appears in the sentence.

You may NOT use any built-in functions for this question.

HINT:

- The word to find is not case sensitive.
- The index where the word first appears in the sentence is the index of the first character in that word (see sample I/O below).

Sample IO *(The input from the user is shown in* ***bold*** *font – do not program this)*:

```
Enter a sentence:
The Quick Brown Fox Jumped Over The Lazy Dog
Enter the word to find:
brown
The word 'brown' is in the sentence.
It first appears at index 10.
```

Sample IO *(The input from the user is shown in* ***bold*** *font – do not program this)*:

```
Enter a sentence:
Hello, world! This is a test.
Enter the word to find:
again
The word 'again' is NOT in the sentence.
```

Sample IO *(The input from the user is shown in* ***bold*** *font – do not program this)*:

```
Enter a sentence:
She completed the assignment in just under three hours.
Enter the word to find:
TUTORIAL
The word 'tutorial' is NOT in the sentence.
```

Submission

Create and submit a Zip file called '`ABCXYZ123.zip`' (where ABCXYZ123 is YOUR student number) containing `punctuation.py`, `references.py`, and `search_word.py`.

NOTES:

1. FOLDERS ARE NOT ALLOWED IN THE ZIP FILE.
2. As you will submit your assignment to the Automarker, the Assignment tab may say something like "Not Complete". THIS IS COMPLETELY NORMAL. IGNORE IT.