Coursework 2

Introduction to Programming (COMP 1012)

You should follow the instructions on preparing your submission. You are only allowed to import modules specified in the question. Standard university penalty of 5% of available marks per day, or part of a day, will apply to late work. Late submissions are acceptable up to 7 days late. Feedback on late submissions may not be provided within 3 weeks of submission deadline.

Submission: You must submit your work via Gradescope.

Deadline: 1000 GMT 01/12/22

Weighting: This coursework is worth 30% of the module grade.

1 Introduction

You are required to implement a `leopard.py` module and JavaScript function in `morsecode.html`.

2 Preparation

Please follow the following instructions.

- Please download the template `cwk2-files.zip` from Minerva Ultra.
- Unzip `cwk2-files.zip` and you should have the file `leopard.py`, diabetes_data.csv, fide2021.csv, student.csv, `morsecode.html`. You should write your code in `leopard.py` and `morsecode.html` and submit them to Gradescope. Do not upload the csv files and do not change the name of the files.
- Write your name at the top of the file indicated by `@author`.

3 `leopard.py` module (35 marks)

You are required to implement a `Leopard` class to read data from any csv file with methods to extract specific data from that csv file.

Given a module template file called `leopard.py`, create a class called `Leopard` with the following methods. You should create appropriate attributes. For this question, you are only allowed to import the `csv` module.

3.1 `__init__(self, filepath: str)-> None` (5 marks)

This method reads in any comma-separated csv file and split the data in the csv file into `header` and the remaining `data` using the csv module. Write codes to handle potential errors with file content (e.g empty file, print `"empty file."`) and file not found (print `"file not found."`).

3.2 `get_header(self) -> list` (1 mark)

This method returns the `header` part of the csv file as a `list`, and `None` if empty file or file not found.

3.3 `get_data(self) -> list` (1 mark)

This method returns the `data` part of the csv file as a `list`, and `None` if empty file or file not found.

3.4 `stats(self) -> dict` (8 marks)

This method returns a dictionary on the count, mean, minimum, and maximum value for each numerical column in the format `{col1_header:{count: value, mean: value, min: value, max: value}, col2_header:{count: value, mean: value, min: value, max: value}, ...}`. You should also consider missing values in cells commonly labelled NA, -, or empty string, by ignoring those values. The *mean* should be rounded to 2 decimal places using function *round(x, 2)*, where x is the calculated mean.

3.5 `html_stats(self, stats: dict, filepath: str) -> None` (10 marks)

(Only attempt this if you have completed 3.4)

This method creates an html file called `filepath` and with data in `stats` (such as those returned from 3.4) formatted as an html table in the file `filepath`. You can present a basic html table or with more sophisticated and professional-looking with formatting and style such as centralised text, different fonts, cell colour, and so on.

3.6 `count_instances(self, criteria, ...) -> int` (10 marks)

(Only attempt this if you have completed 3.1 - 3.5)

This method returns the number of instances meeting the *criteria*. You are to decide the data type format for *`criteria`* as well as the number of arguments to this method. For example, in a csv file with headers `Age`, `Gender`, `Itching`, `Obesity`, and `weakness`, the criteria can be `Age=20, Gender="Male"` or according to the data type format that you have chosen. You have to decide how to pass and accept these as argument(s) in the method. You also need to write detailed docstring documentation (comment in triple quotes) on how to use this method after the method definition.

4 JavaScript (25 marks)

Modify the JavaScript function `encdec()` in `morsecode.html` to encode/decode Morse code/string in the (first) `textarea` with `id="input"`, based on the selection box. When the submit button is clicked, the encoded/decoded Morse code is displayed in the (second) `textarea` with `id="output"` as shown in Figure 2. You can create other functions to be used within the function `encdec()`.

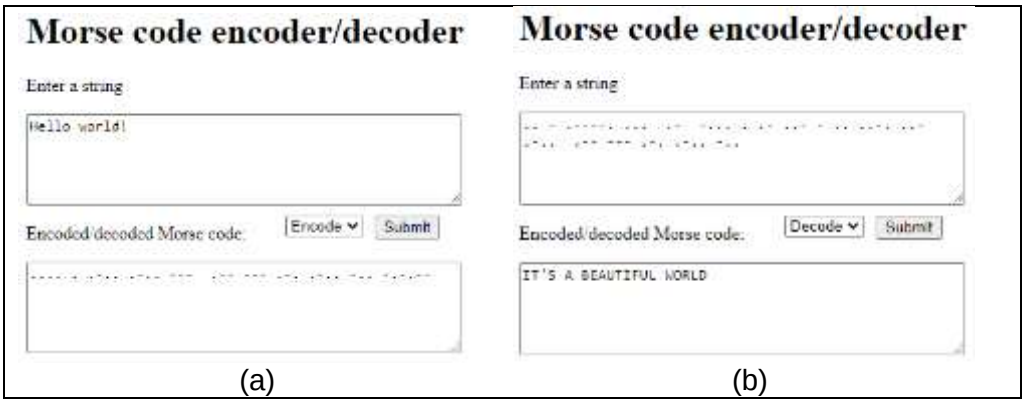

Figure 2 (a) encode a string to Morse code (b) decode a Morse code to string

Each character converted to Morse code is spaced with a blank space, except the last character. The following `mcode` object is provided.

```
mcode = {
'A': '.-',   'B': '-...', 'C': '-.-.', 'D': '-..',   'E': '.',
'F': '..-.', 'G': '--.',  'H': '....', 'I': '..',    'J': '.---',
'K': '-.-',  'L': '.-..', 'M': '--',   'N': '-.',    'O': '---',
'P': '.--.', 'Q': '--.-', 'R': '.-.',  'S': '...',   'T': '-',
'U': '..-',  'V': '...-', 'W': '.--',  'X': '-..-',  'Y': '-.--',
'Z': '--..', ' ': ' ',    '0': '-----','1': '.----', '2': '..---',
'3': '...--',  '4': '....-',  '5': '.....',  '6': '-....',
'7': '--...',  '8': '---..',  '9': '----.',  '&': '.-...',
"'": '.----.', '@': '.--.-.', ')': '-.--.-', '(': '-.--.',
':': '---...', ',': '--..--', '=': '-...-',  '!': '-.-.--',
'.': '.-.-.-', '-': '-....-', '+': '.-.-.',  '"': '.-..-.',
'?': '..--..', '/': '-..-.'
}
```

The table below shows strings and encoded Morse codes. For input with invalid characters, except a space, prompt an alert with “Invalid input” as shown in Figure 3. Pay attention to the spaces between characters converted to Morse code especially for a blank space in the string.

Examples	
String	**Morse code**
Hello world!	.... . .-.. .-.. --- .-- --- .-. .-.. -.. -.-.-- For clarity, the code above is illustrated below where x is a space ....x.x.-..x.-..x---xx.--x---x.-.x.-..x-..x-.-.--

H		e		l		l		o			w		o		r		l		d		!
....		.		.-..		.-..		---			.--		---		.-.		.-..		-..		-.-.--

Examples	
String	**Morse code**
morse code	-- --- .-. ... . -.-. --- -.. . --x---x.-.x...x.xx-.-.x---x-..x.
IT'S A BEAUTIFUL WORLD	.. - .----. ... .- -... . .- ..- - .. ..-. ..- .-.. .-- --- .-. .-.. -.. ..x-x.----.x...xx.-xx-...x.x.-x..-x-x..x..-.x..-x.-..xx.--x---x.-.x.-..x-..
a	.-
$	Alert “Invalid input”
Morse$code	Alert “Invalid input”

Figure 3: Invalid character $ in the input string to encode.

6 Marking

`leopard.py` module	35
Python coding style and comments	5
JavaScript	20
JavaScript coding style and comments	5
Total	65