SCHOOL OF COMPUTING (SOC)

Diploma in Applied AI and Analytics

ST1507 DATA STRUCTURES AND ALGORITHMS (AI)

2021/2022 SEMESTER 2
ASSIGNMENT ONE (CA1)
~ Morse Code Message Analyzer ~

Objective of Assignment

To practice what you have learnt in the module on data structures, algorithms, and object-oriented programming by developing a Morse Code Message Analyzer.

Instructions and Guidelines:

1. This is an individual assignment, and it accounts to **30%** of your final grade.

2. The submission date is **Friday 26 November 5:00 pm**.

3. The development will be carried out in Python using Anaconda.

4. The interviews will be conducted during the DSAA lessons in week 7. You are expected to explain on your code and program logic. Take note that the interview is compulsory. In case you are absent from the interview without valid reasons, you will not get any marks for the assignment.

5. No marks will be given, if the work is copied, or you have allowed others to copy your work.

6. **50% of marks** will be deducted for submission of assignment within **ONE** calendar day after the deadline. **No marks shall** be awarded for assignments submitted **more than one day** after the deadline.

Warning: Plagiarism means passing off as one's own the ideas, works, writings, etc., which belong to another person. In accordance with this definition, you are committing plagiarism if you copy the work of another person and turning it in as your own, even if you would have the permission of that person. Plagiarism is a serious offence and disciplinary action will be taken against you. If you are guilty of plagiarism, you may fail all modules in the semester, or even be liable for expulsion.

Overview of the system

Your job is to implement a *Morse Code Message Analyzer*. The application should be able to convert a line of plain text into morse code.

For example: THIS IS A SOS

Will be converted to: -,....,..,..., ,..,..., ,.-, ,...,---,...

- A Morse Code lookup table is provided in appendix A
- For our conversion, morse words will be separated by spaces, morse letters will be separated by commas.

The application should also be able to analyze a multiline morse code message that is provided in a text file. The objective of the analyses is to extract a shorter message from the original message as to capture the essence of the message. Your application will do that by removing all the redundant 'stop-words'. Then it will make use of the frequency and location of the remaining words as to sort them by importance to form an essential message. A list of stop-words will be provided on the Blackboard.

An example, is shown below:

Morse Code message to analyze:

```
....,.,.-..,.--., ,..-,..., ,...,---,..., ,...,---,..., ,...,---,...
---,..-,.-., ,...,....,..,.--., ,....,.-,..., ,....,..,-, ,.-,-., ,..,-.-.,.,-...,.,.-.,--.
.--.,.-..,.,.-,...,., ,....,.,.-..,.--., ,..-,...
-,....,..,..., ,..,..., ,.-, ,...,---,...
---,..-,.-., ,...,....,..,.--., ,..,..., ,...,..,-.,-.-,..,-.,--.
.--,., ,....,..,-, ,.-,-., ,..,-.-.,.,-...,.,.-.,--.
-,....,..,..., ,..,..., ,.-,-., ,...,---,...
```

Morse Code message converted to plain text:

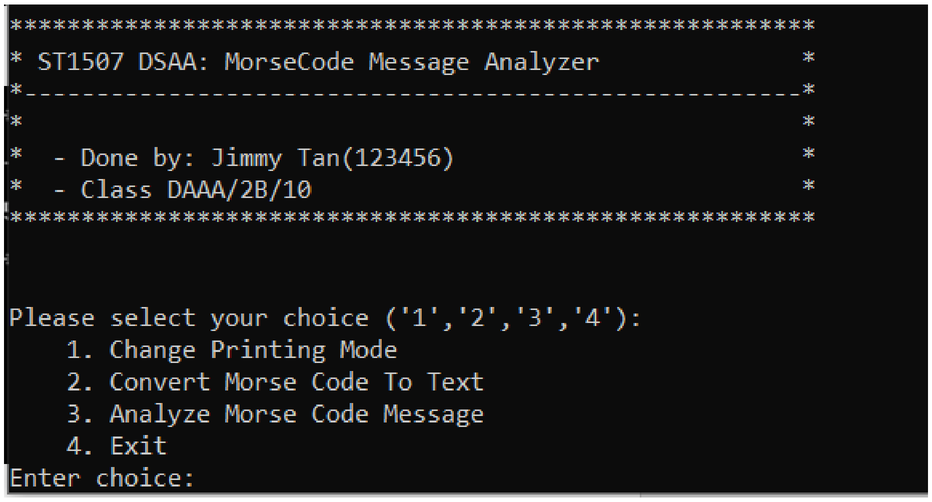
```
HELP US SOS SOS SOS
OUR SHIP HAS HIT AN ICEBERG
PLEASE HELP US
THIS IS A SOS
OUR SHIP IS SINKING
WE HIT AN ICEBERG
THIS IS AN SOS
```

Essential Message with stop words removed and remaining words sorted by frequency and location:

SOS HELP SHIP HIT ICEBERG SINKING

<u>Selection menu</u>

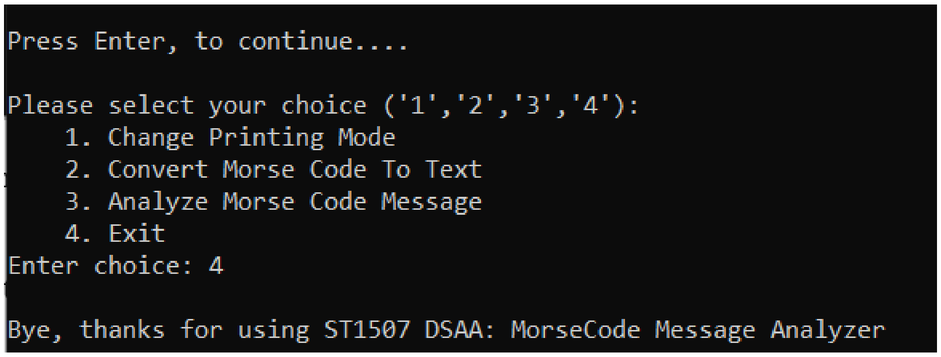
When the application starts, the user will be presented with a menu as is shown below, allowing the user to choose from 4 options (‘1’,’2’,’3’,’4’).



```
********************************************************
* ST1507 DSAA: MorseCode Message Analyzer              *
*------------------------------------------------------*
*                                                      *
*  - Done by: Jimmy Tan(123456)                        *
*  - Class DAAA/2B/10                                  *
********************************************************


Please select your choice ('1','2','3','4'):
    1. Change Printing Mode
    2. Convert Morse Code To Text
    3. Analyze Morse Code Message
    4. Exit
Enter choice:
```

- You are required to follow the above format, please ensure that you display your name, student ID and class in the format as is shown in the example.

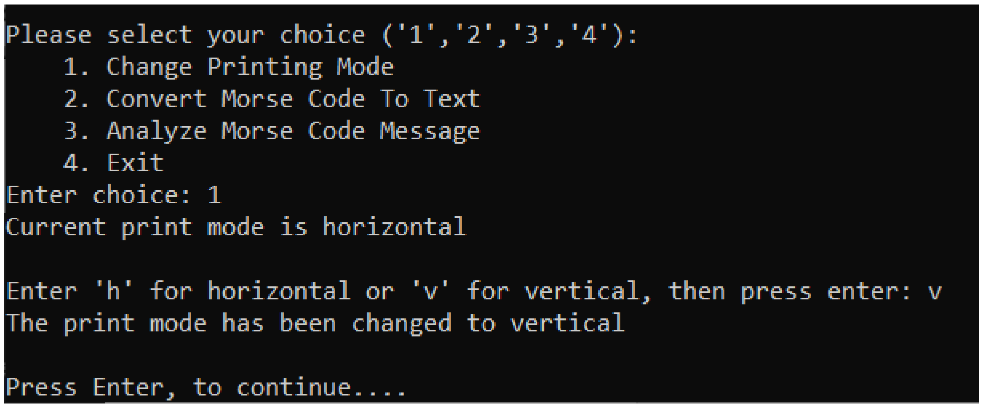
- The user will be able to repeatedly select options from the menu. The application will terminate once the user selects option 4 (Exit).



```
Press Enter, to continue....

Please select your choice ('1','2','3','4'):
    1. Change Printing Mode
    2. Convert Morse Code To Text
    3. Analyze Morse Code Message
    4. Exit
Enter choice: 4

Bye, thanks for using ST1507 DSAA: MorseCode Message Analyzer
```

<u>Option 1: Changing Printing Mode</u>

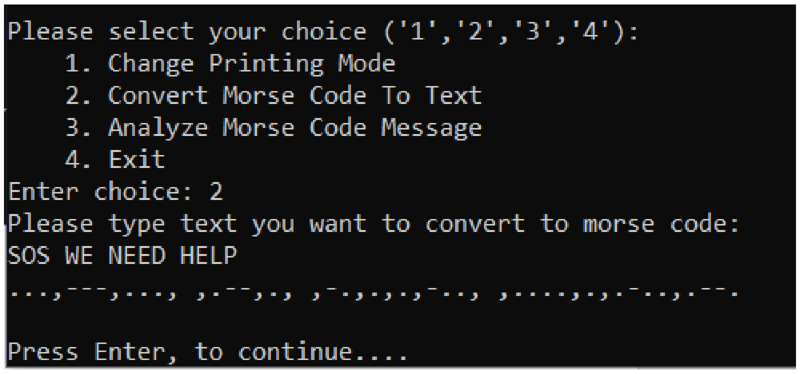
When the user selects option ‘1’, he/she will be allowed to change the printing mode. The Printing mode can be changed to either horizontal (‘h’) or vertical (‘v’). The default Printing Mode is horizontal.



```
Please select your choice ('1','2','3','4'):
    1. Change Printing Mode
    2. Convert Morse Code To Text
    3. Analyze Morse Code Message
    4. Exit
Enter choice: 1
Current print mode is horizontal

Enter 'h' for horizontal or 'v' for vertical, then press enter: v
The print mode has been changed to vertical

Press Enter, to continue....
```

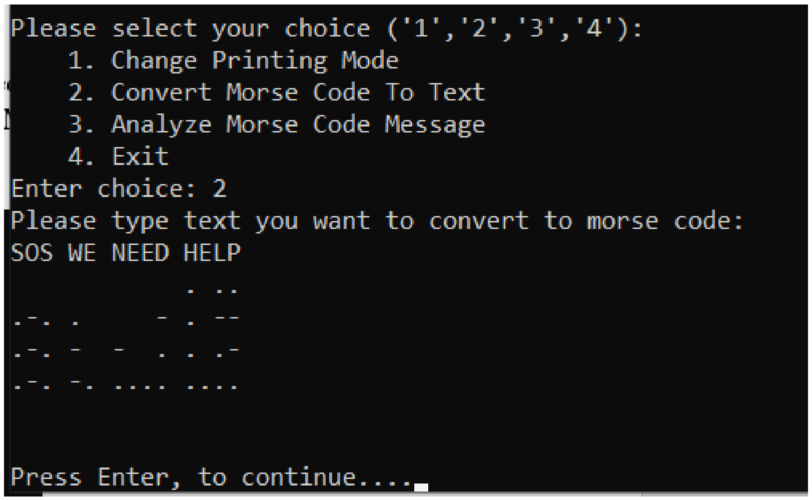
<u>Option 2: Convert Morse Code to Text</u>

When the user selects option ‘2’, he/she will be prompted to enter a line of plain text for conversion to Morse Code. After pressing Enter, it will then print the morse code either horizontally or vertically (depending on the current Printing Mode). Two examples are provided below.



```
Please select your choice ('1','2','3','4'):
    1. Change Printing Mode
    2. Convert Morse Code To Text
    3. Analyze Morse Code Message
    4. Exit
Enter choice: 2
Please type text you want to convert to morse code:
SOS WE NEED HELP
...,---,..., ,.--,., ,-.,.,.,-.., ,....,.,.-..,.--.

Press Enter, to continue....
```

Morse code printed horizontally

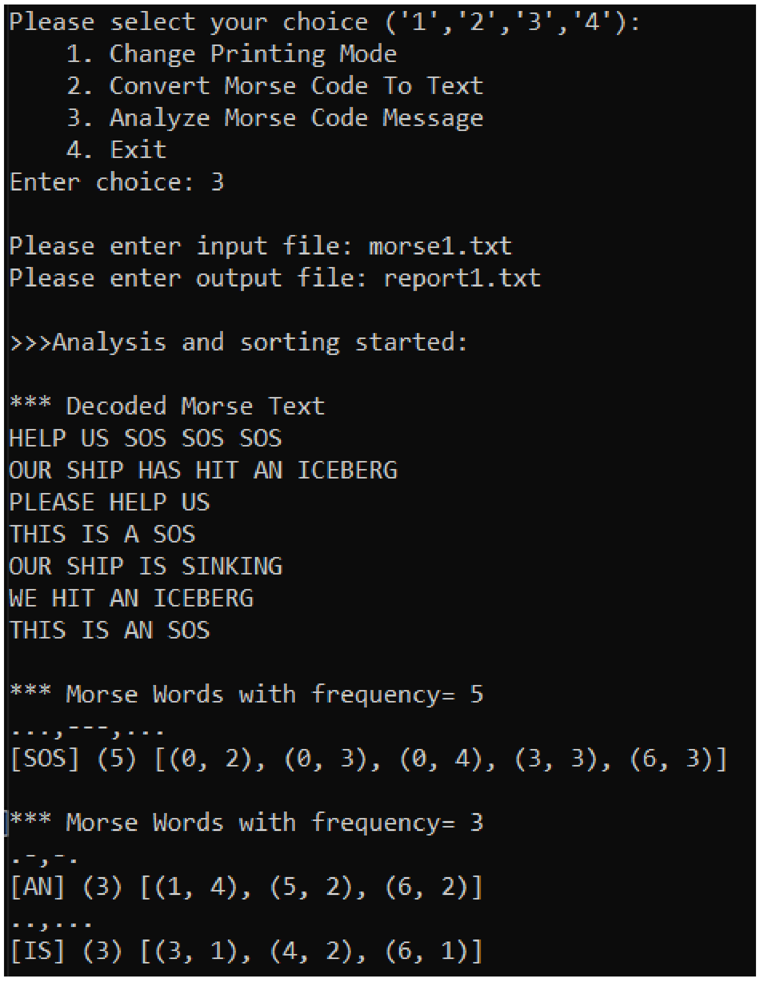


```
Please select your choice ('1','2','3','4'):
    1. Change Printing Mode
    2. Convert Morse Code To Text
    3. Analyze Morse Code Message
    4. Exit
Enter choice: 2
Please type text you want to convert to morse code:
SOS WE NEED HELP
            . ..
.-. .     - . --
.-. -  -  . . .-
.-. -. .... ....


Press Enter, to continue....
```

Morse code printed vertically

Option 3: Analyze Morse Code

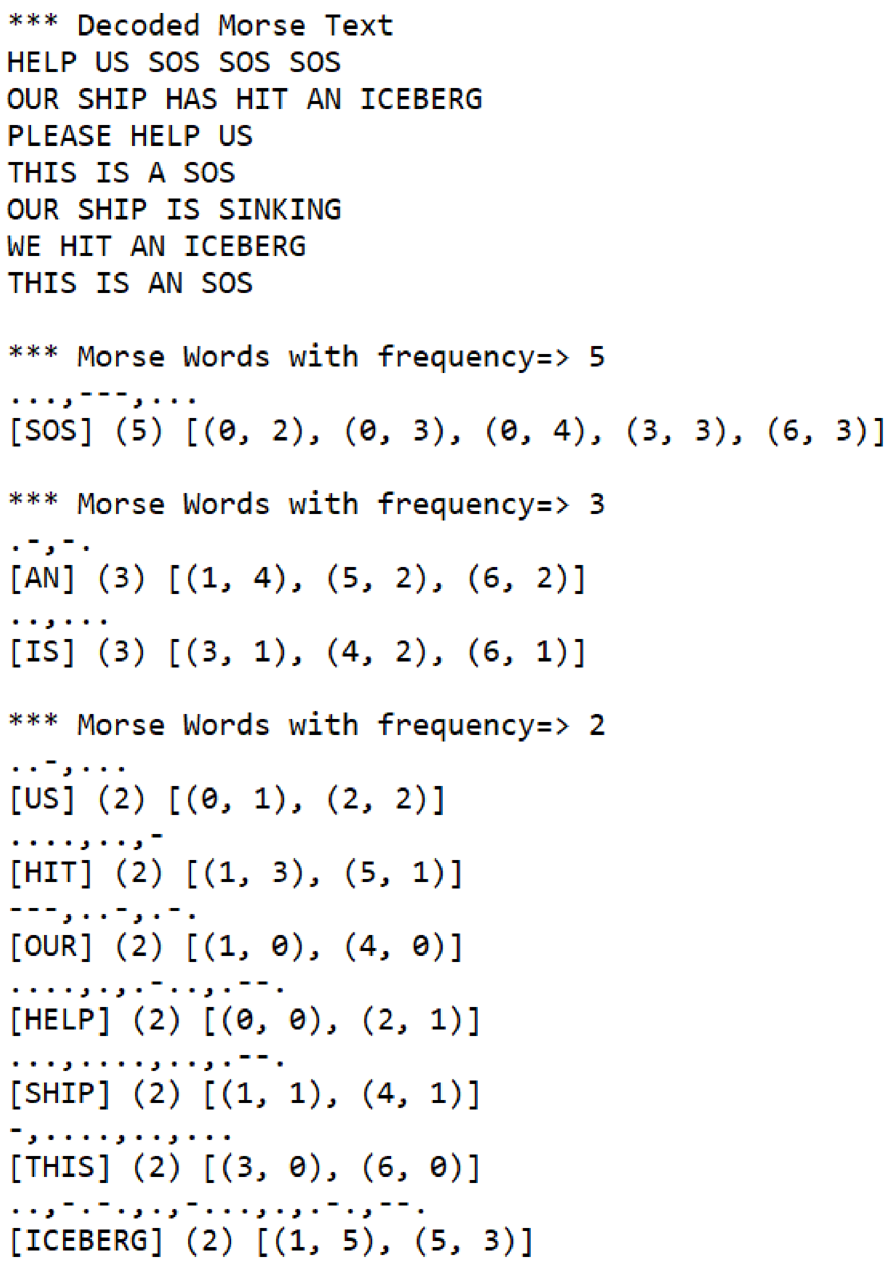
When the user selects option '3', he/she will be prompted to enter an input file and output file. The application reads the Morse Message from the input file, and then generates a report. The output will then, be printed on the screen, as well as written to the output file (output depicted partially below). For an example of a complete report see Appendix B.



```
Please select your choice ('1','2','3','4'):
    1. Change Printing Mode
    2. Convert Morse Code To Text
    3. Analyze Morse Code Message
    4. Exit
Enter choice: 3

Please enter input file: morse1.txt
Please enter output file: report1.txt

>>>Analysis and sorting started:

*** Decoded Morse Text
HELP US SOS SOS SOS
OUR SHIP HAS HIT AN ICEBERG
PLEASE HELP US
THIS IS A SOS
OUR SHIP IS SINKING
WE HIT AN ICEBERG
THIS IS AN SOS

*** Morse Words with frequency= 5
...,---,...
[SOS] (5) [(0, 2), (0, 3), (0, 4), (3, 3), (6, 3)]

*** Morse Words with frequency= 3
.-,-.
[AN] (3) [(1, 4), (5, 2), (6, 2)]
..,...
[IS] (3) [(3, 1), (4, 2), (6, 1)]
```

Below is a portion of the report that shows:

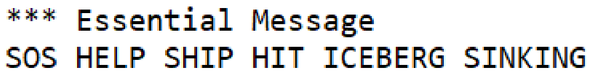
- Decoded Morse Text
- Information on morse words, such as frequency and locations for each word.
- Take note, morse words are sorted by frequency in descending order. Morse words with same frequency, will be sorted by length in ascending order. Words with same frequency and length will be sorted alphabetically in ascending order.



```
*** Decoded Morse Text
HELP US SOS SOS SOS
OUR SHIP HAS HIT AN ICEBERG
PLEASE HELP US
THIS IS A SOS
OUR SHIP IS SINKING
WE HIT AN ICEBERG
THIS IS AN SOS

*** Morse Words with frequency=> 5
...,---,...
[SOS] (5) [(0, 2), (0, 3), (0, 4), (3, 3), (6, 3)]

*** Morse Words with frequency=> 3
.-,-.
[AN] (3) [(1, 4), (5, 2), (6, 2)]
..,...
[IS] (3) [(3, 1), (4, 2), (6, 1)]

*** Morse Words with frequency=> 2
..-,...
[US] (2) [(0, 1), (2, 2)]
....,..,-
[HIT] (2) [(1, 3), (5, 1)]
---,..-,.-.
[OUR] (2) [(1, 0), (4, 0)]
....,.,.-..,.--.
[HELP] (2) [(0, 0), (2, 1)]
...,....,..,.--.
[SHIP] (2) [(1, 1), (4, 1)]
-,....,..,...
[THIS] (2) [(3, 0), (6, 0)]
..,-.-.,.,-...,.,.-.,--.
[ICEBERG] (2) [(1, 5), (5, 3)]
```

Below is a portion of the report that shows:

- The essential message sorts the morse words <u>after</u> the stop words have been removed.
- Take note, morse words are sorted by their frequency in descending order.
- Morse words with same frequency are sorted by their location (with words that showed up first in original message will also come first in essential message).

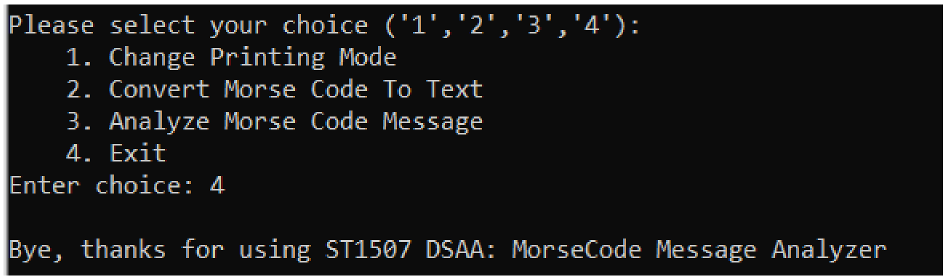


```
*** Essential Message
SOS HELP SHIP HIT ICEBERG SINKING
```

You are required to follow the same format for the report as shown in the example of a complete report in Appendix B.

<u>**Option 4: Exit**</u>

The user can repeatedly select Options 1,2,3. Option 4 is to exit the program. You should follow the output as is shown below.



```
Please select your choice ('1','2','3','4'):
    1. Change Printing Mode
    2. Convert Morse Code To Text
    3. Analyze Morse Code Message
    4. Exit
Enter choice: 4

Bye, thanks for using ST1507 DSAA: MorseCode Message Analyzer
```

Basic requirements:

- You are required to design and write the Python application using an object-oriented programing approach (OOP). You should leverage on your knowledge of encapsulation, function/operator overloading, polymorphism, inheritance etc.

- You may make use of Python’s already build in data structures, such as list, tuple, dictionary etc., however you should refrain from using the classes from the collection library. Instead, you are required to write you own classes to support the various data structures that you may need (for instance for sorting you may need a class *SortedList*). Of course, you may refer to the lecture slides and lab tasks and expand further on those classes that we had previously developed and discussed in tutorials and lab sessions.

- The classes that you develop must be placed in separate python files.

- Pay attention to user input validation. Your application should not crash if a user types in the wrong input. Instead, when a user enters wrong input, you should notify the user, and allow him/her to enter again.

Deliverables

Your deliverables must include:

1. A report (as pdf file) with a maximum **of 6** pages, excluding the front page, Appendix, and the source code. The report should contain:

    - A Front/Title page, that states your name, student ID and class
    - Description, and user guidelines, as on how to operate your application (please include screen shots of your application in action).
    - Description, and purpose of the various algorithms and data structures that you may have implemented (For instance a SortedList). You may discuss on issues such as, challenges faced, performance issues (e.g., Big O) limitations and constraints etc. Discuss your motivation to make use of certain build in python data structures, and/or why you may have chosen to develop you own data structures.
    - Describe how you made use of the Object-Oriented Programming (OOP) approach. You may elaborate of the classes that you have developed, and discus on issues such as encapsulation, function/operator overloading, polymorphism, inheritance etc.
    - You must include all the python source code for the application (the actual code you wrote), as an appendix at the end of your report. Make sure your code is properly commented, and that for each python file you write on top in comments, your name, student ID and class.

2. Submit all the python files (.py files). Make sure the code that you submit is complete, and that it can run from the Anaconda prompt.

Submission deadline

Final Submission is on **Friday 26 November 5:00 pm**.

-Submit all the deliverables (Source Code, Report) in the designated Blackboard drop box before the deadline.

- You must submit it as one Zipped folder (RAR will not be accepted, only zip). Label your submission as:

CA1_Name_StudentID_Class.zip

For example: *CA1_JimmyTan_12345_2B10.zip*

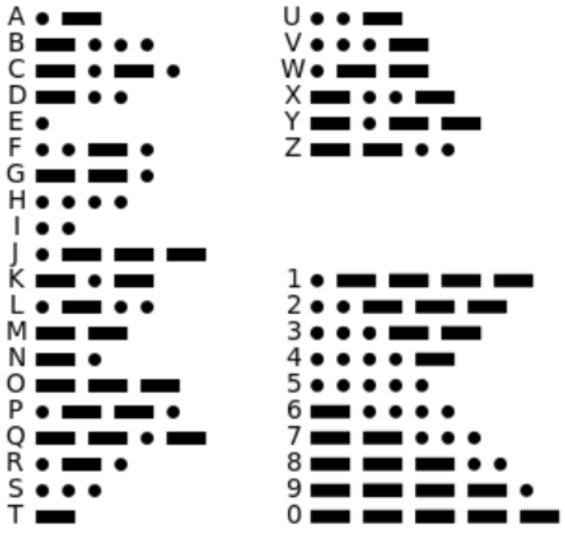
Assessment Criteria

The assignment will be assessed based on the following criteria:

Assessment criteria	**Marks awarded**
User Interface and File IO - An interactive user interface that follows the required format, as shown in screenshots. - Proper user input validation (handles wrong user inputs correctly). - Reading and writing to files is functional.	Max 20
Functionality of the application: - Able to convert plain text to morse code correctly. - Able to print morse code horizontally, as well as vertically, correctly. - Able to analyze morse code message correctly and to produce a report that follows the required format.	Max 30
Programming techniques, efficiency, robustness, and readability of code: - Usage of classes and OOP technology. - Usage of data structures (python build in data structures as well as those data structures you made yourself). - Functions and programming constructs. - Code is properly commented. - Application if free of crashes.	Max 30
Report: - Most follow the required format (as spelled out under deliverables 1)	Max 10
Demo: - Ability to demonstrate and explain the application and the code clearly. - Q&A	Max 10
Grand Total	**100**

APPENDIX A- INTERNATIONAL MORSE CODE*

[*] By Rhey T. Snodgrass & Victor F. Camp, 1922, International Morse Code



A	● ▬	U	● ● ▬
B	▬ ● ● ●	V	● ● ● ▬
C	▬ ● ▬ ●	W	● ▬ ▬
D	▬ ● ●	X	▬ ● ● ▬
E	●	Y	▬ ● ▬ ▬
F	● ● ▬ ●	Z	▬ ▬ ● ●
G	▬ ▬ ●		
H	● ● ● ●		
I	● ●		
J	● ▬ ▬ ▬		
K	▬ ● ▬	1	● ▬ ▬ ▬ ▬
L	● ▬ ● ●	2	● ● ▬ ▬ ▬
M	▬ ▬	3	● ● ● ▬ ▬
N	▬ ●	4	● ● ● ● ▬
O	▬ ▬ ▬	5	● ● ● ● ●
P	● ▬ ▬ ●	6	▬ ● ● ● ●
Q	▬ ▬ ● ▬	7	▬ ▬ ● ● ●
R	● ▬ ●	8	▬ ▬ ▬ ● ●
S	● ● ●	9	▬ ▬ ▬ ▬ ●
T	▬	0	▬ ▬ ▬ ▬ ▬

APPENDIX B – EXAMPLE OF REPORT GENERATED

```
*** Decoded Morse Text
HELP US SOS SOS SOS
OUR SHIP HAS HIT AN ICEBERG
PLEASE HELP US
THIS IS A SOS
OUR SHIP IS SINKING
WE HIT AN ICEBERG
THIS IS AN SOS

*** Morse Words with frequency=> 5
...,---,...
[SOS] (5) [(0, 2), (0, 3), (0, 4), (3, 3), (6, 3)]

*** Morse Words with frequency=> 3
.-,-.
[AN] (3) [(1, 4), (5, 2), (6, 2)]
..,...
[IS] (3) [(3, 1), (4, 2), (6, 1)]

*** Morse Words with frequency=> 2
..-,...
[US] (2) [(0, 1), (2, 2)]
....,..,-
[HIT] (2) [(1, 3), (5, 1)]
---,..-,.-.
[OUR] (2) [(1, 0), (4, 0)]
....,.,.-..,.--.
[HELP] (2) [(0, 0), (2, 1)]
...,....,..,.--.
[SHIP] (2) [(1, 1), (4, 1)]
-,....,..,...
[THIS] (2) [(3, 0), (6, 0)]
..,-.-.,.,-...,.,.-.,--.
[ICEBERG] (2) [(1, 5), (5, 3)]

*** Morse Words with frequency=> 1
.-
[A] (1) [(3, 2)]
.--,.
[WE] (1) [(5, 0)]
....,.-,...
[HAS] (1) [(1, 2)]
.--.,.-..,.,.-,...,.
```

```
[PLEASE] (1) [(2, 0)]
...,..,-.,-.-,..,-.,--.
[SINKING] (1) [(4, 3)]

*** Essential Message
SOS HELP SHIP HIT ICEBERG SINKING
```

-- End --