Portfolio Assignment 2

ENDG 233 Fall 2021

Department of Electrical and Software Engineering
Schulich School of Engineering
Due: October 29, 2021 at 11:59 PM

The purpose of these assignments is to build a portfolio of successful programs that could be used to demonstrate your understanding of coding and Python. This assignment is worth 10% of your final grade.

Portfolio Assignment #2 – Learning Outcomes

- Accept and validate user input through varied menu options
- Process data according to specifications
- Iterate through varied values using compound data types
- Manipulate compound data types
- Print formatted output according to given specifications

Program Specifications

A cipher is an algorithm used for encrypting or decrypting information. You are being asked to design a terminal-based application for encoding and decoding text based on a provided cipher algorithm. Your application must meet the following design specifications:

- Your user interface should prompt the user to input the following information:
    1. Whether they would like to encode their text (1), decode their text (2), or end the program (0)
    2. Enter their text to be encoded/decoded
    3. Enter their cipher
- You may prompt for the input to be entered in any form or order you like, but be sure to give the user clear instructions.
- You must validate that the provided cipher consists of exactly 26 unique elements (no duplicates)
    - Unless you are attempting the bonus, you may assume that only alphanumeric characters will be entered (lowercase characters from a to z or digits from 0 to 9)
    - Each element of the cipher maps to a letter of the lowercase alphabet
    - Example: abcdefghijklmnopqrstuvwxyz → c5zf0hijklgn4bqrmtdvwx6za3
    - Print a message to the user to let them know that their cipher is valid.
- If the cipher does not meet the criteria, you must provide a message back to the user and allow them to re-enter their information without terminating the program.
- Your program must correctly encode/decode the text and print the result to the terminal (see example screenshots).
- The user must be able to encode/decode as many times as they would like. The program should only end if the user inputs 0 in the main menu option prompt.

- You may assume that only lowercase characters will be input as text to be encoded (no spaces, punctation, or numbers).
- You may assume that only lowercase/alphanumeric characters will be input as text to be decoded (as though you had already encoded it with your program).
- A screenshot with example terminal behaviour is included in this handout. You do not need to follow this format exactly as long as all of the specifications are met.
- Your code must include and use at least one dictionary.
- Your code must include and use at least three user-defined functions.
- Your functions may not access global variables.
- All functions must include docstring documentation.
- Your code must follow the conventions discussed so far in the course (names_with_underscores, four spaces for indentations, spaces between variables/operators, comments throughout, etc.).
- You may only use built-in Python functions that support compound data structures, user entry, or casting (such as len(), input() or int()). The only module you may import is string (optional).
- Your code will be run by the TAs as your end user using VS Code and a minimum of Python 3.9.
- FAQs about the assignment will be answered on the D2L discussion boards. Please check the boards for any clarifications before submitting.
- The grading rubric is also included in this handout.

Bonus

You may attempt an optional bonus validation task.

In addition to the validation above (unique 26 elements), you must also perform the following tasks for the input cipher:

- Check that the provided cipher is alphanumeric
- Convert all characters to lowercase
- Remove any duplicate values while retaining the provided order of elements (user may input more than 26 elements as long as after you remove duplicates, you still have 26 unique elements)
- Continue performing the encoding/decoding with the processed cipher

As above, if the cipher does not meet the criteria, you must provide a message back to the user and allow them to re-enter their information without terminating the program.

Assignment Tasks

- Make sure to watch the video lessons for Weeks 1 – 6 and review the corresponding active learning content. You may find Week 7 helpful as well.
- You do not need to submit a flowchart, but it is highly recommended that you start with a flowchart to plan your logic.
- Open VSCode and start a new terminal. Make sure that your virtual environment is activated.
- `encryption.py` is provided as a starting point. Fill in the header with your own information and write your program in this file.
- Remember to test your program execution via the terminal.
    - For example, use `python encryption.py` or the corresponding command for your operating system.
- Take a screenshot of your successful program execution that shows:
    a) Your virtual environment is active when running the program
    b) Your program successfully computes the required functionality (see provided output examples)
- Submit the following items to the Assignment 2 D2L dropbox:
    - Your final 'encryption.py' file with your name in the comment header (do not change the file name)
    - Your execution screenshot (e.g. `00121343_assign2_output.png`)

Assignment 2 Rubric (26 marks + optional bonus 3 marks, 10% of overall grade)
Your code must successfully run to be given full marks. Code that does not execute may be given partial marks for some criterion listed below.

Commenting and Syntax (7 marks):

- (1) Your name must be included in the file header
- (2) Comments must be included throughout the code to explain the functionality
- (2) All functions are fully documented using docstrings (including summary, parameters, and return values)
- (1) All variables and functions have clear and useful names that use lowercase words separated by an underscore
- (1) Code is clearly indented and spaces are included between variables and operators
- One mark will be deducted for each error or missing component, up to a maximum of 7 marks

Code Structure and Semantics (4 marks):

- (3) Solution contains and uses at least three user-defined functions
- (2) Solution contains and uses at least one dictionary
- (1) User-defined functions do not access global variables
- One mark will be deducted for each error or missing component, up to a maximum of 4 marks

User Interface and Functionality (6 marks):

- (1) User is given clear guidance on how to enter the input values
- (1) Program allows the user to choose either encoding or decoding functionality
- (1) Program runs continuously until the user chooses to terminate
- (1) Program accepts two strings (text and cipher)
- (1) Program checks that the cipher has 26 unique elements and prints a message to the user.
- (1) If an invalid cipher is provided, user is prompted for re-entry without terminating the program
- (1) The program outputs a lowercase string with no spaces between the letters
- One mark will be deducted for each error or missing component, up to a maximum of 6 marks

Execution (9 marks):

- Example test cases are provided in the screenshot below, but you may provide your own in your screenshots
- (5) Screenshot of successful execution is shown
    - Your screenshot should include all successful functionality
        - 1) encoding, 2) decoding, 3) handling incorrect cipher input, 4) program termination
    - Your screenshot must also show that your virtual environment is active
- (4) Your program will be executed to test the following cases:
    - Encoding a string of lowercase letters with an alphanumeric cipher
    - Decoding a string of lowercase letters and numbers (as though the text had already been encoded by your program)
    - Entering an invalid cipher such as “99abk0”
    - Continuous input terminated by inputting 0 in the menu
    - All students will have their code tested with the same input values
- One mark will be deducted for each error or missing component, up to a maximum of 9 marks

Optional Bonus (3 marks):

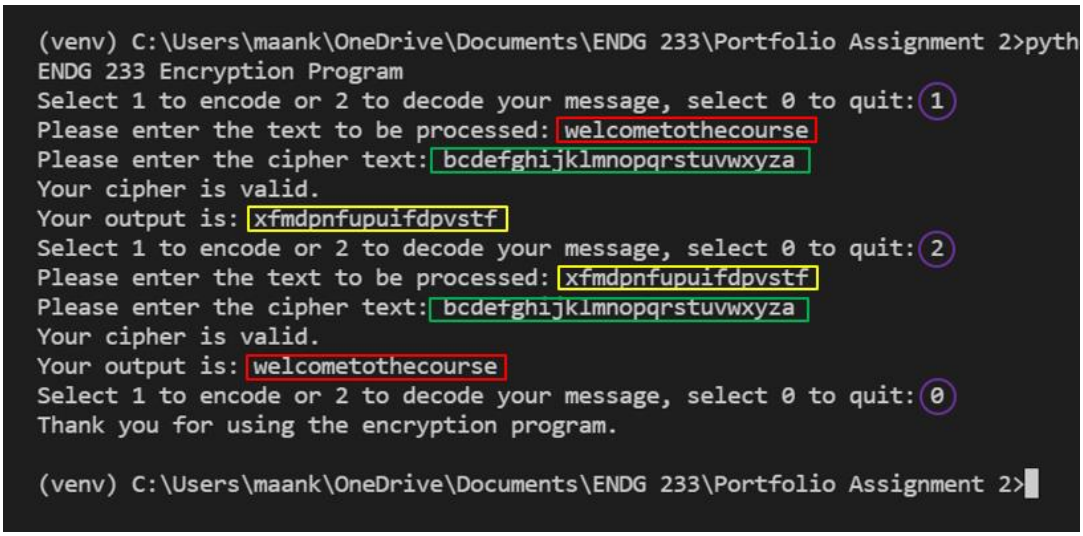
- To receive the bonus marks, you must do the following
- The above screenshots should include successful execution of the bonus
    - Demonstrate that your program can identify when a cipher is not alphanumeric
    - Demonstrate that your program can convert any uppercase letters in the cipher to lowercase
    - Demonstrate that duplicate values are removed while maintaining given element order
- Your program will be executed to test the following cases:
    - Encoding a string of lowercase letters with an alphanumeric cipher that contains uppercase letters and duplicates
    - Decoding a string of lowercase letters and numbers with an alphanumeric cipher that contains uppercase letters and duplicates
    - Entering an invalid cipher such as “c@!!0hijklgn4BQrmtdvwx6za3”
    - All students will have their code tested with the same input values
- One mark will be deducted for each error or missing component, up to a maximum of 3 marks

Assignment 2 Example Output

You may choose how to prompt for input and display output as long as it is clear to the user.

A sample run of the program is shown below, where:

- Purple circles show user selection
  - Initial input selection 1 for encoding a message
  - Second input selection 2 for decoding a message
  - Third input selection 0 for ending program
- First red box shows the original message to be encoded
- First green box shows the 26 character long string representing the cipher
- First yellow box shows the encoded message
- When reversing the process (decoding the generated encoded message):
  - The content of the yellow box is entered for processing
  - The same cipher is given to decode the message back to its original text
  - And finally we get the output which is the original message in the red box



Another sample run with an invalid cipher for first and second attempts then a valid cipher on third attempt is shown below.

```
(venv) C:\Users\maank\OneDrive\Documents\ENDG 233\Portfolio Assignment 2>pyt
ENDG 233 Encryption Program
Select 1 to encode or 2 to decode your message, select 0 to quit: 1
Please enter the text to be processed: welcometothecourse
Please enter the cipher text: dfgfdshjaetjlvzxp
Your cipher must contain 26 unique elements of a-z or 0-9.
Select 1 to encode or 2 to decode your message, select 0 to quit: 1
Please enter the text to be processed: welcometothecourse
Please enter the cipher text: qergdmlvlflksdfvolssd
Your cipher must contain 26 unique elements of a-z or 0-9.
Select 1 to encode or 2 to decode your message, select 0 to quit: 1
Please enter the text to be processed: welcometothecourse
Please enter the cipher text: bcdefghijklmnopqrstuvwxyza
Your cipher is valid.
Your output is: xfmdpnfupuifdpvstf
Select 1 to encode or 2 to decode your message, select 0 to quit: 0
Thank you for using the encryption program.
```

The output for the bonus is identical but the cipher validation algorithm working in the background is different to handle cases with uppercase letters and validate that the cipher is only composed of numbers and alphabets while removing duplicates.

```
(venv) C:\Users\eamarasc\Dropbox\Teaching\ENDG 233\Fall 2021\Portfolio\Assignment 2>python encryption.py
ENDG 233 Encryption Program

Select 1 to encode or 2 to decode your message, select 0 to quit: 1
Please enter the text to be processed: welcometoschulich
Please enter the cipher text: tHHequUickbrownfoxjump3doverthelazyDog!
Your cipher must contain 26 unique elements of a-z or 0-9.

Select 1 to encode or 2 to decode your message, select 0 to quit: 1
Please enter the text to be processed: welcometoschulich
Please enter the cipher text: tHHequUickbrownfoxjump3doverthelazyDog
Your cipher is valid.
Your output is: auwexnudx3ekvwbek

Select 1 to encode or 2 to decode your message, select 0 to quit: 2
Please enter the text to be processed: auwexnudx3ekvwbek
Please enter the cipher text: tHHequUickbrownfoxjump3doverthelazyDog
Your cipher is valid.
Your output is: welcometoschulich

Select 1 to encode or 2 to decode your message, select 0 to quit: 0
Thank you for using the encryption program.
```

Additional suggested test cases:

String to be encoded: tellmeandiforgetteachmeandirememberinvolvemeandilearnbenjaminfranklin
Cipher: bcd5fghijk60nopq1stu43xy2a
Result: uf00nfbo5jgpshfuufbdinfbo5jsfnfncfsjo3p03fnfbo5j0fbsocfokbnjogsbo60jo

String to be decoded:
ui2c25uzoenp5uc2zvujgvmuijoh5joui2xpsme1zoopuc2522ops2w2oupv1i2eui2bnv5uc2g2muxjuiui2i2zsu
i2m2ol2mm2s
Cipher: zc1e2ghij6lmnop4rs5uvwxyba
Result:
thebestandmostbeautifulthingsintheworldcannotbeseenoreventouchedtheymustbefeltwiththehearthel
enkeller