Contents

[Aims and objectives 2](#_Toc88471181)

[Analysis 3](#_Toc88471182)

[Design 4](#_Toc88471183)

[Developmental Testing. 5](#_Toc88471184)

[Testing 6](#_Toc88471185)

[Evaluation 1](#_Toc88471186)

Tkinter Encryption Project

Harry irving

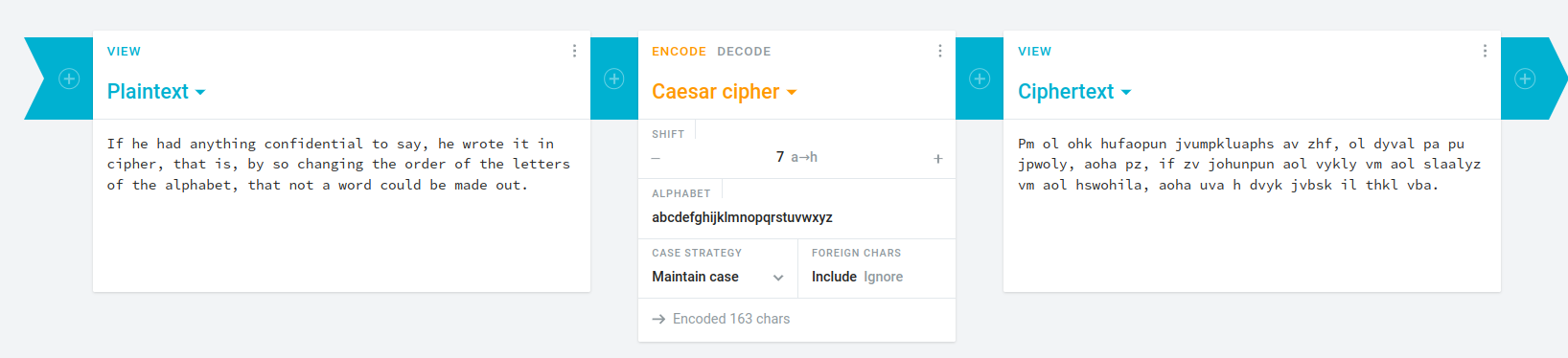
## Aims and objectives

## For me, the aim of this project was to create a modern gui application using tkinter that features several encryption algorithms. I have used tkinter in the past, however only to make a very basic gui application with an inbuilt calculator. The gui of the calculator project wasnt exactly good looking, the main focus of that project was on the actual calculator part.

## For this project the gui was the focus for me as i had never managed to make a gui that looked modern.

## Analysis

Some of the inspiration for this project came from this caesar cipher website



This website allows you to input text to put through a caesar cipher. To build on this project, i decided to make my project work with files. You can input the name of a .txt file in the input box on my app and it will encrypt the contents of the file and output it to a new file

For My interface, I divided it into groups of widgets divided by their functionality within the program.

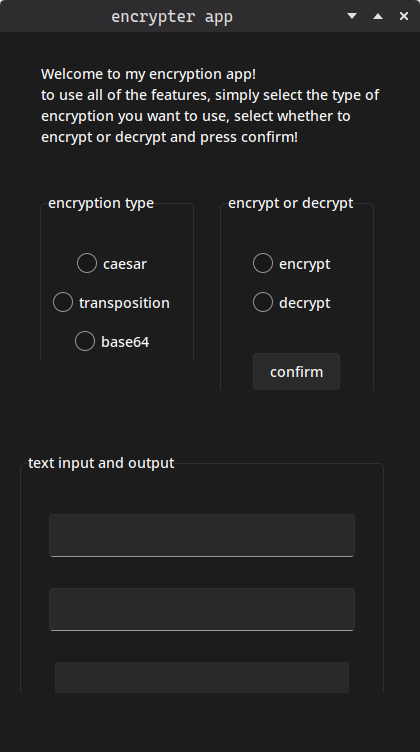
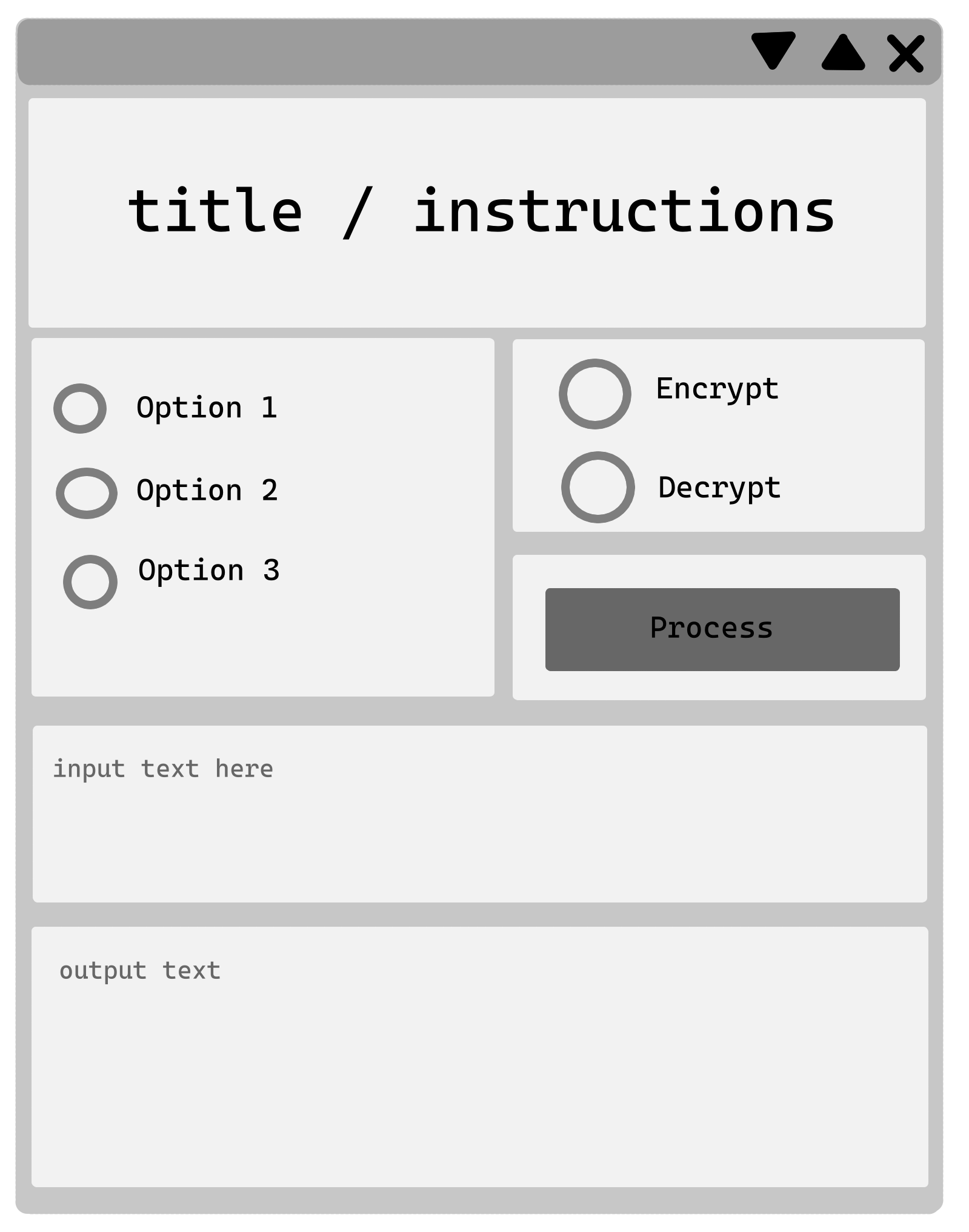
The program was initially intended to be in the form of a roughly 3:2 height:width window that would be able to fit easily on most desktops. This meant that the window would have to be compact.

The top section is simply a short paragraph detailing how to use the program. From there the middle section was divided into a left and right half.

The left half is the selection box for which cipher should be used whereas the right box features the option to encrypt or decrypt and also the button that actually fetches the input and feeds it to the processing functions.

Finally, the bottom section of the program is the input box where the text to be sent to the encryption algorithm should be typed and the output box, where the encrypted or decrypted data is returned to the user.

Within this program, each of the sections is defined by a different block of code inside a constructor function which builds the majority of the interface. I separated this part from the init function to keep the code more understandable as it shows what the program is doing



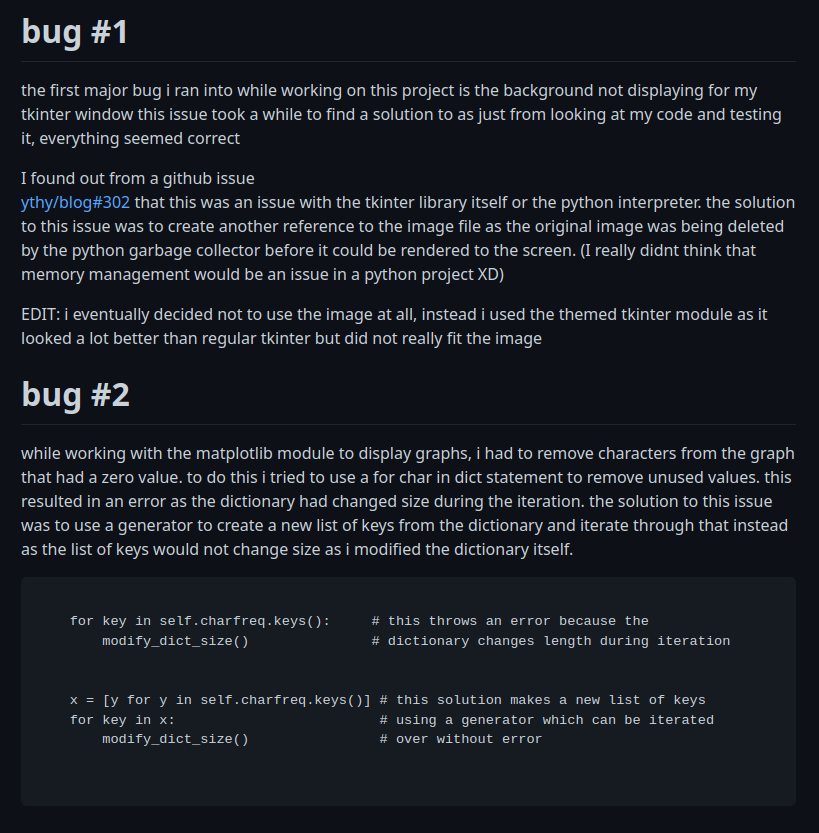
As you can see from the screenshot of the finished GUI, the application ended up looking pretty much like my initial plan. The only noticeable difference between the plan and the final GUI aside from the theming is the bottom section. I used two input boxes instead of one because i had forgotten at the time that the user would need to be able to input a key.

In my intial gui design, I implemented a background image, however I felt that it didnt fit with the UI elements I wanted to use as they would simply cover over the background. As tkinter widgets cant have transparent backgrounds, this meant that the image would just be covered over by the boxes of the GUI which would have looked messy so I opted for the simpler more minimal dark grey background.

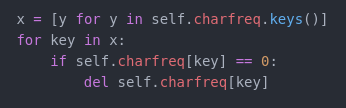
For the theming, I used the tkinter.ttk module along with sv\_ttk, a great modern theme for tkinter which is what really made the gui look clean and minimal. Tkinter.ttk is simply a set of widgets to replace the majority of the default tkinter widgets. Instead of being made just to work, they have support for themes.

## Developmental Testing.

As you create your program you will make mistakes and need to correct these. Show evidence of at least 5 developmental tests. Print screen your code, explain what went wrong and how you will fix it.

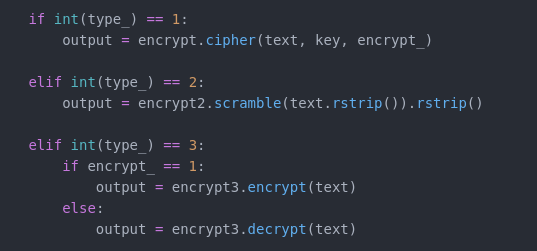


This is the working code, i had previously tried to simply iterate over the keys, however removing the keys from the dict while it was being iterated over caused an error



Fixing this problem simply required the type\_ variable to be converted to an integer before it was compared.

This error had been covered up by the fact that I had used an else statement which resulted in the algorithm giving the wrong output instead of none at all



## Testing

Complete the following testing table, you will split the table and have lots of images.

For the caesar cipher i used

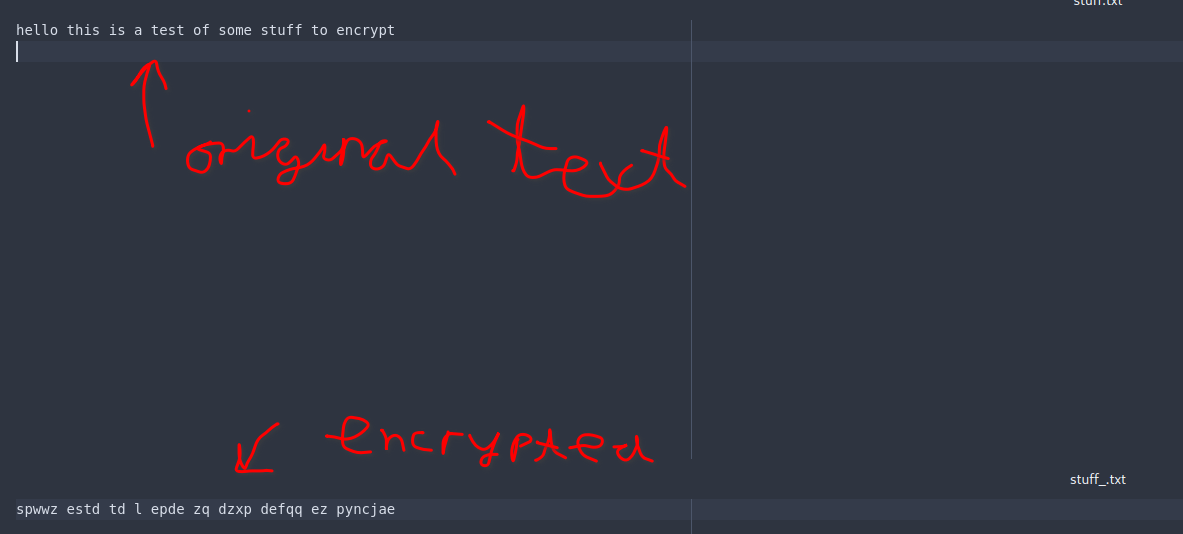
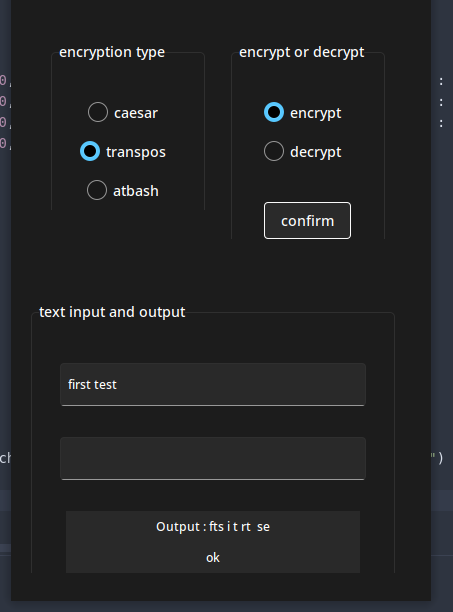
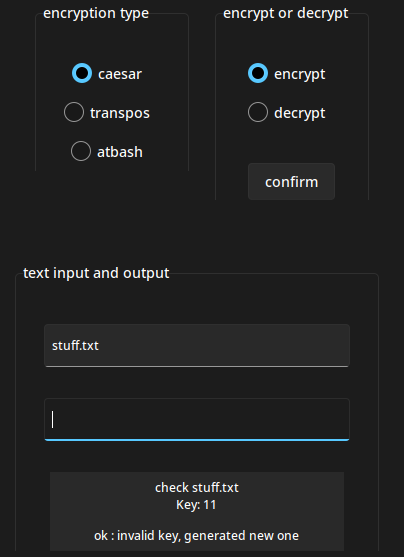
<https://cryptii.com/pipes/caesar-cipher>

To check my outputs with the correct ones.

For the transposition cipher i encrypted and decrypted the text.

If the text

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | Sample data | Expected result | Actual result | Comment |
| Test #1 for caesar cipher | Text: Hello there  Key: 6 | nkrru znkxk | nkrru znkxk | This test worked correctly |
| Caesar cipher test 2 | Text: This Test will hopefully work Properly  Key: 20 | nbcm nymn qcff bijyzoffs qile jlijylfs | nbcm nymn qcff bijyzoffs qile jlijylfs | This test also worked correctly  In this test i used a key that would overflow back to the start of the list and it worked fine. |
| Caesar cipher test 3 | Text: final test  Key: 100 | bejwh paop | bejwh paop | The key was significantly larger than the maximum value accepted however it still gave the correct output  The real test is if it decrypts properly |
| Reverse caesar #1 | Text: nkrru znkxk  Key: 6 | hello there | hello there | This simple decrypt initially failed but after adding a line of code which preserved the spaces, it worked fine. |
| Reverse caesar #2 | Text: nbcm nymn qcff bijyzoffs qile jlijylfs  Key: 20 | this test will hopefully work properly | this test will hopefully work properly | This test worked as expected |
| Reverse caesar #3 | Text: bejwh paop  Key: 100 | final test | final test | This test worked as expected as well. |
| Transposition #1 | Text: first test | fts i t rt se | fts i t rt se | This test worked as expected |
| Reverse Transposition #1 | fts i t rt se | First test | First test | The reverse of the test also worked as expected |
| Transposition #2 | this is a proper test to see how this program performs | ta or h tstgf ipeehro srseiar ot smm ip h s setopp rowre | ta or h tstgf ipeehro srseiar ot smm ip h s setopp rowre | This longer test works correctly |
| Reverse Transposition #2 | ta or h tstgf ipeehro srseiar ot smm ip h s setopp rowre | this is a proper test to see how this program performs | this is a proper test to see how this program performs | This reverse test also works correctly |
| File test #1 | File with unencrypted text | Text is encrypted using the correct cipher | Text is encrypted using the correct cipher | This text was successful  Outputted file with the same name but with an \_ at the end |
| User input compensation test #1 | Key not entered for caesar encryption | Program generates new key and encrypts | Program generates new key and encrypts | Test was successful |
| User input compensation test #2 | Key not entered for caesar encryption | Returns error to user: invalid key | Returned invalid key error to user | Test was successful |
| Atbash cipher #1 | firsttest | firsttest | firsttest | Test was successful  (test produced the original input after encrypting and decrypting again) |
| Atbash cipher #2 | Does this work properly | Does this work properly | Does this work properly | Test was successful  (test produced the original input after encrypting and decrypting again) |
| Frequency graphs | Input “hello” to the caesar cipher and “3” as the key | Frequency bar graph: e, h, o have a height of 1 and l has a height of 2  Frequency bar graph, h, k, r have a height of 1 and o has a height of 2 | Frequency bar graph: e, h, o have a height of 1 and l has a height of 2  Frequency bar graph, h, k, r have a height of 1 and o has a height of 2 | Test was successful |



## Evaluation

Produce a detailed evaluation of the effectiveness your solution, you should consider the following.

* How does it compare to the sites you researched at the beginning
* Have you met all the aims and objectives? You should refer to these by number
* What features of Python have you used and how did this help? Or not?
* What you have done well and what you would like to change if you were to repeat the task.

In my opinion, this project came out well, I have succeeded in creating a modern and minimal gui that can perform several different ciphers on a given input. The program can handle files, this feature was a nice extra, but not necessary for the core functionality of the project. This project has taught me a lot about GUI design and interaction; the last time i tried making a GUI in tkinter, it looks basic and cobbled together, on the other hand this project looks clean and unintrusive. Throughout this project i learned a few new skills, one of which was generators. Generators are small pieces of code used to turn an iterator into a list. An iterator is a function that can be called with a different output each time, for example a list.

If i were to repeat this project, I would add the option to use more ciphers. I feel like three basic ciphers is too limited for a full gui application. Even though this was the requirement of the project, remaking this project to be more powerfull / polished would require me to add more ciphers and extend the functionality.

Another thing that I would do if I were to remake this project is make a context menu for the app, this would bring up a help window which would explain how to use it, as at this point, its not completely clear whether you need to input a key for certain ciphers. For example, only the caesar cipher actually requires an encryption key