Coursework Report

Steven Pyper
40319882@napier.ac.uk
Edinburgh Napier University – Web Tech (SET08101)

Contents

1.Introduction

The main scope of this coursework will be to build a website that contains at least 2 ciphers. These will take inputs from the user that allow them to specify the key(if the cipher takes one) and the text they would want to convert. These must also be reversable so that the user can uncypher the text after they have ciphered it.

1.2 Caesar Cipher and Rot13

The ciphers I have chosen to do are a simple Caesar cipher. From reading (wikipedia, n.d.) about this cipher it should be somewhat simple to implement but also is a useful cipher that can be explained well to people who aren't tech savvy, this allows it to be a good first option for the website as a introduction to ciphers as well as a good introduction to JavaScript due to it in theory not being complicated. In addition to the Caesar Cipher the user will be allowed to use a rot13 cipher which is essentially just the Caesar but with a fixed shift of 13, this will be made after the Caesar as it should take less code to go that way rather than reverse it, this is another simple cipher for the user to get introduced to and is there for a good early choice.

1.3 Direct Substitution Cipher

Another cipher I will implement is a direct substitution cipher, I believe that this will be a more complicated cipher but also will be much more interesting in terms of what the user would be able to do with it. As it relies on the user specifying their own alphabet(key) this will allow them to choose their own alphabet to start with, allowing it to work with more than just English(like the Caesar Cipher) and then create their own, but also allow them to cipher it multiple times and as long as they have all the keys in the right order get back to where they started after multiple deciphers (Wikipedia, n.d.). Due to all of this it should be a more difficult cipher both to code and for the user to understand, while researching about this and realising that it will be difficult for the user to understand I have decided that although all ciphers will be available on the first page, each cipher will have its own page explaining how to use the cipher and how it works. This will allow the user to easily understand how to use the cipher and maybe even learn more about it.

1.4 Morse Code

I will also implement Morse code this is the wildcard out of all the other ones as its technically not a cipher but is still very interesting and fits with the idea of the website of converting text, this also allows for the use of audio as it should be possible to play the sound of the Morse to the user through the browser. This should be the most difficult to implement as it will have to deal with time between each character and the time between each word. This should still allow the user to convert between text and Morse and play audio going both ways(so if the user enters Morse to convert to text, or enters text to convert to Morse). I would like to try to stick to standard (Wikipadia, n.d.)(See Appendix A) however if this would take to much development time as long as it is very clear what the difference between dots, dashes, spaces between characters, and spaces between words I will be happy with the implementation.

2. Software design

The initial design will include just one page(index.html) which will contain all the ciphers and information about them. This information will contain a little idea behind the cipher and what it does, with the possibility of having a little explanation of how to use it. This will also include an image that might help the user to understand what the cipher does by showing what it does as well as writing what it does. This however most likely will be placeholder and only be used as a testing page which will allow me to see how much room will be needed as well as make changes to how the output will be displayed.

There will be a page for each of the ciphers except for the Caesar and rot13 which will be together as they are very similar and can be explained together which will help the user with the simple version first as well as making it tidier

2.1 Pages

All pages will be accessible from every other page (can go from index to anywhere and anywhere to index)

(See Appendix B for NavMap)

(See Appendix C for home page layout)

(see Appendix D for Other Page Layouts)

Unsure where the Navigation bar will go, it will either go straight down the left side of the page or below the heading depending on which looks nicer at the time of creation, therefor none of the layouts include the navigation bar!

2.1.1 Home Page(index)

The final design will be to have the first page only explain some of the strengths and weakness and the history of each cipher, this page will then contain links to each of the individual ciphers which the user would be able to choose either from a navigation bar or at the end of the explanations of the ciphers. This will allow the user to then decide what cipher they might want to use. This page will most likely contain 3 rows and columns, one row for each of the ciphers (with Caesar and rot13 together) that way everything can be split up in a way that makes sense, the name will include a link to the page that has the cipher. One column will be used as the heading for the cipher name and will also give little run down about the cipher which will include what it does and a little history about it. The next column will include the pros of using that cipher as well as good use case scenarios for using it. The final column will include the negatives of the cipher and the ways that it can be easily broken (both in terms of user input and by cracking methods).

2.1.2 Caesar and Rot 13

Due to how similar these two are they will both be put on one single page. Rot13 can help explain the Caesar cipher and as are likely to either use the same base coding, or similar, so it makes sense to keep them together on one page. The page will consist of two rows and two (with an extra row which will be single column for the output). The first row and column will include information about the cipher and how it works, this will include information like how it moves the alphabet by a certain

amount. The second column will include how to use the cipher which will be dependent on how it ends up being coded. For instance, the key may need to be converted to negative by the user when they want to decode, or I could make the code deal with this while decoding. This will help the user with how to use the cipher and explain what each input box and button will do. On the second row the first column will include a selection box (drop down menu) which will allow the user to decide between whether they would like to encrypt or decrypt their message. Depending on what the user selects different messages will appear in the next column which will ask the user to enter information and then click a button which will then run the inputs through the program. The outputs from the program will be displayed on the extra row.

2.1.3 Direct Substitution Cipher

This page will follow the same style as the Caesar and rot13 page in terms of layout however the information will include more detail about helping the user with the cipher as it is more complicated due to needing more inputs than the other ciphers. It will allow the user to encrypt their text based off the original alphabet they enter (which will most likely be the full English alphabet) and the new version of the alphabet (their own modified version), this will be the same for decrypting where they will enter the same information but getting the decrypted text back.

2.1.4 Morse Code

This page will also follow the same style as the previous ciphers however will include extra buttons while converting which will have a button to play audio (as well to stop it from being played). This page will include information about Morse code and tell the user if it applies to standards or not, as well as how to use it. There will be a choice between converting to Morse and converting to text, and during both there will be an option to play either the text (which will be converted to Morse) or the Morse that was entered as audio output as well as text output.

2.2 JavaScript

there will be a script for each cipher on the website and there will also be a script that will allow the user to decide how they would like to use the cipher (encrypt or decrypt). On the test index there will also be a script that will allow the user to decide between each cipher however this will most likely go unused in the final version as each script would be on its own page meaning that the user won't need this functionality.

2.2.1 Caesar Cipher and ROT13

This should be a fairly simple program to write due to the fact that I have wrote similar programs before, from this I know that there are two ways to do it, one which is referenced on the Wikipedia page for Caesar ciphers is to do it numerically, this method means using maths to calculate what the remainder would be from adding the original character number(with a being 0) with the key, this would mean that if the character z was shifted by 1 it would become the number 26 which would leave no remainder so the number would become 0 which would reference a. this is a system that does work however the flaw is that it can end up being more complicated to debug when things don't work. therefor I will be using the other method where you have one large string that is your character set (which is just all the characters A-Z), using this it is possible to get a character form the user, get where it would be in the character set(so a would still be 0) and you can then apply the key to it, this allows you to have more control over what the script should be doing depending on what number the character would be converted to, as an example the number could go below 0 or above 25 and these would need to be dealt with differently. This script will have 3/4 different functions, this will be dependent on if there will be a "main" function which will choose which type you are using or if there will just be the functions for dealing with encryption, decryption and cracking (where the key is not known). The reason for keeping the main function would be that you could have the button for rot13 still call up the Caesar function but pass in a key of 13 instead of asking the user to enter a key.

2.2.2 Substitution Cipher

The substitution cipher should be more difficult than the Caesar as you are having to deal with two different alphabets, one which will be the original alphabet (the English alphabet for example) and the modified alphabet which user can define themselves. It will include functions for encrypting and decrypting the messages although they will likely be very similar but with different references. The simplest way I found to two link two alphabets together would be the use of maps (alligator.io, n.d.). this should allow me to have a function that will link the two separate alphabets together and allow for the program to look through each character entered, compare it to the original alphabet and then find the value that would be returned from that key in the map. This should allow for an efficient and fast way to take the users inputs and convert them to the new alphabet. There may need to be more functions to change the alphabet depending on if its being encrypted or decrypted although I aim to only have one function doing all the setting up of the map

2.2.3 Morse Code

The Morse code JavaScript will most likely be the most complicated one of them all, this is because it is not a standard switch where you convert one character into another single character. It will mean that you must convert a single character into multiple, for example "a" would become ".-" , this becomes further complicated when you add multiple characters and then multiple words as there has to be spaces between characters but also spaces between words so you will have to distinguish between the two, this will impact the code heavily as it will need to know what the difference is between them so that when it converts it knows where each letter starts and ends, where each word starts and ends and where actual spaces should be. There is also the added difficultly when then converting this to audio, there is a standard in place of how long each sound should last which must be represented in the code, this means learning about the audio api and just getting a sound to play and then getting it to play and stop when it should, the aim will be to have it follow the standard but if that is not possible then I will get it to a point where it is fairly easy for the user to figure out the difference between characters, words and true spaces between words. Getting the text converted to Morse and then back to text will greatly benefit from using maps like the substitution cipher, this is due to the

fact that we have two predefined languages to convert between, it will be possible to have a map that links the exact Morse linked with its respective English character (".-","A") and the opposite is also true, therefor two maps will likely be created that are opposites of each other, one will be used when converting to text and the other when converting to Morse. This should help simplify some of the converting as it means the main chunk of code will be used to parse what spaces should be and where each character starts and ends. Finally, for Morse there will have to be a function dedicated to playing the sound, this will likely call one of the converting functions depending on what the user has input, in theory it should then be possible to take what is converted and have a loop that goes through each character and plays for different lengths depending on which character is read, from research it should be possible to do this with the setInterval or setTimeout functions. It is most likely that I will use an oscillator as it can be set to a certain frequency and by using a gain node it should also be possible to make the sound a comfortable level. The user should also be able to choose their own volume

2.2.4 Change Cipher/ Change Method

These scripts will be used to change the information on the page depending on what the user is wanting; the change cipher script would allow the user to choose between all the ciphers (and Morse code) and the change method will allow the user to choose how they would like to use that cipher(encrypt/decrypt), the change cipher method will likely only be used for testing while there is only one page as each cipher will have its own page in the final design meaning that the user has already made that choice by going to the page. The change cipher function will add in a new selection box onto the page which contain the options for change method for the user to select (therefor if they choose Caesar there will be 3 options in the method selection box). The change method function will allow the user to choose their method and then it will add all the required information onto the screen such as new input boxes and buttons that are required for the specific task. This will be required on each page that has a cipher.

3 Implementation

3.1 Implementation of design

The Design of the Website was implemented to an acceptable level, I am not happy with it, but it has everything it needs, although it is definitely lacking in aesthetics. The main home page contains a lot of information about each of the ciphers as well as links to websites that you can read up more about what is being spoke about, follows the structure previously laid out with some minor adjustments(the nav bar which is the same for all pages) as well as changing from whites and blues to greys, browns and yellows.(See Appendix E)

Also, all the ciphers mentioned above were implemented with most of the functionality discussed. They follow a similar layout to each other, as well as have a similar colour scheme to index. (See appendix F)

3.2 implementation of Code

3.2.1 Implementation of Caesar Cipher and Rot 13

The Caesar cipher and rot13 were both implemented very close to the plan laid out, all functionality is present allowing the user to cipher, decipher and crack text that is entered. I did implement it using the second method that was discussed and as such it does function correctly under normal circumstances allowing the user to choose between encrypting, decrypting, cracking, a rot13 encryption and rot 13decryption. In the end it was possible to use a main function for deciding what was going to happen and then 3 functions (encrypt, decrypt, crack) which did the actual work. (see appendix g and h)

3.2.2 Implementation of Substitution Cipher

The substitution Cipher was also fully implemented allowing the user to enter any alphabet they want and compare it to the original and get text encrypted and decrypted. The code was much shorter than expected although there was some trouble at first getting it to deal with spaces constantly (see appendix i and j)

3.2.3 Implementation of Morse Code

The implementation of Morse code was initially very easy due to having planned to make use of maps which led to the encryption and decryption part being very quick to make. The troubles that were planned for with the program having ot interpret spaces was more difficult but not for the expected reasons, the browser would take away any double spaces or tabs leading to having to find a new solution of adding spaces, this meant I had to find a alternative way of dealing with double spaces which led me to finding out that using   does not get shrunk by the browser. This meant slightly changing the program to look for   and outputting it instead of double spaces but it allowed the program to run quickly. Another major difficulty was getting the sound to work with correct timing(initially getting it to work at all was difficult) but eventually the implementation of it is mostly correct, it still has some weird audio drop outs or cut-offs(that happen infrequently) which is due to the use of setinterval although I wasn't sure a better way to implement it. Also, the code is not up to standard although it is very easy for the user to distinguish between all the options. (See appendix k)

4 Critical Evaluation

4.1 Comparison to requirements

Believe that the website matches most of the requirements that were set in the descriptor. This is due to the fact that it includes two ciphers and an encoding scheme. However, the index.html does not include any of the requirements laid out, this was a design choice however as it was a better idea to move each cipher to its own page and use the index.html page as a introduction to each of the ciphers and allow the user to make a choice based on what they have read about it. This means that each page had its own cipher which allowed all the information on the respective pages to be able the cipher. Each cipher page fits the criteria by allowing the user to enter text depending on what is required by the cipher, a place to show outputs depending on weather or not is was encrypted or decrypted would change the message. There is also a button

that the user must press to do the computation and there is a way to recover messages. However due to this there is no way to choose between ciphers, what I should have done is have one page that includes a way to change between them (as the code is in place to do this from the testing early on) that way more experienced users could swap between them very quickly if they already know what they are doing and new users can still learn by going to the individual pages and reading about them. The main flaw I find with the website is the aesthetics, it is very basic, and I think that some of the design choice including the colours and the way I dealt with images leads to a website that seems cluttered and not very pleasing to look at.

4.2 Possible Improvements

There are many improvements that could be made to all areas of the website. The results from the design are very poor (and may even be replaced before this assignment is handed in), there are a few issues with the ciphers mainly revolving around lack of input validation and the audio for the Morse code works but lacks consistency as sometimes it works flawlessly and sometimes it cuts out a lot.

4.2.1 Design

Since writing the implementation part of the website I decided that the design was just too poor and as such decided to see what little tweaks could be made to make it look slightly better and as such, I did make a change. The index page has changed very little apart from making the background colour slightly brighter which made the page feel less bland, however the cipher pages changed a lot, instead of each individual box having colour the entire grid has been changed to one brighter version, this also leads ot the page being much more interesting to look at even though it is a minor improvement. Overall next time I should spend more time planning how the site should look and think about colour schemes more and the impact that it will have with text, images and links.

4.2.2 Morse Code

Although I am unsure if there is a better way to write the Morse code audio section, I feel like there must be a better way to implement it, the way that I implemented it feels too much like a bodge and has flaws that I don't understand, I believe these were caused by a lack of depth in understanding about JavaScript and the functions that it contains so I would use functions like setinterval even though there may have been better way to get it working through the web API. I also feel on the page itself I should have included an image showing the conversion so that less experienced users would be able to look at the image and then write then own Morse that way. However, the dealing with the conversion of Morse itself I am quite happy with as it seems to be instant conversion both ways and no errors appear (unless they are down to the user)

4.2.3 Lack of Input Validation and error correction

Over all the scripts that have been wrote there is no input validation, this is something I always forget about as I assume that the user will be competent enough to enter correct values, due to the fact the ciphers focus around using trees if the user enters a value that is not valid it will most likely return undefined which to the user does not really help, I should either have had a way to check the inputs of the user or at the minimum had a way of checking outputs that showed errors and had a way to explain them to the user so that if an accidental input happens the user will know why it didn't work.

4.2.4 More ciphers

Although enough ciphers were implemented from the descriptor, I wish I had implemented either more ciphers or more varied ciphers, by the time I was done writing them they all felt like the did a very similar job (it is possible for all the ciphers(excluding Morse) to do the exact same job of a rot13) and as such it feels like the website is lacking even though it isn't. it also feels like I did not learn as much JavaScript due to them being similar, even though the code is mostly different there are enough similarities between them which make it feel less interesting to work on. A big change I would like to have made was make the ciphers more interactive. The example that comes to mind is the Morse code page, there is a webpage online that lets the user use their keyboard and based on when they push and release any button it will take it as a dot or dash or a space, this seems like a much more interesting implementation of Morse code that I wish I had though about implementing during the planning stage, another idea after creation was having the option for the user to click on buttons on screen which would represent the Morse code(again similar to the way the website displays the Morse code alphabet) and let them interact with that to either encrypt or decrypt. (bennadel, n.d.)

5. Personal Evaluation

I believe that I learnt a lot during this project although I wish I had started much earlier so that I could have implemented a lot more as well as had more time to learn more about the design elements. I learnt a lot about JavaScript and some of its quirks as well as issues that can arise from a browser that wouldn't typically appear from just a coding stand point (like the browser shrinking spaces). There were quite a few issues that arise from JavaScript acting in some unexpected ways, one of the main issues I kept running into was when numbers were being calculated wrong and I couldn't figure it out what it was doing, eventually I figured out that they were reading as strings and as such were being concatenated together rather than being added together. I found that the fix for this is to use a + sign before each and this would fix the issue as it would be treated as a number. (ChaosPandion, n.d.) I also feel like I learnt that the use of maps in almost all languages is useful, this is something I used in python and java (although different versions of maps) and they all seem to be very efficient at getting groups together and can lead to saving a lot of time programming. Overall, I am somewhat happy with my coursework. I still feel like the design elements are lacking however this is something that I am poor at overall, designs are not something I'm good at I typically go for the approach of make something that's efficient and works rather than looks nice. With that in mind I am happy with most of the programming, excluding the lack of input validation the code is in a state that I am happy with and believe will work under most

circumstances, it also does what its required plus a little more. However next time I am doing a project like this I will start earlier allowing for a deeper knowledge of what I will be using as well as well as allowing for more time to plan and to make changes at the end, as this time I ran out of time to redo the design again as it had already been restarted multiple times as I was unhappy, and still am, with the entire design. I feel that the pages ended up either feeling empty or cluttered but never quite right.

I also feel like I should have wrote way more comments in the code and any free time from now till due date will be spent adding comments, most of the code is fairly readable as is for a programming however there are some sections that are more confusing and as such I feel would benefit from the use of comments, normally I have a tendency to over comment my code and as I stopped myself from commenting everything as I went it lead to me forgetting to comment the vast majority of code(which by the time this is handed in that should have changed) . Also mid-way through the coursework I realised that JavaScript does not require ; which means that out of habit sometimes I would write the ; and sometimes I wouldn't leading to code being beside each other that looks like its in completely different formats, I will also try to fix this before the coursework is officially handed in

References from code

There were not many times in the code I used a bulk of what I found online however there were a few times that I could take someone's concept and change it, here I will link to all of those that do not have a single point in the code and as such referencing inside the code seemed pointless

SetInterval (w3Schools, n.d.), css grids (css-tricks.com, n.d.), CharAt and IndexOf (Barot, n.d.), Custom select boxes(such as having an option show then disabled (kaspergotcha, n.d.),

All audio (this includes oscillator page (Sheppy, n.d.)

References

alligator.io. (n.d.). *https://alligator.io/js/maps-introduction/*. Retrieved from https://alligator.io/js.

Barot, V. (n.d.). *http://raovishal.blogspot.com/2012/02/use-of-indexof-and-charat-in-javascript.html*. Retrieved from http://raovishal.blogspot.com.

bennadel.(n.d.). *https://www.bennadel.com/resources/demo/morse_code/*. Retrieved from https://www.bennadel.com.

ChaosPandion.(n.d.). *https://stackoverflow.com/questions/8976627/how-to-add-two-strings-as-if-they-were-numbers*. Retrieved from https://stackoverflow.com.

css-tricks.com.(n.d.). *https://css-tricks.com/snippets/css/complete-guide-grid/*. Retrieved from https://css-tricks.com/.

kaspergotcha.(n.d.). *https://developer.mozilla.org/en-US/docs/Web/HTML/Element/select*. Retrieved from https://developer.mozilla.org.

Sheppy.(n.d.). *https://developer.mozilla.org/en-US/docs/Web/API/AudioContext*. Retrieved from https://developer.mozilla.org.

w3Schools.(n.d.). *https://www.w3schools.com/jsref/met_win_setinterval.asp*. Retrieved from https://www.w3schools.com.

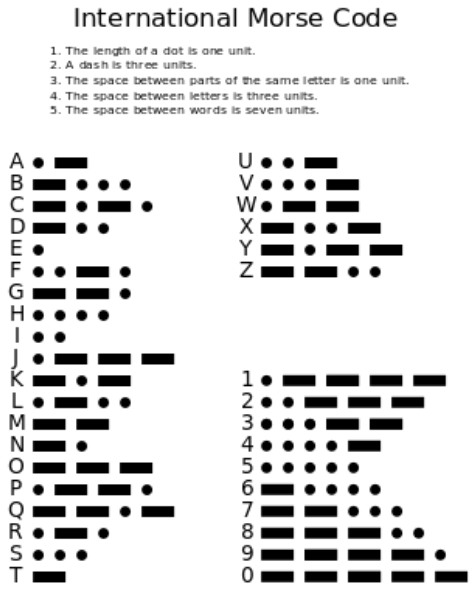
Wikipadia. (n.d.). *https://en.wikipedia.org/wiki/Morse_code*. Retrieved from https://en.wikipedia.org.

wikipedia. (n.d.). *https://en.wikipedia.org/wiki/Caesar_cipher*. Retrieved from https://en.wikipedia.org.

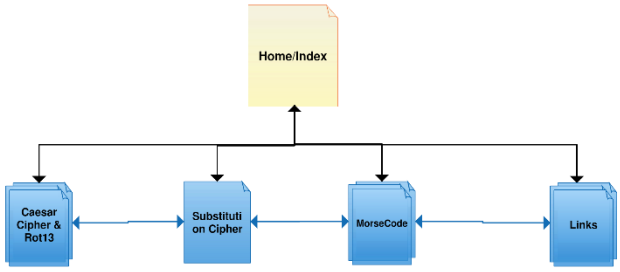
Wikipedia.(n.d.). *https://en.wikipedia.org/wiki/Substitution_cipher*. Retrieved from https://en.wikipedia.org.

Appendix

Word was being difficult when adding this so im not sure if it will format properly at all, I have tried to keep the appendix reference as obvious as possible but due to page formatting and images moving themselves around this was not always possible

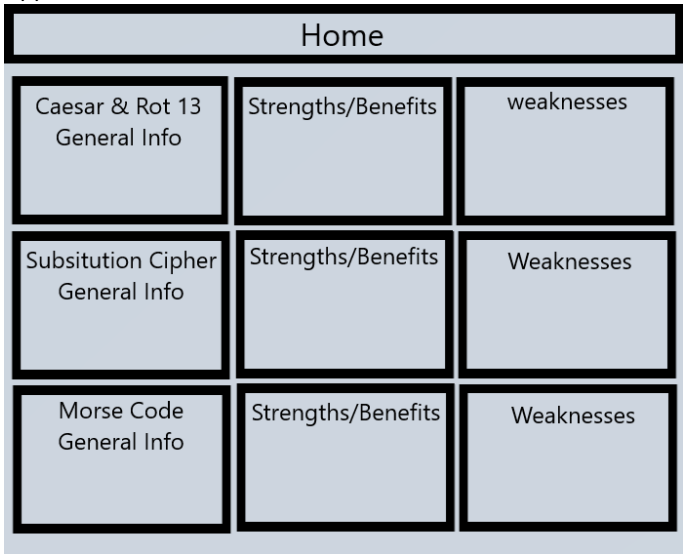


Appendix A - (Wikipadia, n.d.)

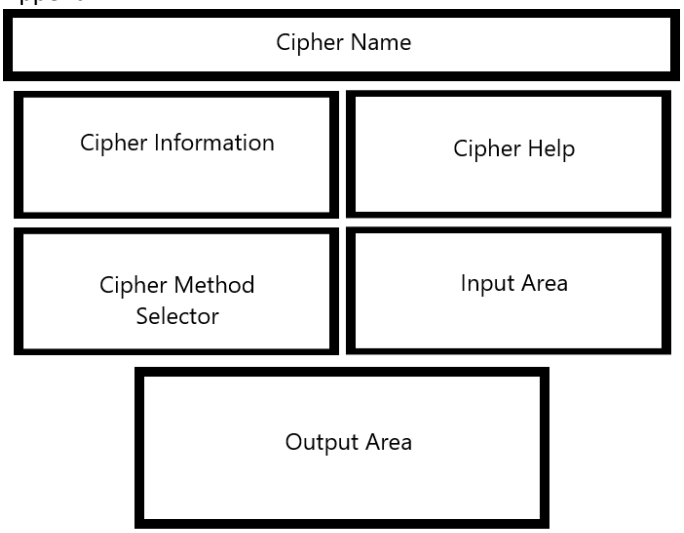


Appendix B – (created on Creately.com)

Appendix C –

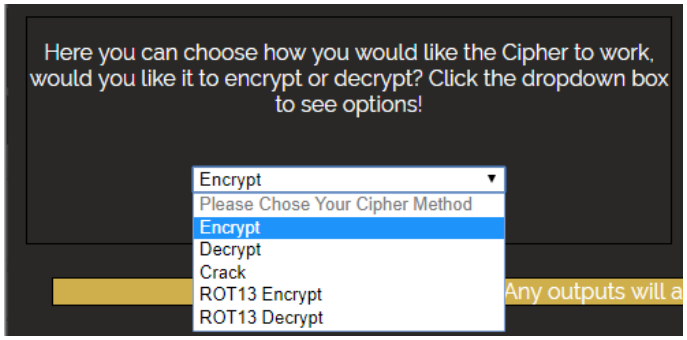


Appendix D-

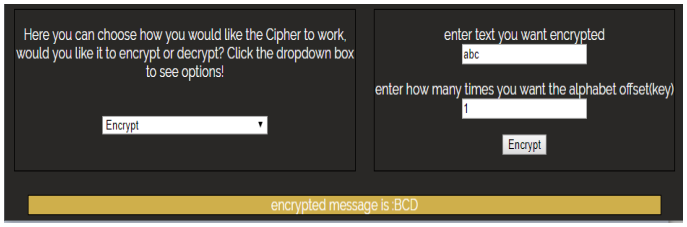


Appendix – E(rotated due to formatting with the page being funky)

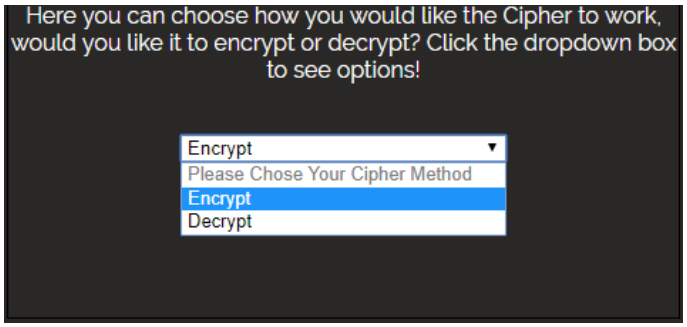
appendix – F



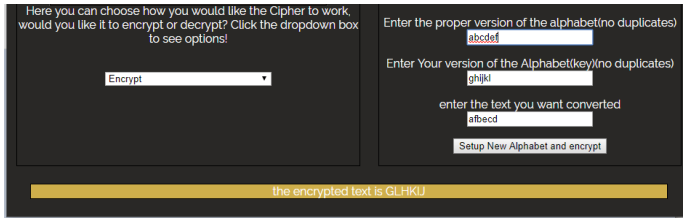
Appendix G



Appendix H



Appendix i



Appendix j

Appendix k