# Morse code application report:

Course: introduction to programming

Name: Mohamed Moheb

Id:2201214

introduction:

Confidentiality, integrity, authentication are three pillars of information security world and commonly known by the acronyms CIA, used for protection of any communication between two or multiple entities. To achieve this protection, the information must only be accessible for authorized individuals or corporations which is the purpose of confidentiality, it can be achieved through security measures such as the encryption. The art of hiding a message, which is referring to the encryption uses various tools to transform or hide an original message until it is successfully sent to the designated individual who will be able to decrypt it. One of these tools is the morse code application which is a historic communication tool that allows a user to encrypt or decrypt messages using morse code characters based on dotes (.) and dashes (-). It is considered as user friendly because of the easy steps to follow for achieving the result, the user simply enters the message in the application and choose to encrypt, following this step a morse code version of the message will be generated as an output by the application. Morse code isn’t only for encrypting, if the user has a morse code and chooses to decrypt it, the application can generate the initial or the decrypted message as an output. This report discusses distinct aspects of a morse code application, coded using a programming language called Python, such as some instructions for using the application, unit tests to ensure the functionality of the application, describing the different algorithms and choices made and finally discussing the time complexity of the algorithms.

Instructions:

In some cases, messages cannot be sent as a plain text and must be encrypted to avoid the risk of exposing valuable information from malicious individuals that can be positioned between the sender and the receiver. In the case of the morse code application, this encryption process is done by executing a source code given, in a python compiler because python is the coding language used for writing the code. Following the execution, the user gets asked by a message displayed (“please enter encrypt or decrypt”) to choose one of the two following methods: encrypt or decrypt. Once a method has been chosen and executed, the user will be asked to enter an input by a message displayed (“please enter your input”), this input will be depending on the method previously chosen. If the method chosen was encrypt, the user will enter an input consisting of an English phrase that will be encrypted after the execution to morse code. If the method chosen was decrypt, the user will enter an input consisting of a morse code that will be decrypted after execution to an English phrase. An error can occur in case the user enters an invalid method, as a solution a message will be displayed (“u have entered a wrong method, try again with encrypt or decrypt”) guiding the user to choose one of the functioning methods.

Table of unit tests:

Here is a table of unit test that can be found in the unit test file stored on GitHub:

|  |  |  |
| --- | --- | --- |
| Test | Method | Expected output |
| “hello world” | encrypt | “.... . .-.. .-.. --- / .-- --- .-. .-.. -..” |
| “this is a test” | encrypt | “- .... .. ... / .. ... / .- / - . ...” |
| “help” | encrypt | “.... . .-.. .--.” |
| “S O S” | encrypt | “... / --- / ...” |
| “UPPER lower” | encrypt | “..- .--. .--. . .-. / .-.. --- .-- . .-.” |
| “Give me the water” | encrypt | “--. .. ...- . / -- . / - .... . / .-- .- - . .-.” |
| “.... . .-.. .-.. --- / .-- --- .-. .-.. -..” | decrypt | “hello world” |
| “- .... .. ... / .. ... / .- / - . ... –“ | decrypt | “this is a test” |
| “.... . .-.. .--.” | decrypt | “help” |
| “... / --- / ...” | decrypt | “S O S” |
| “..- .--. .--. . .-. / .-.. --- .-- . .-.” | decrypt | “UPPER lower” |
| “--. .. ...- . / -- . / - .... . / .-- .- - . .-.” | decrypt | “Give me the water” |
| “12345” | encrypt | “” |
| “1 2 3 4 5” | encrypt | “” |
| “2 dogs and 1 cat” | encrypt | “” |

Repository:

Storing codes and files of an application can be done in an easier way, thanks to repositories that helps managing project assets and can be used as a tracking tool to see the changes done in the project overtime. Sharing the morse code application to the world can be done by uploading the file in a public repository on GitHub for example, so any user can encrypt or decrypt a desired message. Here is a link for the morse code repository:

Algorithm:

To achieve a functioning application such as the morse code application, having a code that runs properly is essential because it transitions between the abstract world of logical thinking to the complex world of computers. However, the abstract world of thinking is what leads to achieving a good code, this concept can be summed up in one word: algorithm. Algorithm is a procedure that works step by step to solve a problem and can be written in any language or even in pseudocode, it is known as the blueprint to solve a problem, meanwhile the code is the implementation of this blueprint. To be able to implement the code of the morse code application, there is a first encryption algorithm used to convert a message to a morse code by checking every letter in the message and for every letter get its morse code from pre-defined dictionary and add the morse code to an empty variable. Here’s a step-by-step breakdown of the encryption algorithm:

1)variable containing empty string where morse code will be stored (constant time)

2)-iterating every letter in the message (constant time)

-for every letter get its morse from the dictionary

-append it to the variable

3)return the variable

In addition, a second decryption algorithm is used to convert a morse code to a message by iterating every character and adding it to a current morse variable, once a space is encountered check for the letter corresponding to the characters in the current morse variable and add the letter to another decrypted variable and starts over until there are no characters left. Here’s a step-by-step breakdown of the decryption algorithm:

1)variable called current morse containing empty string where characters will be stored (constant time)

2)variable called decrypted containing empty string where letters will be stored (constant time)

3)iterating every character in the code (constant time)

-if the current character is not a space append it to the current morse variable

-if the current character is a space look up the corresponding letter of the characters stored in current morse and append it to decrypt and set current morse as an empty variable again

4)return decrypt variable

A clear breakdown of the algorithms has been made, but what is (constant time) found in almost every line of the algorithms­? Constant time refers to the time taken by the algorithm to iterate over every character and every letter, to make it clear each iteration takes constant time. Which leads to the time complexity being O(n) and n being the morse code or the message string length, a complexity of O(n) will allow the code to run efficiently.

Conclusion:

In conclusion, protecting the confidentiality of a sent message is important especially if the message contains sensitive information, which can be done by encrypting the message using a tool like the morse code application. This application is user friendly as the user only inputs whether to encrypt to decrypt, if the chosen input is to encrypt the user must enter an English phrase that will be converted after the execution of the code to a morse code. In the other hand, if the chosen input is to decrypt the user must enter a morse code that will be converted after the execution of the code to an English phrase. The user will be guided by a series of instructions that will be displayed on screen. A series of unit test are done to make sure the code is completing its function successfully even in the presence of exceptions, which has a separate file from the code, stored in the repository called GitHub where future changes can be done. A detailed explanation of the algorithms that were used as blueprints for the implementation of the code has been done which led to the discovery of the algorithm’s complexity that turned out to be (O)n. Explaining that every iteration is done in a constant time which is the case for the encryption and the decryption algorithm, this type of complexity is considered as efficient when running the code. A reflection can be made about the efficiency of the morse code application from a point of view of securing a message and whether there are better alternatives with user friendly usage.