Submission: Oct – Nov, 2020

INT213: Python Programming

Project Report

On

"Morse Code Translator"

Towards fulfillment for the undergraduate degree in

"Bachelor of Technology (B.Tech)"

As per

Lovely Professional University, Punjab.

Submitted by:

Anubhav Chawla	Saneet Kaul	Sachin Kumar Paswan
11914161	11913912	12000724
Roll No. 37	Roll No. 05	Roll No. 20

This page is intentionally left blank.

Acknowledgement

The success and final outcome of this assignment required a lot of guidance and assistance from my people and sources and we are extremely fortunate to have got this all along the completion of our project work. Whatever we have done is only due to such guidance and assistance and we would not forget to thank them. We respect and thank our professor, **Mrs. Neha Bagga** for giving an opportunity to do this assignment work and providing us all the support and guidance, which made us complete the project on time.

The assignment could not be completed without the efforts and cooperation from our group members. We would like to express our gratitude to our friends and respondents for support and willingness to help us.

Anubhav Chawla
11914161
Roll No. 37

Saneet Kaul
11913912
Roll No. 05

Sachin Kumar Paswan
12000724
Roll No. 20

Contents

The project is uploaded at:

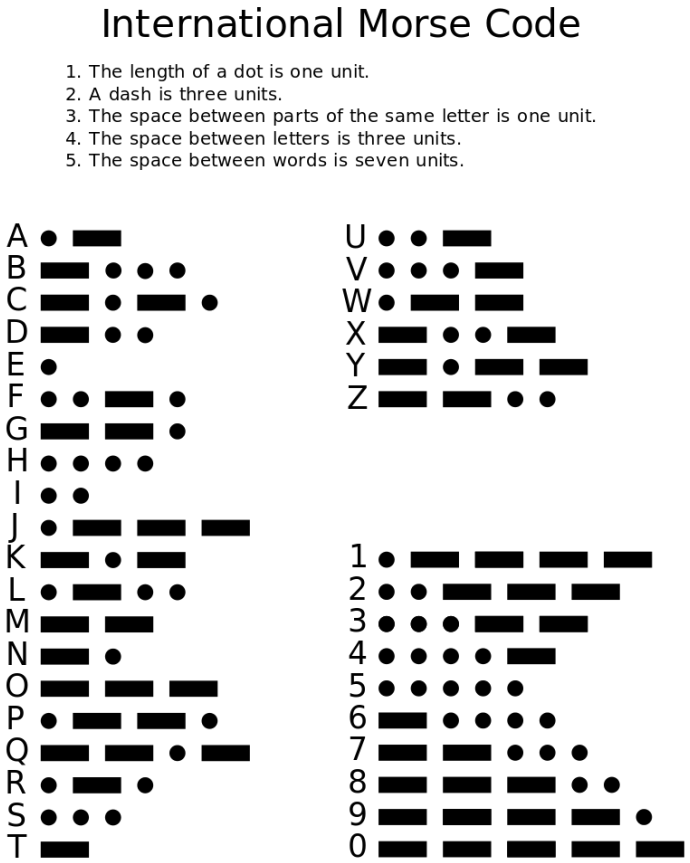
https://github.com/AnubhavChawla/INT213-Project

Introduction

Morse code is a method used in telecommunication to encode text characters as standardized sequences of two different signal durations, called ***dots*** and ***dashes.*** Morse code is named after **Samuel Morse**, an inventor of the telegraph.

The International Morse Code encodes the 26 English letters A through Z, some non-English letters, the Arabic numerals and a small set of punctuation and procedural signals (prosigns). There is no distinction between upper and lower case letters.

Morse code is usually transmitted by on-off keying of an information-carrying medium such as electric current, radio waves, visible light, or sound waves. The current or wave is present during the time period of the dot or dash and absent during the time between dots and dashes.



Morse code can be memorized, and Morse code signaling in a form perceptible to the human senses, such as sound waves or visible light, can be directly interpreted by persons trained in the skill.

In an emergency, Morse code can be generated by improvised methods such as turning a light on and off, tapping on an object or sounding a horn or whistle, making it one of the simplest and most versatile methods of telecommunication.

The most common distress signal is **SOS** – three dots, three dashes, and three dots – internationally recognized by treaty.

Morse code Translator:

Morse Code Translator lets anyone translate text to Morse code and decode Morse code to text easily. With the Morse code translator, anyone can convert any plain text containing English alphabets and numerals to Morse code and vice versa.

Project Objectives

- Develop an understanding about the Morse code.
- Use Python programming language and tkinter tools and widgets to create a fully-functional user interface.
- Write a program to encrypt plain text to Morse code, and to decrypt Morse code to English alphabets and numerals.
- Implement the program on a user interface.

System Requirements

- A desktop computer/laptop with MacOs, Windows or Linux operating system
- Python interpreter
- Tkinter package

GUI Development

Python provides various options for developing graphical user interfaces (GUIs).

- **Tkinter:** Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. Creating a GUI application using Tkinter is an easy task.

Tkinter Widgets:

Widget	Description	Attributes
Frame	The frame widget is used as a container widget to organize other widgets.	background, title, geometry
Toplevel	The Toplevel widget is used to provide a separate window container.	Background, title, geometry
Label	The label widget is used to provide a single-line caption for other widgets.	Text, background, foreground, font, relief, bd, height, width
Entry	The Entry widget is used to display a single-line text field for accepting values from a user.	textvariable, font, background, foreground, show, width
Text	The Text widget is used to display text in multiple lines.	font, foreground, background, wrap, width, height, bd
Button	The Button widget is used to display buttons in your application.	Text, background, font, foreground, width, height, command
tkMessageBox	This module is used to display message boxes in your application.	-

Morse Code Translator – Application

The application contains two different windows – Login Page and Morse Code Translator.

- Login Page – Login Page allows a user to gain access to the application by entering their user ID and password. Only authorized users can use the Morse Code Translator application.
- Morse Code Translator window – It is a translator that lets the user translate plain text (including English alphabets and numerals) to Morse code and vice versa. It contains separate multi-line text fields for the entry and retrieval of data.

Login Page

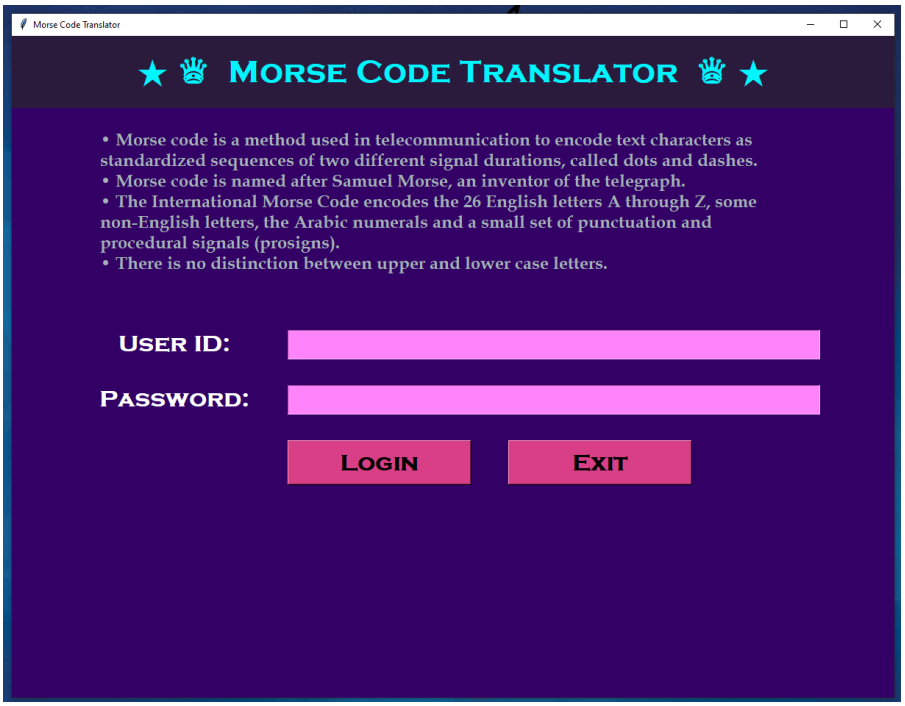
The login page contains the following widgets:

- Label –
    1. Header (Morse Code Translator)
    2. User ID
    3. Password
- Text –
    1. textarea1 (below the header label)
- Entry –
    1. User ID textbox
    2. User password textbox (modified to show characters entered as “stars”.)
- Button –
    1. Login Button (opens the Morse code translator window only if the user is authorized.)
    2. Exit Button (exits the application)

The Morse code Translator window can be only accessed by authorized users. The list of user IDs and their respective passwords is given below.

User ID	Password
anubhav11914161	7528045625
saneet11913912	9501230787
sachin12000724	6280823640

If the user ID and password do not match, a popup message box appears on the screen that asks user to either retry or exit the application.



The above screenshot shows the Login Page.

The AskRetryCancel Message box appears if the user enters invalid credentials.

User interface creation of Login Page

```
login_page = Tk()

login_page.configure(bg="#330066")

login_page.title("Morse Code Translator")

login_page.geometry("1200x900+350+50")



header = Label(login_page,text="\u2605 \u265b  Morse Code Translator \u265b \u2605",background="#2A1B3D",font=('Copperplate Gothic Bold',32,'bold'),foreground='turquoise1',width=46,height=2,relief=RAISED,bd=0)

header.pack()



textarea1 = Text(login_page,font=('Book Antiqua',18,'bold'),bg="#330066",bd=0,foreground='#A4B3B6',height=10)

textarea1.insert(INSERT,"\n\u2022 Morse code is a method used in telecommunication to encode text characters as \tstandardized sequences of two different signal durations, called dots and dashes.\n\u2022 Morse code is named after Samuel Morse, an inventor of the telegraph.")

textarea1.insert(INSERT,"\n\u2022 The International Morse Code encodes the 26 English letters A through Z, some \tnon-English letters, the Arabic numerals and a small set of punctuation and \tprocedural signals (prosigns).\n\u2022 There is no distinction between upper and lower case letters.")

textarea1.configure(state="disabled",wrap=WORD)

textarea1.pack()

user_id_val = StringVar()

user_pwd_val = StringVar()
```

```
user_id = Label(login_page,text="User
ID:",background="#330066",font=('Copperplate Gothic
Bold',24,'bold'),foreground="white",width=10,relief=SUNKEN,bd=0).pla
ce(x=100,y=400)

user_id_textbox =
Entry(login_page,textvariable=user_id_val,width=45,font=('Times New
Roman',24,'bold'),foreground='gray11',bg="orchid1").place(x=375,y=40
0)



user_pwd =
Label(login_page,text="Password:",background="#330066",font=('Copper
plate Gothic
Bold',24,'bold'),foreground='white',width=10,relief=RAISED,bd=0).pla
ce(x=100,y=475)

user_pwd_textbox =
Entry(login_page,textvariable=user_pwd_val,show="\u2605",width=45,fo
nt=('Times New
Roman',24,'bold'),foreground='gray11',bg="orchid1").place(x=375,y=47
5)



login_button =
Button(login_page,text="Login",background="#d83f87",font=('Copperpla
te Gothic
Bold',24,'bold'),foreground='black',width=10,command=login_procedure
).place(x=375,y=550)

exit_button =
Button(login_page,text="Exit",background="#d83f87",font=('Copperplat
e Gothic
Bold',24,'bold'),foreground='black',width=10,command=login_page.dest
roy).place(x=675,y=550)
```

- The Login Page procedure works on the function called login_procedure. This functionality is the command executed by Login Button (login_button). This function uses an if-else statement to validate the user IDs and their respective passwords.

The code for **login_procedure** is mentioned below:

```
def login_procedure():

    if((user_id_val.get()=="anubhav11914161" and
user_pwd_val.get()=="7528045625") or
(user_id_val.get()=="saneet11913912" and
user_pwd_val.get()=="9501230787") or
(user_id_val.get()=="sachin12000724" and
user_pwd_val.get()=="6280823640")):

        Open_Morse_Code_Translator()

    else:

        messagebox.askretrycancel("Unauthorised Access", "Invalid
credentials! You may retry or exit the application.")
```

- The exit_button uses the in-built method to destroy the Login Page window i.e. **login_page.destroy**

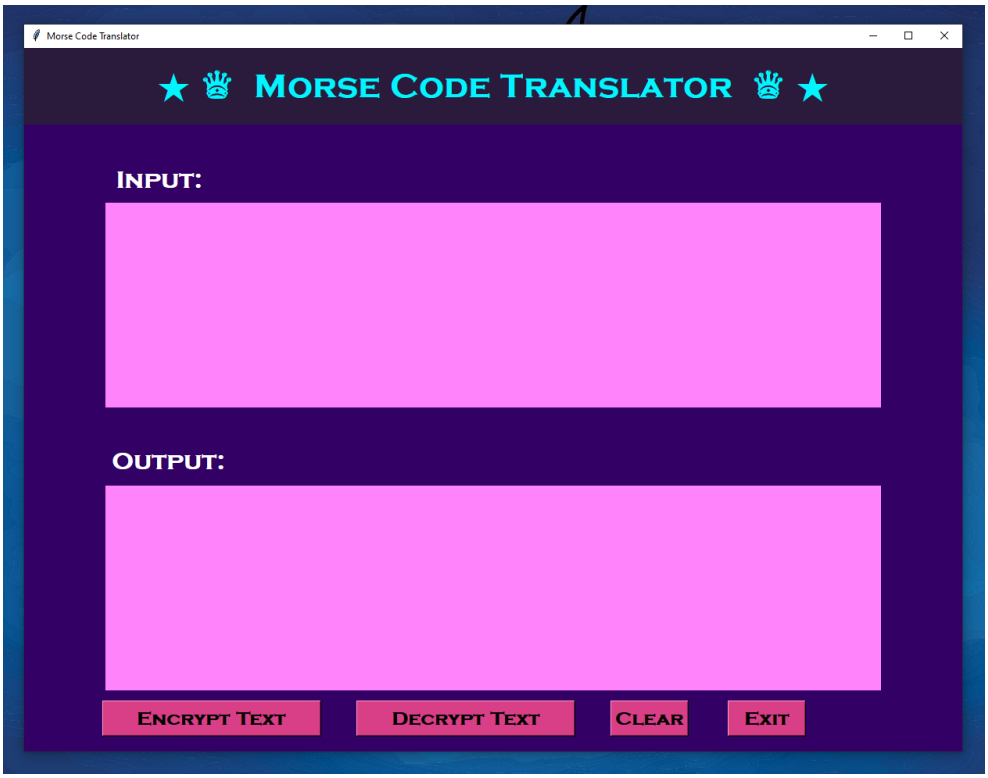
If the user is authorized, a new Top level window opens on click of the button. That window is the Morse code Translator window. The new window opens and Login Page gets hidden using below mentioned function.

```
def Open_Morse_Code_Translator():

    translator.deiconify()

    login_page.withdraw()
```

Morse Code Translator Window

The Morse Code Translator is a Top level window used as a separate container for the main translator functionality. It contains the following widgets:

- Label –
    1. Header (Morse Code Translator)
    2. Input Label
    3. Output Label
- Text –
    1. Input text area (user entered text; can be either plain text or Morse code)
    2. Output text area (text after encryption or decryption)
- Button –
    1. Encrypt Text Button (converts plain text containing English alphabets and numerals to Morse code)
    2. Decrypt Text Button (converts Morse code to plain text)
    3. Clear Button (clears both Input and output text areas)
    4. Exit Button (exits the application)



The above screenshot shows the Morse Code Translator window.

User interface of Morse code Translator window

```
translator = Toplevel()

translator.withdraw()

translator.title("Morse Code Translator")

translator.configure(bg="#330066")

translator.geometry("1200x900+350+50")


header = Label(translator,text="\u2605 \u265b  Morse Code Translator \u265b \u2605",background="#2A1B3D",font=('Copperplate Gothic Bold',32,'bold'),foreground='turquoise1',width=46,height=2,relief=RAISED,bd=0)

header.pack()


input_label = Label(translator,text="Input:",background="#330066",foreground="white",width=6,font=("Copperplate Gothic Bold",24,'bold'),relief=RAISED,bd=0).place(x=100,y=150)

morse_code_label = Label(translator,text="Output:",background="#330066",foreground="white",width=7,font=("Copperplate Gothic Bold",24,'bold'),relief=RAISED,bd=0).place(x=100,y=510)


input_text = Text(translator,font=('Book Antiqua',16),bd=0,foreground='gray2',bg='orchid1',height=10,width=90,wrap=WORD)

input_text.pack(padx=100,pady=100)
```

```
output_text = Text(translator,font=('Book
Antiqua',16),bd=0,foreground='gray2',bg='orchid1',height=10,width=90
,wrap=WORD)

output_text.pack(padx=100)



encrypt_btn = Button(translator,text="Encrypt
Text",background="#d83f87",font=('Copperplate Gothic
Bold',18,'bold'),foreground='black',width=15,height=1,command=Encryp
t_Button_Function).place(x=100,y=835)

decrypt_btn = Button(translator,text="Decrypt
Text",background="#d83f87",font=('Copperplate Gothic
Bold',18,'bold'),foreground='black',width=15,command=Decrypt_Button_
Function).place(x=425,y=835)

clear_btn =
Button(translator,text="Clear",background="#d83f87",font=('Copperpla
te Gothic
Bold',18,'bold'),foreground='black',width=5,command=Clear_Button_Fun
ction).place(x=750,y=835)

exit_btn =
Button(translator,text="Exit",background="#d83f87",font=('Copperplat
e Gothic
Bold',18,'bold'),foreground='black',width=5,command=translator.destr
oy).place(x=900,y=835)



login_page.mainloop()
```

Encrypt Button Functionality:

```
def Encrypt_Button_Function():
    Loophole_Check()
    input_text_val = input_text.get("1.0","end-1c")
    input_text_val = input_text_val.upper()
    encrypted_text=""
    for letter in input_text_val:
        if(letter != " "):
            encrypted_text=encrypted_text+Morse_Code_DICT[letter]+"
"
        else:
            encrypted_text += " "
    output_text.delete("1.0",END)
    output_text.insert(INSERT,encrypted_text)
    output_text.config(state="normal")
```

- The Encrypt_Button_Function gets the input text from the input text area and converts it to uppercase as recognized by Morse code dictionary.
- Every plain text character is converted to Morse Code by referencing from dictionary. The conversion is done using for loop. Encrypted text contains a space between each Morse code alphabet.

Morse Code Dictionary:

Morse_Code_DICT={

'A' : ".-", 'B' : "-...", 'C' : "-.-.", 'D' : "-..", 'E' : ".", 'F' : "..-.",

'G' : "--.", 'H' : "....", 'I' : "..", 'J' : ".---", 'K' : "-.-", 'L' : ".-..",

'M' : "--", 'N' : "-.", 'O' : "---", 'P' : ".--.", 'Q' : "--.-", 'R' : ".-.",

'S' : "...", 'T' : "-", 'U' : "..-", 'V' : "...-", 'W' : ".--", 'X' : "-..-",

'Y' : "-.--", 'Z' : "--..",

'0' : "-----", '1' : ".----", '2' : "..---", '3' : "...--", '4' : "....-",

'5' : ".....", '6' : "-....", '7' : "--...", '8' : "---..", '9' : "----."

}

- This dictionary is used as a reference for the conversion of alphabets and numerals to their respective Morse code equivalents.
- It is used in Encrypt button and Decrypt button functionality.

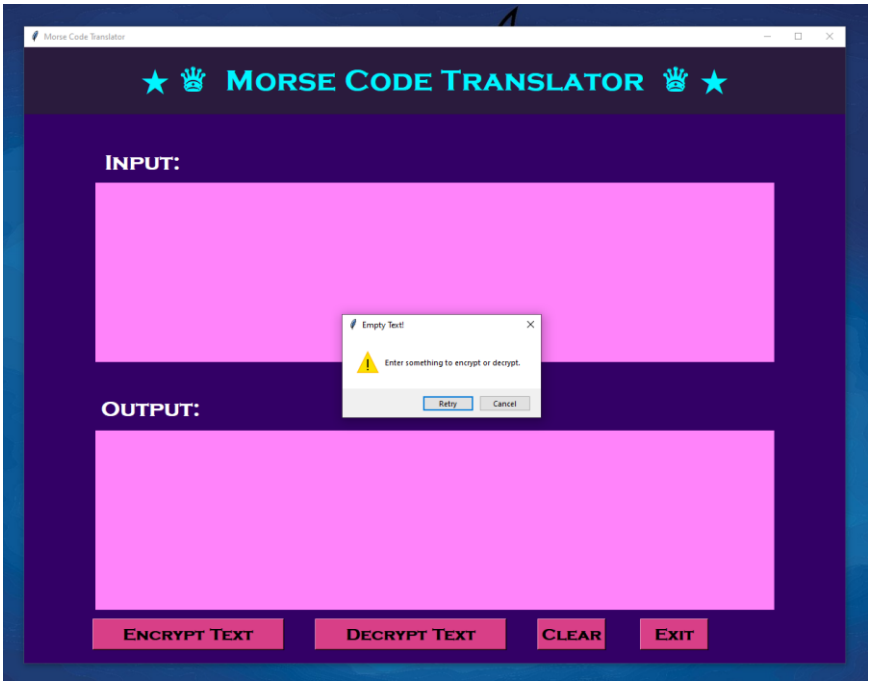
Decrypt Button Functionality:

```
def Decrypt_Button_Function():
    Loophole_Check()
    input_text_val = input_text.get("1.0","end-1c")
    input_text_val +=" "
    key_list = list(Morse_Code_DICT.keys())
    val_list = list(Morse_Code_DICT.values())
    morsecode=""
    decrypted_text=""
    space_found=0
    for letter in input_text_val:
        if(letter != " "):
            morsecode += letter
            space_found = 0
        else:
            space_found += 1
            if (space_found >= 2):
                decrypted_text += " "
            else:
                decrypted_text = decrypted_text +
key_list[val_list.index(morsecode)]
                morsecode=""
```

```
 output_text.delete("1.0",END)

 output_text.insert(INSERT,decrypted_text)

 output_text.config(state="normal")
```

- The Decrypt Button uses the dictionary values as reference to convert Morse code to plain text.
- It recognizes a Morse code combination and the spaces between and references them accordingly. The converted plain is without spaces between each alphabet and is uppercase.
- The decrypt and encrypt buttons use a Loophole_Check function to check for empty string and give an error message box.

```
def Loophole_Check():

    input_text_val = input_text.get("1.0","end-1c")

    if input_text_val=="":

        messagebox.askretrycancel("Empty Text!","Enter something to
encrypt or decrypt.")

    else:

        pass
```



The AskRetryCancel message box if there is no input to decrypt or encrypt.

- The Clear button uses a function to delete all the text present in Input and Output text areas. The functionality is mentioned below.

```
def Clear_Button_Function():

    input_text.delete("1.0",END)

    output_text.delete("1.0",END)
```

Results:

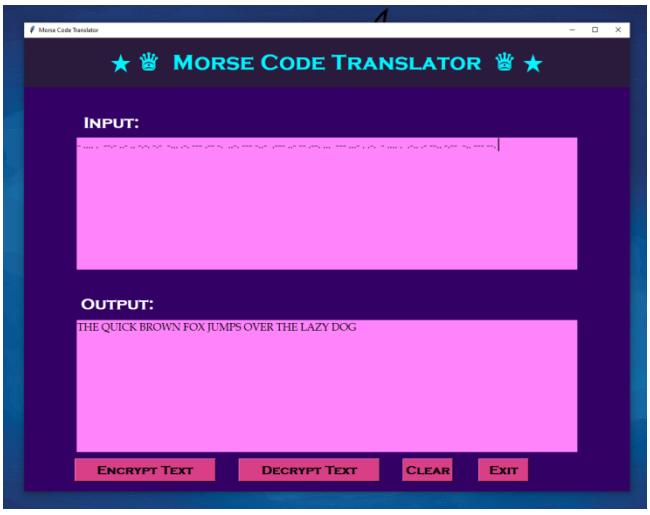
The below given screenshots are a result of fully-functional Morse code translator.

Example used below is, “The quick brown fox jumps over the lazy dog”

The translator converts this text to Morse code on clicking of Encrypt button.

The result is: “- .... . --.- ..- .. -.-. -.- -... .-. --- .-- -. ..-. --- -..- .--- ..- -- .--. ... --- ...- . .-. - .... . .-.. .- --.. -.-- -.. --- --.”

Similarly, the decrypt button converts the Morse code to plain text on the click of Decrypt button. For example, “- .... . --.- ..- .. -.-. -.- -... .-. --- .-- -. ..-. --- -..- .--- ..- -- .--. ... --- ...- . .-. - .... . .-.. .- --.. -.-- -.. --- --.” gets converted to “THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG”



The project is uploaded at:

https://github.com/AnubhavChawla/INT213-Project

Member Contributions

1. **Anubhav Chawla (11914161):**

- Development of GUI of Login Page and Morse Code Translator
- Login Page Functionality
- Morse Code Translator functionality
- Project Report editing

2. **Saneet Kaul (11913912):**

- Project Report writing
- Leadership (direction, organization and management of group)
- Encrypt/ Decrypt functionality
- Suggestions for GUI creation

3. **Sachin Kumar Paswan (12000724):**

- Collection of theoretical information
- GUI Layout
- Data analysis

The assigned project could not be completed without the joint efforts, suggestions and support of the group members. The valuable suggestions on this proposed project gave us an inspiration to improve our assignment.

References

- ✓ PYTHON PROGRAMMING USING PROBLEM SOLVING APPROACH - REEMA THAREJA
- ✓ https://en.wikipedia.org/wiki/Morse_code
- ✓ https://www.britannica.com/topic/Morse-Code
- ✓ https://morsedecoder.com/