FINAL PROJECT REPORT
NOTEPAD

By:

Jeremy Ponto (2301891525)

Class:

L1AC

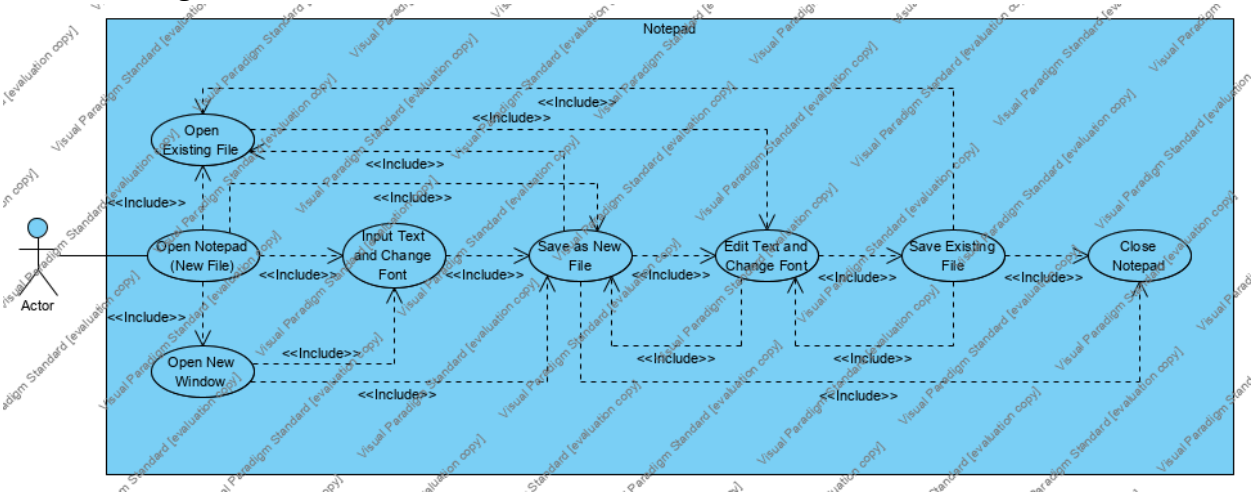
1. Project Specifications

Notepad is a text editor which has several purposes, such as making notes, to-do list, configuration files, etc. When using notepad, the user can open new file or existing file, save current file, or save them with another name. Notepad also features edit commands, such as undo, cut, copy, paste, delete, and select all. The user can also change the font configuration according to the user's preference.

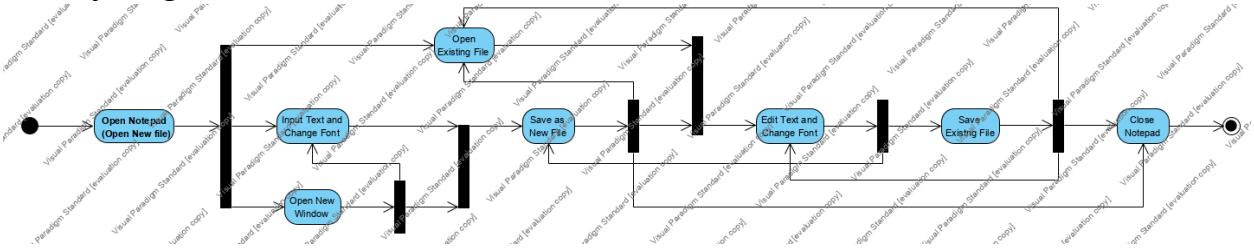
2. Solution Design

Class Diagram:

Use Case Diagram:



Activity Diagram:



3. A Discussion about What is Implemented and How It Works

Notepad is made using tkinter, a library which is used for making applications with GUI. To use the panels and widgets, the library must be imported.

```
# module for making GUI
from tkinter import *
```

When making an application, the “About” tab often appears and it usually displays the information of the author or the version of the application. In this case, the author’s information is displayed using a messagebox. Therefore, the library specific for the messagebox also will be imported.

```
# module to show messagebox
from tkinter.messagebox import *
```

The user will save files when the have finished working on it and there is a possibility that the user wants to open the corresponding file to see if there are something to change, so the library dealing with opening and saving files is as mentioned below.

```
# module to open and save file
from tkinter.filedialog import *
```

After opening or saving file, file name will be displayed as the title. To get the file name, the path of the file is required. Thus, the library is as follows.

```
# module for getting file name
import os
```

When working on a file, sometimes user wants to change the font according to their needs. Hence, the library is needed.

```
# module to change font
from tkinter.font import *
```

The libraries are ready. Now, to make the application, frame must be created so that panels can be placed to create a neat layout and all of those elements must be initialized so they can use all the functions, including making new file, opening and saving file, edit commands, reading author's information and applying changes to font. To do this, custom class "NotepadFrame" which inherit the class "Frame" will be created and initialized with the settings below. Do not forget to set the file path to "None" since a new file is opened by default.

```
# class for making frame
class NotepadFrame(Frame):
    def __init__(self, master = None):
        # the beginning of Notepad, so there is no file path stored
        self.currentfilepath = None

        # initialize Notepad frame
        Frame.__init__(self, master)
        self.notepadframe = self.master

        # call function to initialize Notepad frame settings
        self.setNotepadFrameSettings()
        # call function to initialize Notepad controls
        self.makeNotepadControls()

    def setNotepadFrameSettings(self):
        # set Notepad frame title
        self.notepadframe.title("Untitled - Notepad")
        # set Notepad frame size
        self.notepadframe.geometry("1200x675")
```

As seen above, a custom function "self.makeNotepadControls()" is called. This function is for making panels and widgets for the frame. First, menu must be placed so user can perform the functions mentioned before, excluding changing font. The code is as typed below.

```
    def makeNotepadControls(self):
        # make menubar
        self.menubar = Menu(self.notepadframe)
```

```
        # add File menu
        self.filemenu = Menu(self.menubar, tearoff = 0)
        self.menubar.add_cascade(label = "File", menu = self.filemenu)

        # add New menu item in File menu
        self.filemenu.add_command(label = "New", command = self.fileNew)
        # add New Window menu item in File menu
        self.filemenu.add_command(label = "New Window", command = self.fileN
ewWindow)
        # add Open menu item in File menu
        self.filemenu.add_command(label = "Open", command = self.fileOpen)
        # add Save menu item in File menu
        self.filemenu.add_command(label = "Save", command = self.fileSave)
        # add Save As menu item in File menu
        self.filemenu.add_command(label = "Save As", command = self.fileSave
As)
        # add Separator
        self.filemenu.add_separator()
        # add Exit menu item in File menu
        self.filemenu.add_command(label = "Exit", command = self.quit)

        # add Edit menu
        self.editmenu = Menu(self.menubar, tearoff = 0)
        self.menubar.add_cascade(label = "Edit", menu = self.editmenu)

        # add Undo menu item in Edit menu
        self.editmenu.add_command(label = "Undo", accelerator = "Ctrl+Z", co
mmand = self.editUndo)
        # add Redo menu item in Edit menu
        self.editmenu.add_command(label = "Redo", accelerator = "Ctrl+Y", co
mmand = self.editRedo)
        # add Separator
        self.editmenu.add_separator()
        # add Cut menu item in Edit menu
        self.editmenu.add_command(label = "Cut", accelerator = "Ctrl+X", com
mand = self.editCut)
        # add Copy menu item in Edit menu
        self.editmenu.add_command(label = "Copy", accelerator = "Ctrl+C", co
mmand = self.editCopy)
        # add Paste menu item in Edit menu
        self.editmenu.add_command(label = "Paste", accelerator = "Ctrl+V", c
ommand = self.editPaste)
        # add Delete menu item in Edit menu
        self.editmenu.add_command(label = "Delete", accelerator = "Del", com
mand = self.editDelete)
```

```
        # add Separator
        self.editmenu.add_separator()
        # add Select All menu item in Edit menu
        self.editmenu.add_command(label = "Select All", accelerator = "Ctrl+
A", command = self.editSelectAll)

        # add About menu
        self.aboutmenu = Menu(self.menubar, tearoff = 0)
        self.menubar.add_cascade(label = "About", menu = self.aboutmenu)

        # add About Notepad menu item in About menu
        self.aboutmenu.add_command(label = "About Author", command = self.ab
outAboutAuthor)

        # displaying menubar in frame
        self.notepadframe.config(menu = self.menubar)
```

Next, custom functions will be made so the menu will be fully functional.

```
    def fileNew(self):
        # make new file
        self.notepadframe.title("Untitled - Notepad")
        self.file = None
        self.text.delete(1.0, END)

    def fileNewWindow(self):
        # make new Notepad frame
        self.notepadwindow = Toplevel(self.notepadframe)
        self.notepadframe = NotepadFrame(self.notepadwindow)

    def fileOpen(self):
        # file types
        self.filetypes = [("Text Document", "*.txt"), ("All Files", "*.*")]

        # ask whether user wants to open file
        self.currentfilepath = askopenfilename(filetypes = self.filetypes, d
efaultextension = ".txt")

        if self.currentfilepath == "":
            # no file name, so user cannot open file
            self.currentfilepath = None

        # open file
        else:
            # delete previous contents in text
            self.text.delete(1.0, END)
```

```
            # open saved file to read new contents
            file = open(self.currentfilepath, "r")

            # insert new contents to text
            self.text.insert(1.0, file.read())

            # close read file
            file.close()

            # read current font settings from files
            file = open("C:/Users/user/.vscode/Final Project/notepadfont.txt
", "r")
            self.currentfont = file.read()
            file.close()
            file = open("C:/Users/user/.vscode/Final Project/notepadfontstyl
e.txt", "r")
            self.currentfontstyle = file.read()
            file.close()
            file = open("C:/Users/user/.vscode/Final Project/notepadfontsize
.txt", "r")
            try:
                self.currentfontsize = int(file.read())
            except:
                pass
            file.close()

            # load current font settings if available
            if self.currentfont in families() and self.currentfontstyle in [
"Regular", "Bold", "Italic", "Underline", "Overstrike"] and self.currentfont
size in [8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 26, 28, 36, 48, 72]:
                if self.currentfontstyle == "Regular":
                    self.fontsettings = Font(family = self.currentfont, size
 = self.currentfontsize, weight = "normal")
                elif self.currentfontstyle == "Bold":
                    self.fontsettings = Font(family = self.currentfont, size
 = self.currentfontsize, weight = "bold")
                elif self.currentfontstyle == "Italic":
                    self.fontsettings = Font(family = self.currentfont, size
 = self.currentfontsize, weight = "normal", slant = "italic")
                elif self.currentfontstyle == "Underline":
                    self.fontsettings = Font(family = self.currentfont, size
 = self.currentfontsize, weight = "normal", underline = 1)
                elif self.currentfontstyle == "Overstrike":
```

```
                    self.fontsettings = Font(family = self.currentfont, size
 = self.currentfontsize, weight = "normal", overstrike = 1)
                self.text.config(font = self.fontsettings)

                # display font settings
                self.fontvariable.set(self.currentfont)
                self.fontstylevariable.set(self.currentfontstyle)
                self.fontsizevariable.set(self.currentfontsize)

            # change Notepad window title
            self.notepadframe.title(os.path.basename(self.currentfilepath) +
 " - Notepad")

    def fileSave(self):
        # file types
        self.filetypes = [("Text Document", "*.txt"), ("All Files", "*.*")]

        # ask whether user wants to save file
        if self.currentfilepath == None:
            # the beginning of Notepad, so Save acts the same as Save As
            self.currentfilepath = asksaveasfilename(initialfile = 'Untitled
.txt', filetypes = self.filetypes, defaultextension = ".txt")

            if self.currentfilepath == "":
                # no file name, so user cannot save file
                self.currentfilepath = None

            else:
                # save file
                file = open(self.currentfilepath, "w")
                file.write(self.text.get(1.0, END))
                file.close()

                # write font settings into new files if available
                if self.currentfont in families() and self.currentfontstyle
in ["Regular", "Bold", "Italic", "Underline", "Overstrike"] and self.current
fontsize in [8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 26, 28, 36, 48, 7
2]:
                    file = open("C:/Users/user/.vscode/Final Project/notepad
font.txt", "w")
                    file.write(self.currentfont)
                    file.close()
                    file = open("C:/Users/user/.vscode/Final Project/notepad
fontstyle.txt", "w")
                    file.write(self.currentfontstyle)
```

```
                    file.close()
                    file = open("C:/Users/user/.vscode/Final Project/notepad
fontsize.txt", "w")
                    file.write(str(self.currentfontsize))
                    file.close()

                # change Notepad window title
                self.notepadframe.title(os.path.basename(self.currentfilepat
h) + " - Notepad")

        else:
            # save file changes
            file = open(self.currentfilepath, "w")
            file.write(self.text.get(1.0, END))
            file.close()

            # write font settings into new files if available
            if self.currentfont in families() and self.currentfontstyle in [
"Regular", "Bold", "Italic", "Underline", "Overstrike"] and self.currentfont
size in [8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 26, 28, 36, 48, 72]:
                file = open("C:/Users/user/.vscode/Final Project/notepadfont
.txt", "w")
                file.write(self.currentfont)
                file.close()
                file = open("C:/Users/user/.vscode/Final Project/notepadfont
style.txt", "w")
                file.write(self.currentfontstyle)
                file.close()
                file = open("C:/Users/user/.vscode/Final Project/notepadfont
size.txt", "w")
                file.write(str(self.currentfontsize))
                file.close()

    def fileSaveAs(self):
        # file types
        self.filetypes = [("Text Document", "*.txt"), ("All Files", "*.*")]

        # ask whether user wants to save file
        if self.currentfilepath == None:
            # the beginning of Notepad, so file name is Untitled
            self.currentfilepath = asksaveasfilename(initialfile = 'Untitled
.txt', filetypes = self.filetypes, defaultextension = ".txt")

        else:
```

```
            # the saved file has already been named, so user must change the
 file name
            self.currentfilepath = asksaveasfilename(initialfile = os.path.b
asename(self.currentfilepath), filetypes = self.filetypes, defaultextension
= ".txt")

        if self.currentfilepath == "":
            # no file name, so user cannot save file
            self.currentfilepath = None

        else:
            # save file
            file = open(self.currentfilepath, "w")
            file.write(self.text.get(1.0, END))
            file.close()

            # write font settings into new files if available
            if self.currentfont in families() and self.currentfontstyle in [
"Regular", "Bold", "Italic", "Underline", "Overstrike"] and self.currentfont
size in [8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 26, 28, 36, 48, 72]:
                file = open("C:/Users/user/.vscode/Final Project/notepadfont
.txt", "w")
                file.write(self.currentfont)
                file.close()
                file = open("C:/Users/user/.vscode/Final Project/notepadfont
style.txt", "w")
                file.write(self.currentfontstyle)
                file.close()
                file = open("C:/Users/user/.vscode/Final Project/notepadfont
size.txt", "w")
                file.write(str(self.currentfontsize))
                file.close()

            # change Notepad window title
            self.notepadframe.title(os.path.basename(self.currentfilepath) +
 " - Notepad")

    def editUndo(self):
        # generate Undo command
        self.text.edit_undo()

    def editRedo(self):
        # generate Redo command
        self.text.edit_redo()
```

```
    def editCut(self):
        # generate Cut command
        self.text.event_generate("<<Cut>>")

    def editCopy(self):
        # generate Copy command
        self.text.event_generate("<<Copy>>")

    def editPaste(self):
        # generate Paste command
        self.text.event_generate("<<Paste>>")

    def editDelete(self):
        # generate Delete command
        self.text.event_generate("<<Clear>>")

    def editSelectAll(self):
        # generate Select All command
        self.text.event_generate("<<SelectAll>>")

    def aboutAboutAuthor(self):
        # author info
        messagebox.showinfo("About Author", "Jeremy Ponto\n2301891525")
```

Because menus are now available, panels for placing widgets can be placed.

```
# add main panel for frame
        self.panel1 = PanedWindow(self.notepadframe)
        self.panel1.pack(fill = BOTH, expand = 1)

        # add second panel inside main panel with vertical orientation for c
ombining font configuration and text
        self.panel2 = PanedWindow(self.panel1, orient = VERTICAL)
        self.panel1.add(self.panel2)

        # add third panel inside second panel with horizontal orientation fo
r placing font configuration option menus
        self.panel3 = PanedWindow(self.panel2, orient = HORIZONTAL)
        self.panel2.add(self.panel3)
```

Panels are ready. Now, option menus for font configuration can be created and placed in the panel with horizontal orientation so user can change font settings. Font settings must be “None” since none of the selection for changing font is done.

```
        # make Font option menu
        self.fontvariable = StringVar(self.panel3)
        self.fontvariable.set("Please select the Font of your choice")
```

```
        self.fontoptionmenu = OptionMenu(self.panel3, self.fontvariable, 'Sy
stem', '8514oem', 'Fixedsys', 'Terminal', 'Modern', 'Roman', 'Script', 'Cour
ier', 'MS Serif', 'MS Sans Serif', 'Small Fonts', 'Marlett', 'Arial', 'Arabi
c Transparent', 'Arial Baltic', 'Arial CE', 'Arial CYR', 'Arial Greek', 'Ari
al TUR', 'Arial Black', 'Bahnschrift Light', 'Bahnschrift SemiLight', 'Bahns
chrift', 'Bahnschrift SemiBold', 'Bahnschrift Light SemiCondensed', 'Bahnsch
rift SemiLight SemiConde', 'Bahnschrift SemiCondensed', 'Bahnschrift SemiBol
d SemiConden', 'Bahnschrift Light Condensed', 'Bahnschrift SemiLight Condens
ed', 'Bahnschrift Condensed', 'Bahnschrift SemiBold Condensed', 'Calibri', '
Calibri Light', 'Cambria', 'Cambria Math', 'Candara', 'Candara Light', 'Comi
c Sans MS', 'Consolas', 'Constantia', 'Corbel', 'Corbel Light', 'Courier New
', 'Courier New Baltic', 'Courier New CE', 'Courier New CYR', 'Courier New G
reek', 'Courier New TUR', 'Ebrima', 'Franklin Gothic Medium', 'Gabriola', 'G
adugi', 'Georgia', 'Impact', 'Ink Free', 'Javanese Text', 'Leelawadee UI', '
Leelawadee UI Semilight', 'Lucida Console', 'Lucida Sans Unicode', 'Malgun G
othic', '@Malgun Gothic', 'Malgun Gothic Semilight', '@Malgun Gothic Semilig
ht', 'Microsoft Himalaya', 'Microsoft JhengHei', '@Microsoft JhengHei', 'Mic
rosoft JhengHei UI', '@Microsoft JhengHei UI', 'Microsoft JhengHei Light', '
@Microsoft JhengHei Light', 'Microsoft JhengHei UI Light', '@Microsoft Jheng
Hei UI Light', 'Microsoft New Tai Lue', 'Microsoft PhagsPa', 'Microsoft Sans
 Serif', 'Microsoft Tai Le', 'Microsoft YaHei', '@Microsoft YaHei', 'Microso
ft YaHei UI', '@Microsoft YaHei UI', 'Microsoft YaHei Light', '@Microsoft Ya
Hei Light', 'Microsoft YaHei UI Light', '@Microsoft YaHei UI Light', 'Micros
oft Yi Baiti', 'MingLiU-ExtB', '@MingLiU-ExtB', 'PMingLiU-ExtB', '@PMingLiU-
ExtB', 'MingLiU_HKSCS-ExtB', '@MingLiU_HKSCS-
ExtB', 'Mongolian Baiti', 'MS Gothic', '@MS Gothic', 'MS UI Gothic', '@MS UI
 Gothic', 'MS PGothic', '@MS PGothic', 'MV Boli', 'Myanmar Text', 'Nirmala U
I', 'Nirmala UI Semilight', 'Palatino Linotype', 'Segoe MDL2 Assets', 'Segoe
 Print', 'Segoe Script', 'Segoe UI', 'Segoe UI Black', 'Segoe UI Emoji', 'Se
goe UI Historic', 'Segoe UI Light', 'Segoe UI Semibold', 'Segoe UI Semilight
', 'Segoe UI Symbol', 'SimSun', '@SimSun', 'NSimSun', '@NSimSun', 'SimSun-
ExtB', '@SimSun-
ExtB', 'Sitka Small', 'Sitka Text', 'Sitka Subheading', 'Sitka Heading', 'Si
tka Display', 'Sitka Banner', 'Sylfaen', 'Symbol', 'Tahoma', 'Times New Roma
n', 'Times New Roman Baltic', 'Times New Roman CE', 'Times New Roman CYR', '
Times New Roman Greek', 'Times New Roman TUR', 'Trebuchet MS', 'Verdana', 'W
ebdings', 'Wingdings', 'Yu Gothic', '@Yu Gothic', 'Yu Gothic UI', '@Yu Gothi
c UI', 'Yu Gothic UI Semibold', '@Yu Gothic UI Semibold', 'Yu Gothic Light',
 '@Yu Gothic Light', 'Yu Gothic UI Light', '@Yu Gothic UI Light', 'Yu Gothic
 Medium', '@Yu Gothic Medium', 'Yu Gothic UI Semilight', '@Yu Gothic UI Semi
light', 'HoloLens MDL2 Assets', 'Century', 'Wingdings 2', 'Wingdings 3', 'Te
mpus Sans ITC', 'Pristina', 'Papyrus', 'Mistral', 'Lucida Handwriting', 'Kri
sten ITC', 'Juice ITC', 'French Script MT', 'Freestyle Script', 'Bradley Han
d ITC', 'MS Outlook', 'Arial Narrow', 'Book Antiqua', 'Garamond', 'Monotype
Corsiva', 'Century Gothic', 'Algerian', 'Baskerville Old Face', 'Bauhaus 93'
```

```
, 'Bell MT', 'Berlin Sans FB', 'Bernard MT Condensed', 'Bodoni MT Poster Com
pressed', 'Britannic Bold', 'Broadway', 'Brush Script MT', 'Californian FB',
 'Centaur', 'Chiller', 'Colonna MT', 'Cooper Black', 'Footlight MT Light', '
Harlow Solid Italic', 'Harrington', 'High Tower Text', 'Jokerman', 'Kunstler
 Script', 'Lucida Bright', 'Lucida Calligraphy', 'Lucida Fax', 'Magneto', 'M
atura MT Script Capitals', 'Modern No. 20', 'Niagara Engraved', 'Niagara Sol
id', 'Old English Text MT', 'Onyx', 'Parchment', 'Playbill', 'Poor Richard',
 'Ravie', 'Informal Roman', 'Showcard Gothic', 'Snap ITC', 'Stencil', 'Viner
 Hand ITC', 'Vivaldi', 'Vladimir Script', 'Wide Latin', 'Tw Cen MT', 'Tw Cen
 MT Condensed', 'Script MT Bold', 'Rockwell Extra Bold', 'Rockwell Condensed
', 'Rockwell', 'Rage Italic', 'Perpetua Titling MT', 'Perpetua', 'Palace Scr
ipt MT', 'OCR A Extended', 'Maiandra GD', 'Lucida Sans Typewriter', 'Lucida
Sans', 'Imprint MT Shadow', 'Haettenschweiler', 'Goudy Stout', 'Goudy Old St
yle', 'Gloucester MT Extra Condensed', 'Gill Sans Ultra Bold Condensed', 'Gi
ll Sans Ultra Bold', 'Gill Sans MT Condensed', 'Gill Sans MT', 'Gill Sans MT
 Ext Condensed Bold', 'Gigi', 'Franklin Gothic Medium Cond', 'Franklin Gothi
c Heavy', 'Franklin Gothic Demi Cond', 'Franklin Gothic Demi', 'Franklin Got
hic Book', 'Forte', 'Felix Titling', 'Eras Medium ITC', 'Eras Light ITC', 'E
ras Demi ITC', 'Eras Bold ITC', 'Engravers MT', 'Elephant', 'Edwardian Scrip
t ITC', 'Curlz MT', 'Copperplate Gothic Light', 'Copperplate Gothic Bold', '
Century Schoolbook', 'Castellar', 'Calisto MT', 'Bookman Old Style', 'Bodoni
 MT Condensed', 'Bodoni MT Black', 'Bodoni MT', 'Blackadder ITC', 'Arial Rou
nded MT Bold', 'Agency FB', 'Bookshelf Symbol 7', 'MS Reference Sans Serif',
 'MS Reference Specialty', 'Berlin Sans FB Demi', 'Tw Cen MT Condensed Extra
 Bold')
        self.panel3.add(self.fontoptionmenu)

        # set current font to None because there is no font selected
        self.currentfont = None

        # make Font Style option menu
        self.fontstylevariable = StringVar(self.panel3)
        self.fontstylevariable.set("Please select the Font Style of your cho
ice")
        self.fontstyleoptionmenu = OptionMenu(self.panel3, self.fontstylevar
iable, "Regular", "Bold", "Italic", "Underline", "Overstrike")
        self.panel3.add(self.fontstyleoptionmenu)

        # set current font style to None because there is no font selected
        self.currentfontstyle = None

        # make Font Style option menu
        self.fontsizevariable = StringVar(self.panel3)
        self.fontsizevariable.set("Please select the Font Size of your choic
e")
```

```
        self.fontsizeoptionmenu = OptionMenu(self.panel3, self.fontsizevaria
ble, 8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 26, 28, 36, 48, 72)
        self.panel3.add(self.fontsizeoptionmenu)

        # set current font size to None because there is no font selected
        self.currentfontsize = None

        # make Apply button to apply font changes if font has been configura
ted
        self.applybutton = Button(self.panel3, text = "Apply", command = sel
f.applyChanges)
        self.panel3.add(self.applybutton)

        # make invisible label so Apply button will not expand
        self.invisiblelabel = Label(self.panel3, text = "")
        self.panel3.add(self.invisiblelabel)
```

To confirm font changes, button and custom function for applying font changes must be created. The code for creating a button have been stated above.

```
    def applyChanges(self):
        # get value from Font, Font Style, and Font Size option menu
        self.currentfont = self.fontvariable.get()
        self.currentfontstyle = self.fontstylevariable.get()
        self.currentfontsize = self.fontsizevariable.get()

        # set font changes
        if self.currentfontstyle == "Regular":
            self.fontsettings = Font(family = self.currentfont, size = self.
currentfontsize, weight = "normal")
        elif self.currentfontstyle == "Bold":
            self.fontsettings = Font(family = self.currentfont, size = self.
currentfontsize, weight = "bold")
        elif self.currentfontstyle == "Italic":
            self.fontsettings = Font(family = self.currentfont, size = self.
currentfontsize, weight = "normal", slant = "italic")
        elif self.currentfontstyle == "Underline":
            self.fontsettings = Font(family = self.currentfont, size = self.
currentfontsize, weight = "normal", underline = 1)
        elif self.currentfontstyle == "Overstrike":
            self.fontsettings = Font(family = self.currentfont, size = self.
currentfontsize, weight = "normal", overstrike = 1)
        self.text.config(font = self.fontsettings)
```

When applying font changes, user must save the file in order to keep the following settings and to open the corresponding file with the saved settings. The syntax can be seen at

the custom functions “fileSave(self)”, “fileSaveAs(self)” and “fileOpen(self)” as typed above.

Finally, what is left to do is to create the text with a vertical scroll bar and place it in the panel with vertical orientation which is as follows.

```
        # make text
        self.text = Text(self.panel2, undo = True)
        self.panel2.add(self.text)

        # make vertical scroll bar for text
        self.textscrollbar = Scrollbar(self.text, command = self.text.yview)
        self.textscrollbar.pack(side = RIGHT, fill = Y)

        # scroll bar will adjust according to text content
        self.text.config(yscrollcommand = self.textscrollbar.set)
```

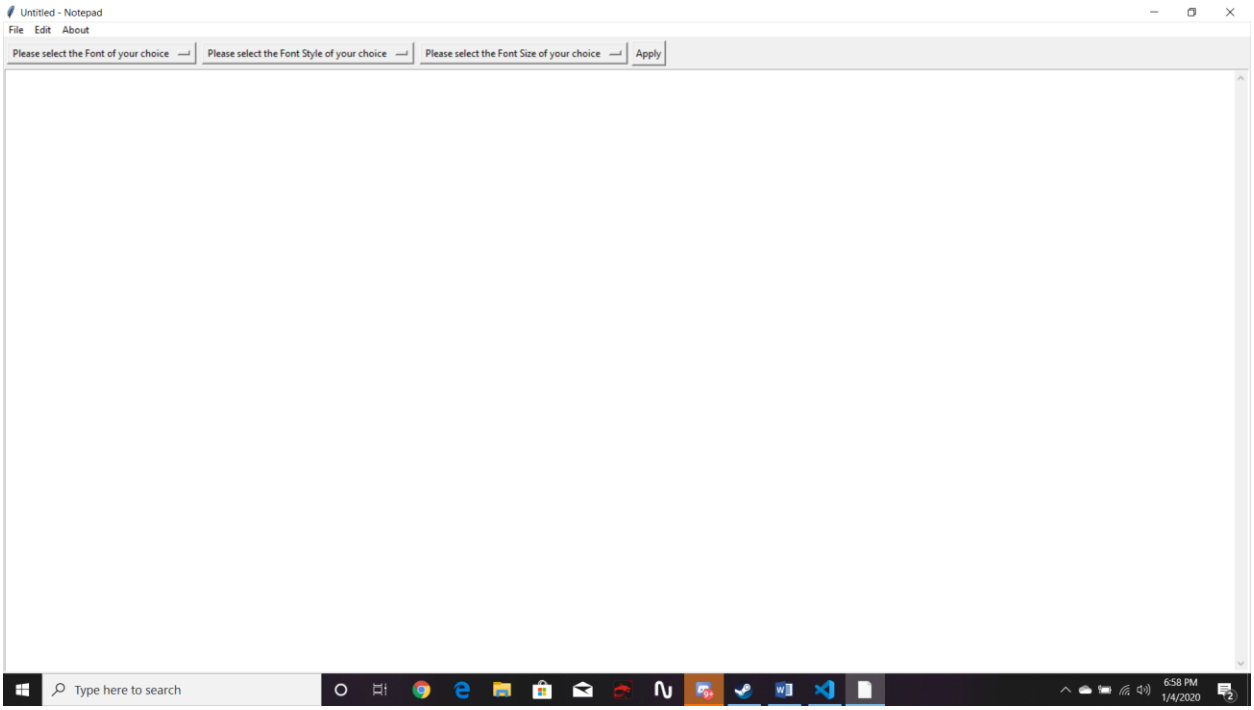
Now, the application is fully functional with the features mentioned above.

To run the application, new object will be created and run as seen below.

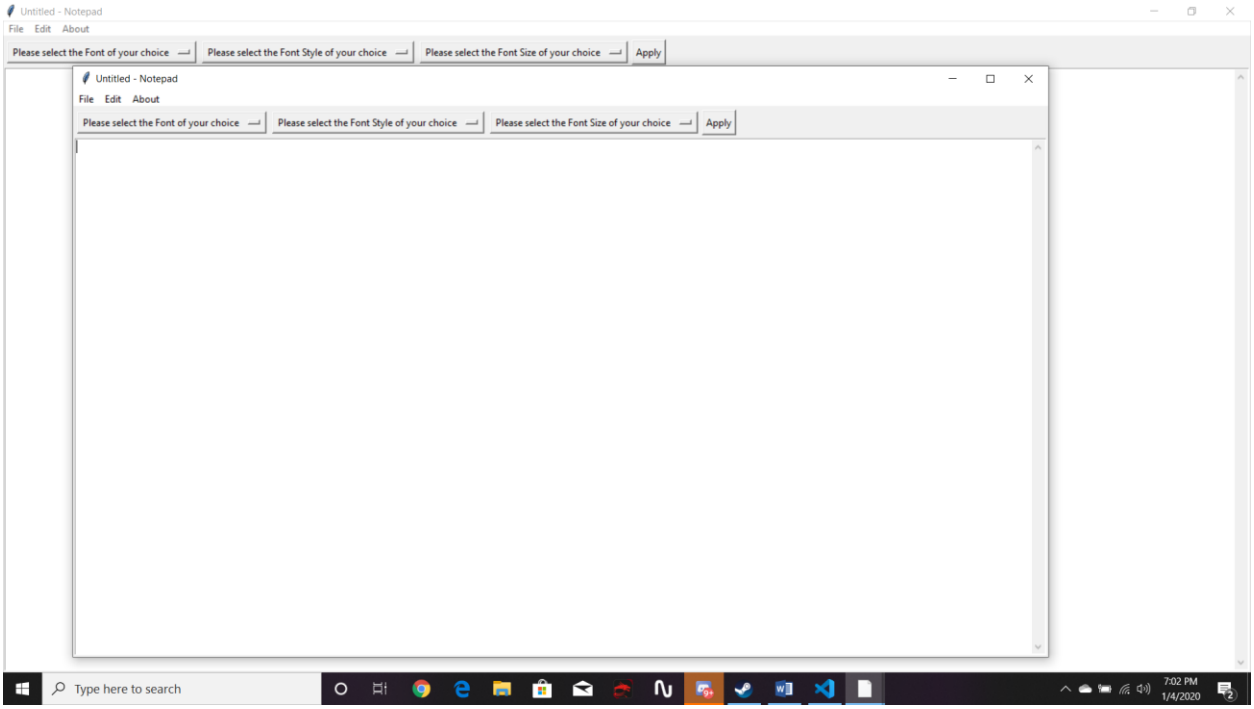
```
# make new object for running notepad
notepad = NotepadFrame()

# run notepad
notepad.mainloop()
```

4. Evidence of Working Program

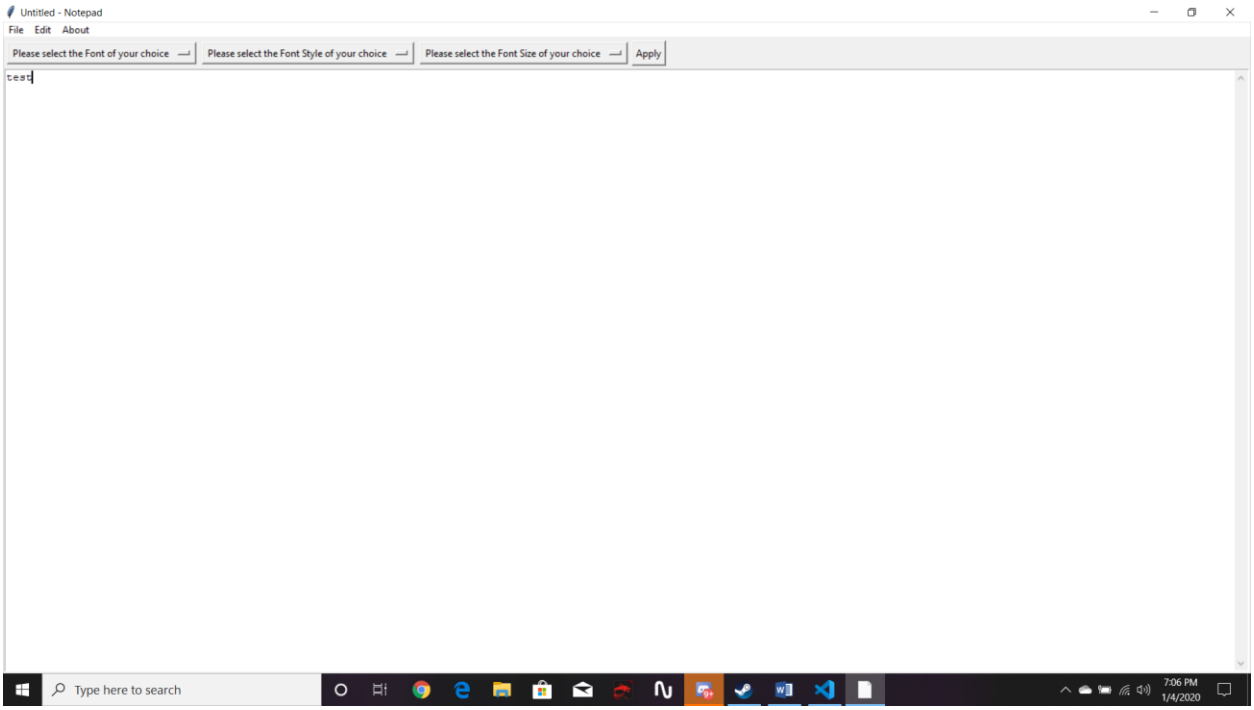
Screenshot of opening application:

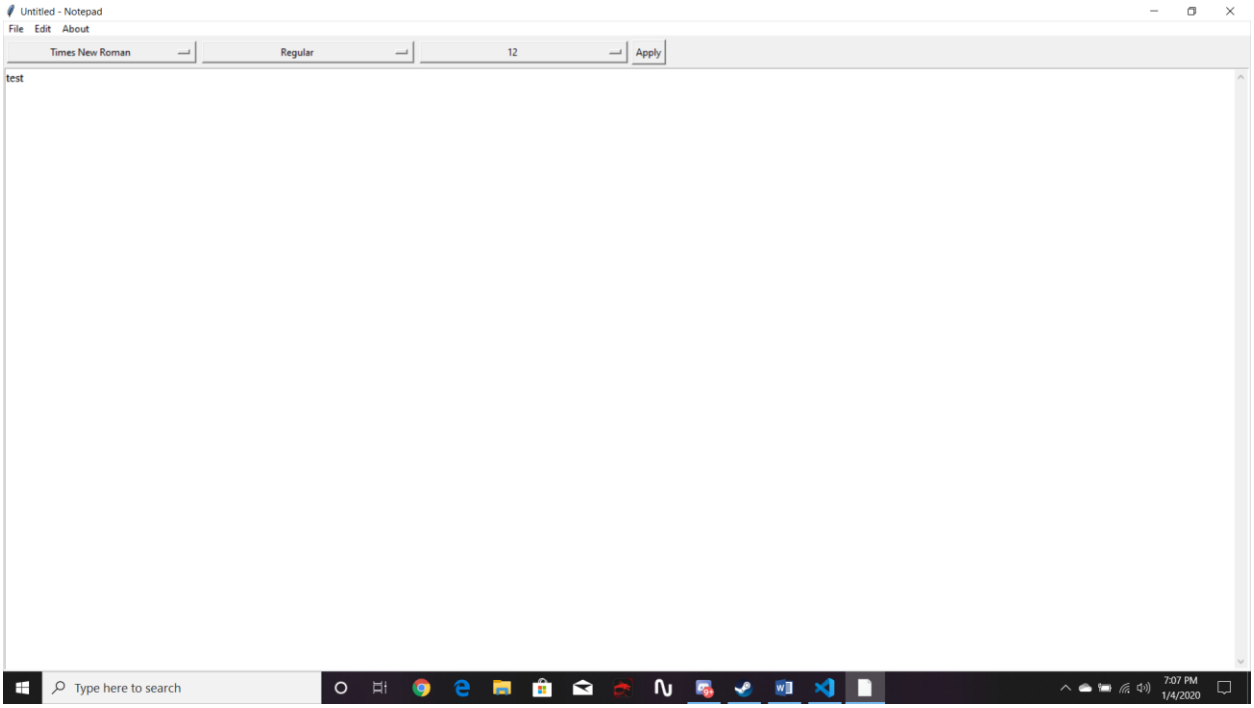


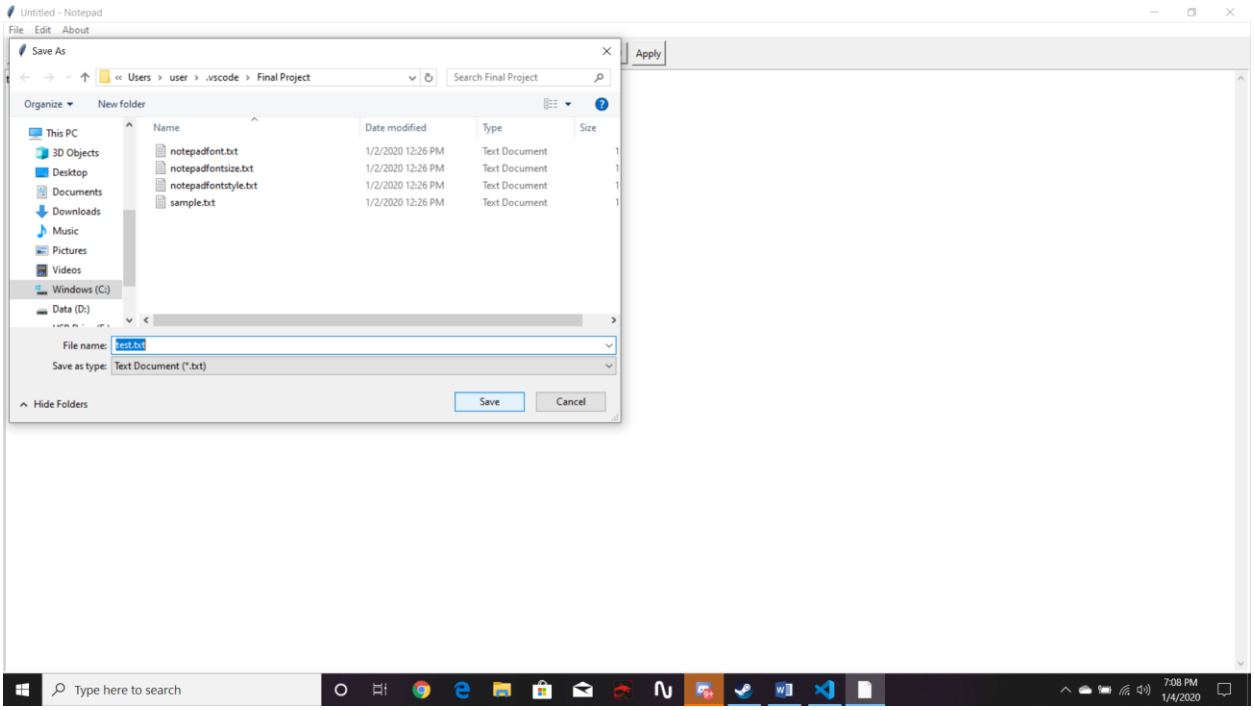
Screenshot of opening new window:

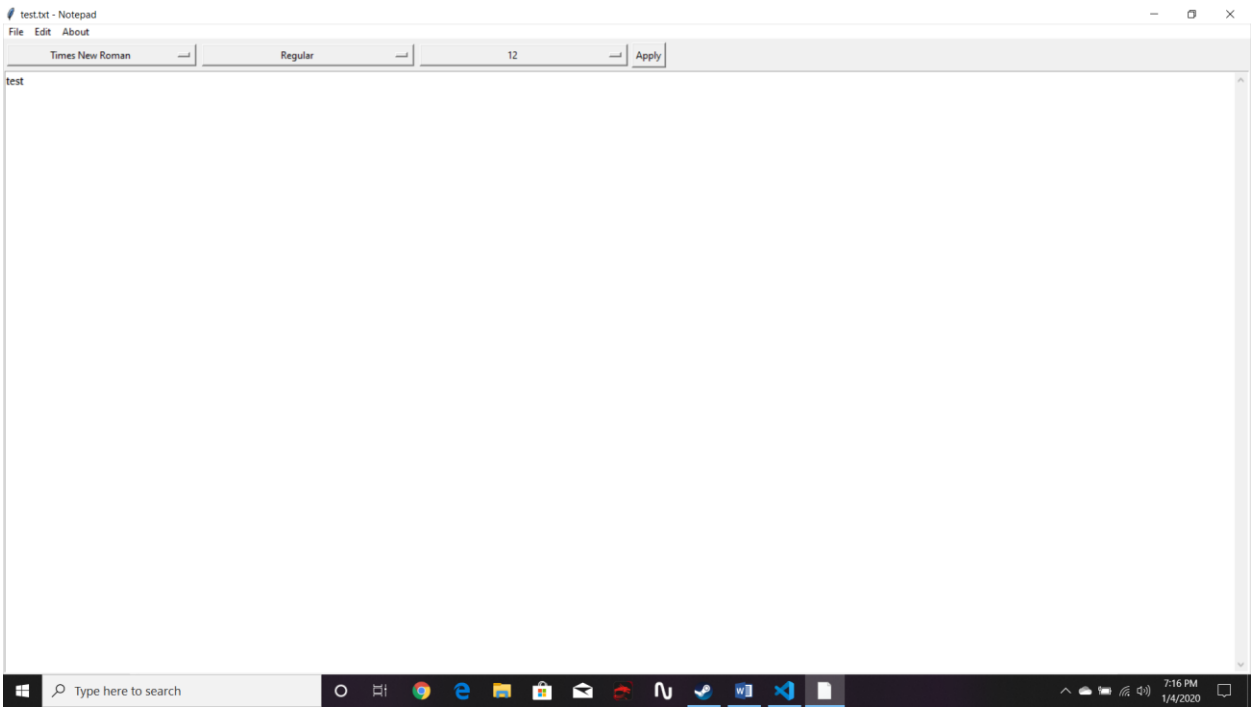


Screenshot of saving file after changing font settings:

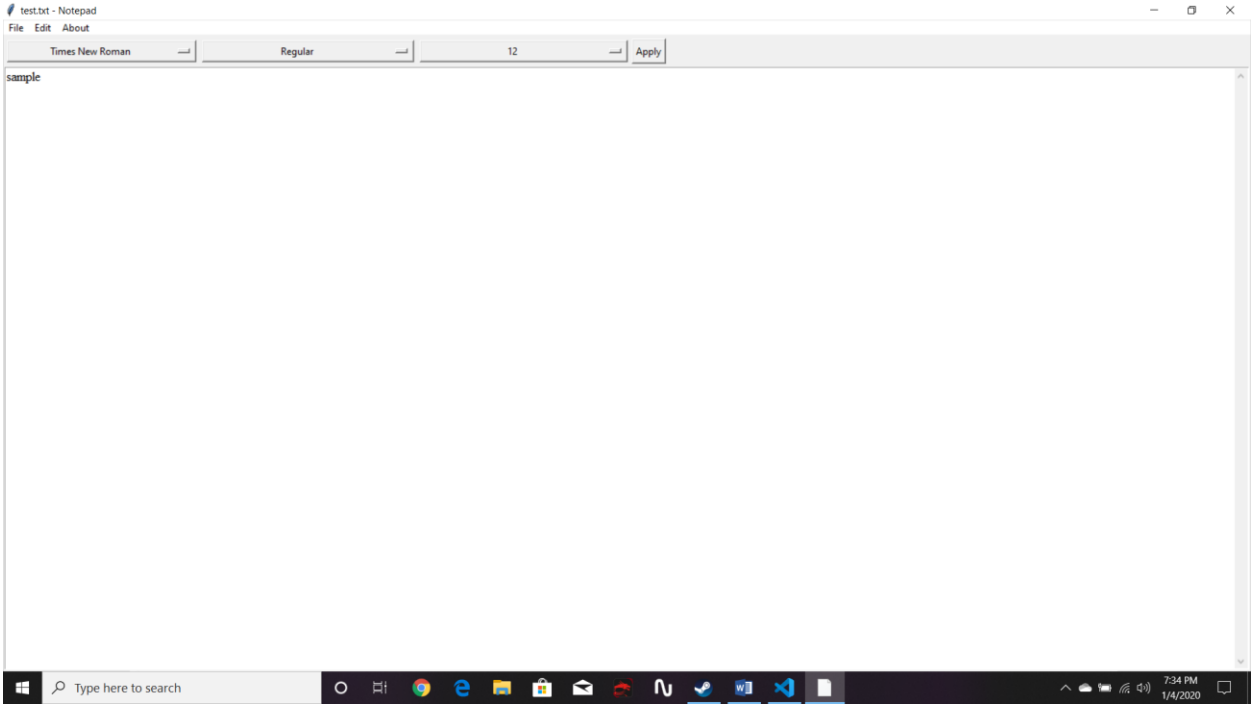


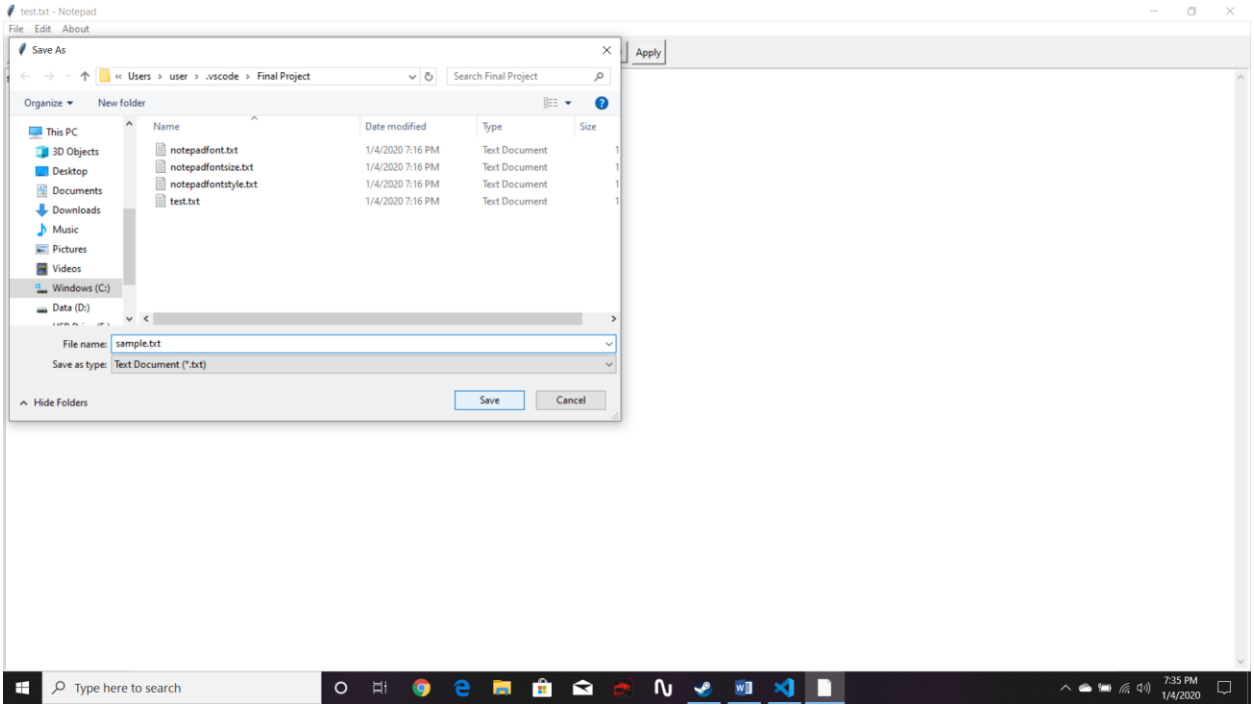

Untitled - Notepad
File Edit About
Times New Roman
Regular
12
Apply
test
Type here to search
7:07 PM
1/4/2020

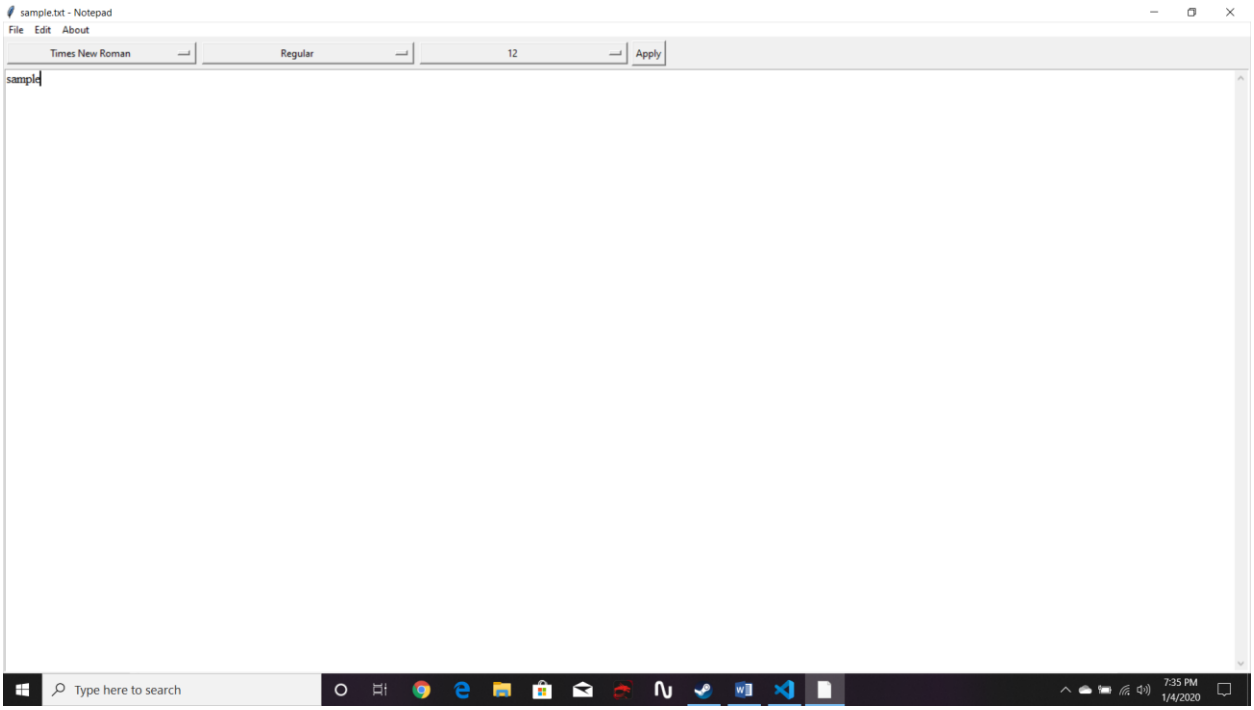

Untitled - Notepad
File Edit About
Apply
Save As
Users › user › .vscode › Final Project
Search Final Project
Organize
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
Data (D:)
Name
Date modified
Type
Size
notepadfont.txt 1/2/2020 12:26 PM Text Document
notepadfontsize.txt 1/2/2020 12:26 PM Text Document
notepadfontstyle.txt 1/2/2020 12:26 PM Text Document
sample.txt 1/2/2020 12:26 PM Text Document
File name: test.txt
Save as type: Text Document (*.txt)
Hide Folders
Save
Cancel
Type here to search
7:08 PM
1/4/2020



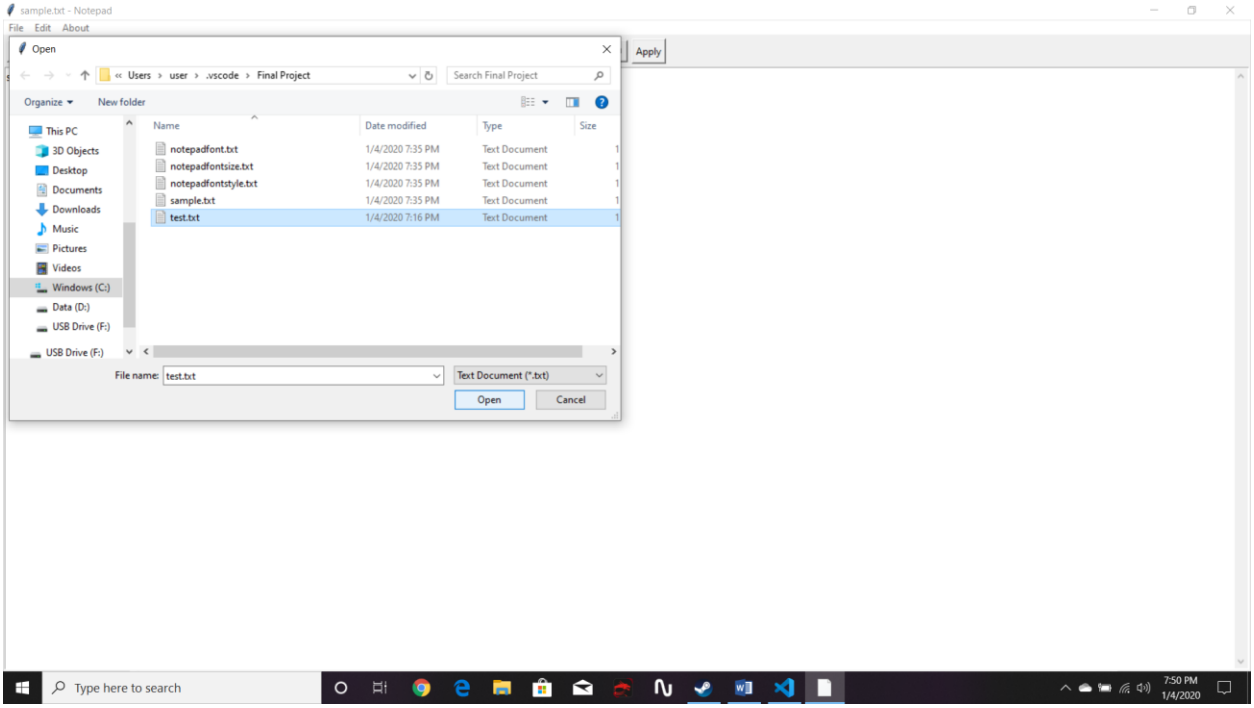
Screenshot of saving as new file:

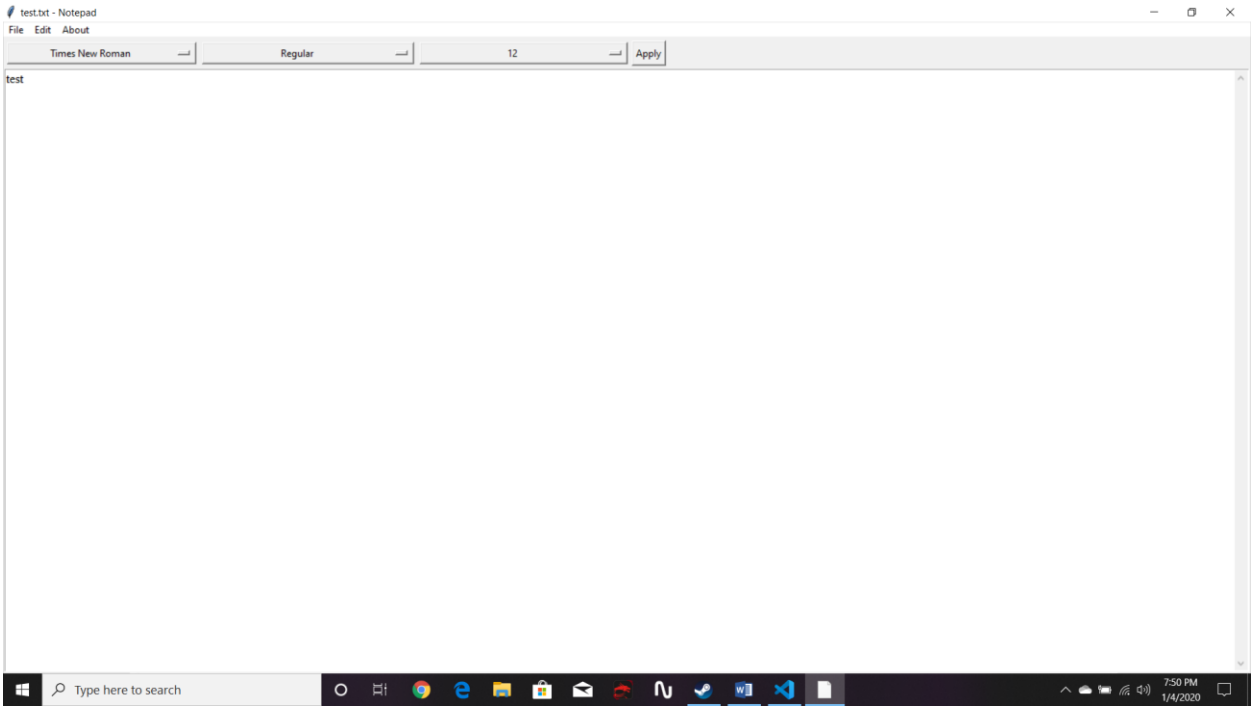



test.txt - Notepad
File Edit About
Apply
Save As
« Users › user › .vscode › Final Project
Search Final Project
Organize
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
Data (D:)
Name
Date modified
Type
Size
notepadfont.txt 1/4/2020 7:16 PM Text Document
notepadfontsize.txt 1/4/2020 7:16 PM Text Document
notepadfontstyle.txt 1/4/2020 7:16 PM Text Document
test.txt 1/4/2020 7:16 PM Text Document
File name: sample.txt
Save as type: Text Document (*.txt)
Hide Folders
Save
Cancel
Type here to search
7:35 PM
1/4/2020

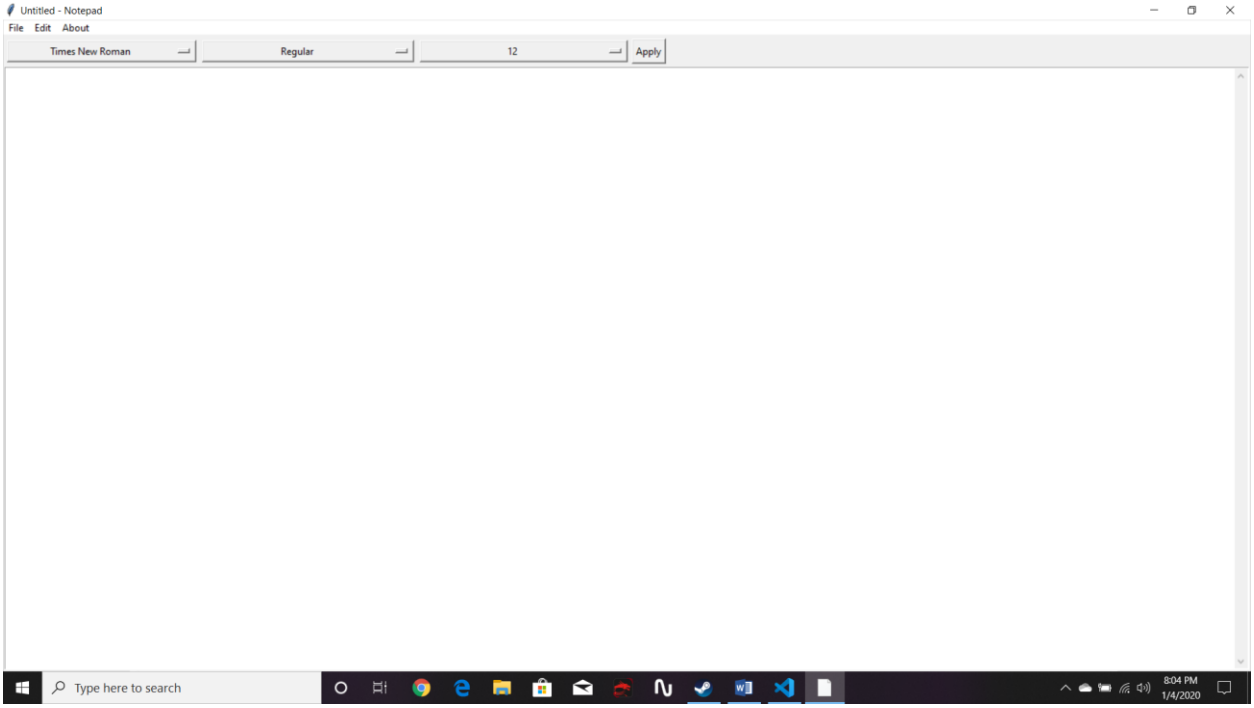

sample.txt - Notepad
File Edit About
Times New Roman
Regular
12
Apply
sample
Type here to search
7:35 PM
1/4/2020

Screenshot of opening existing file:

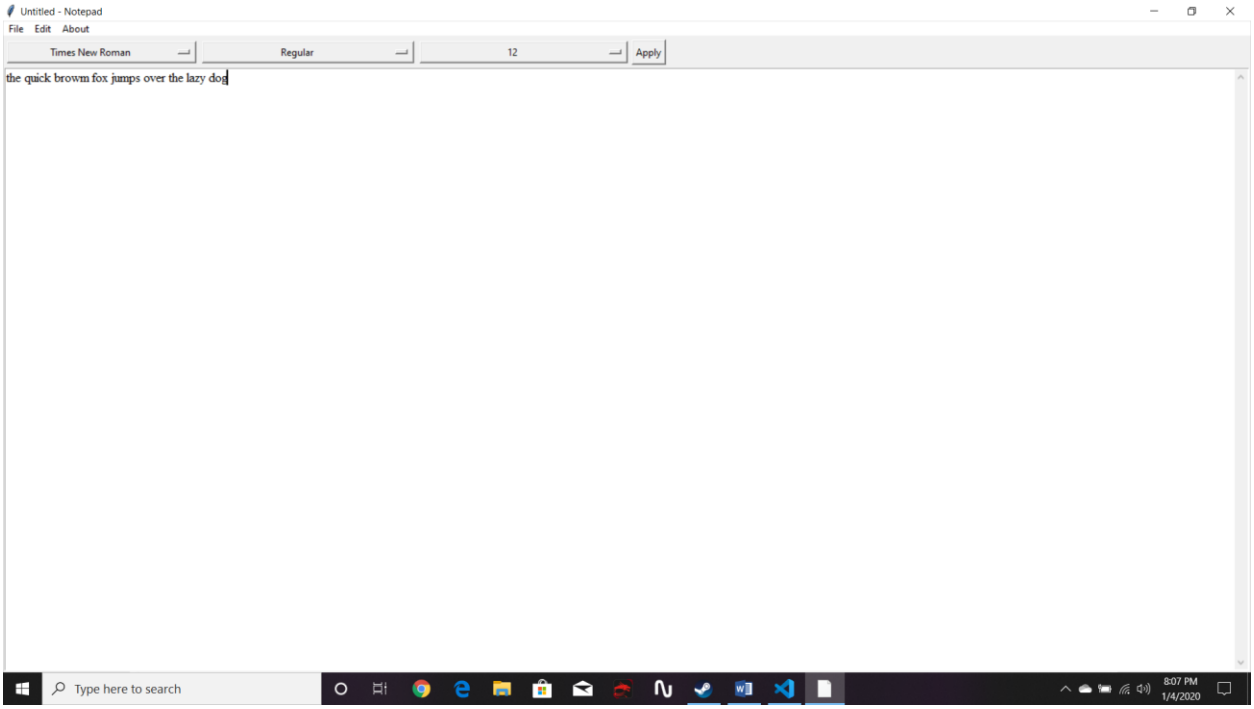


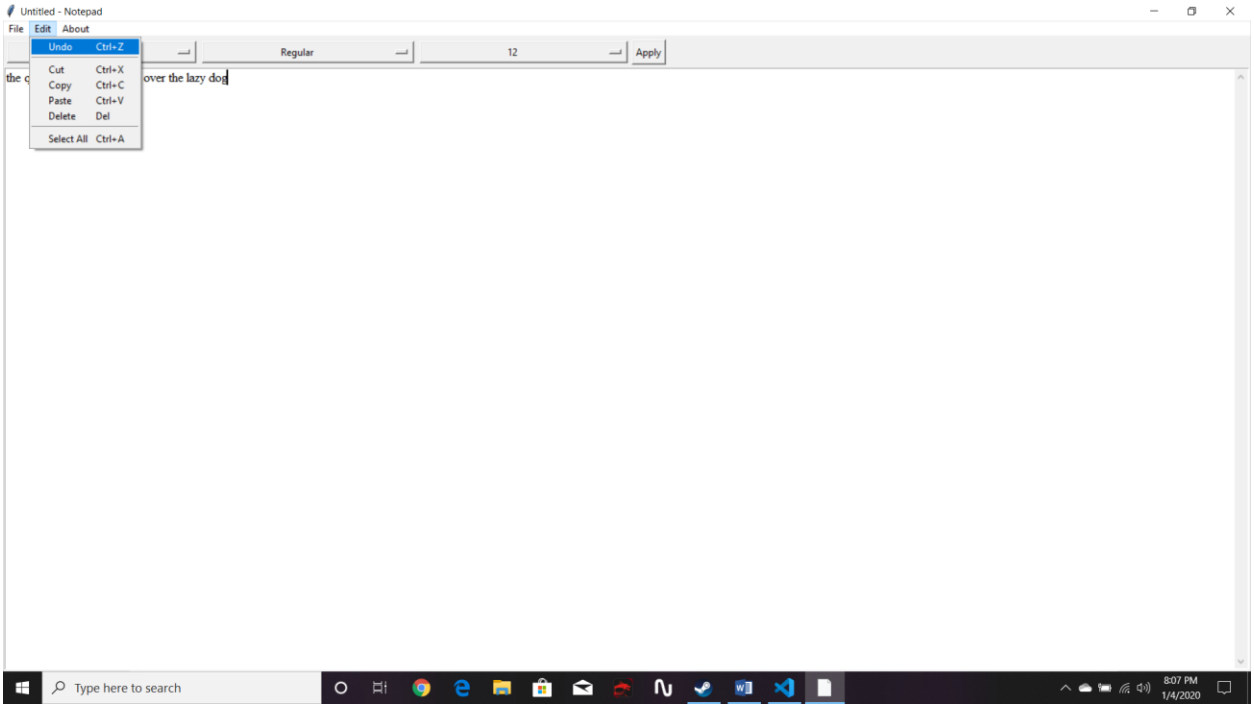


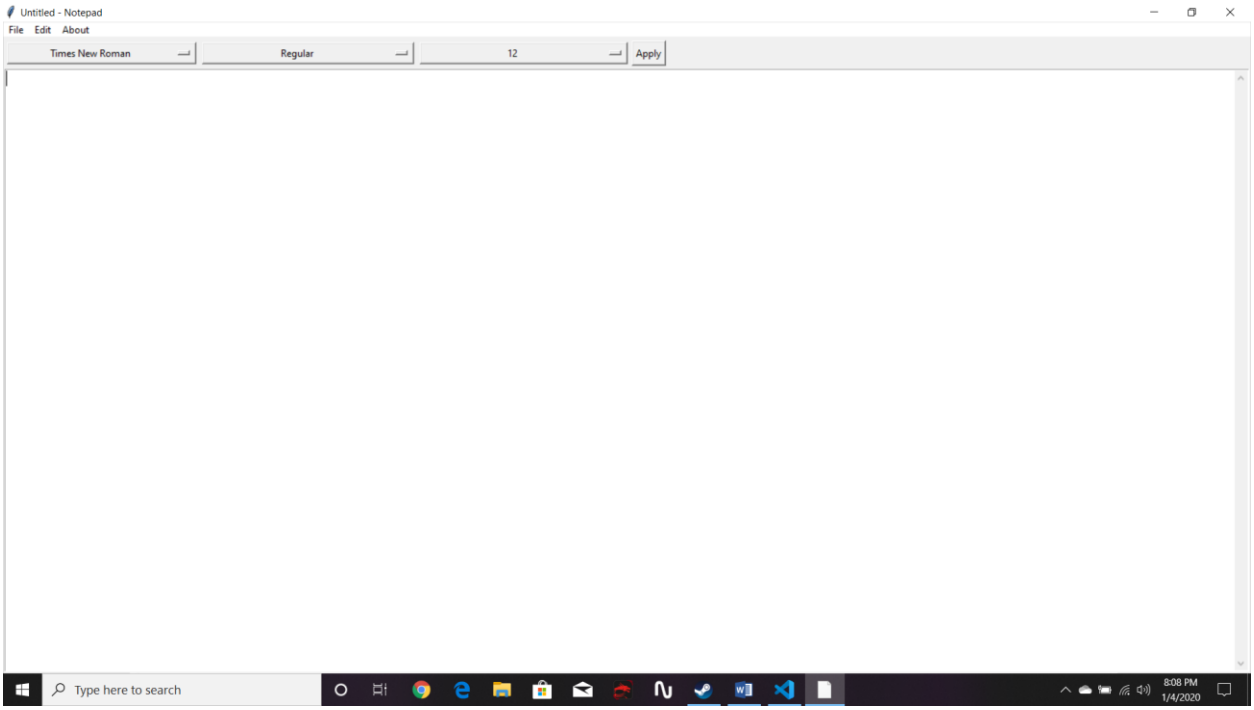
Screenshot of opening new file:



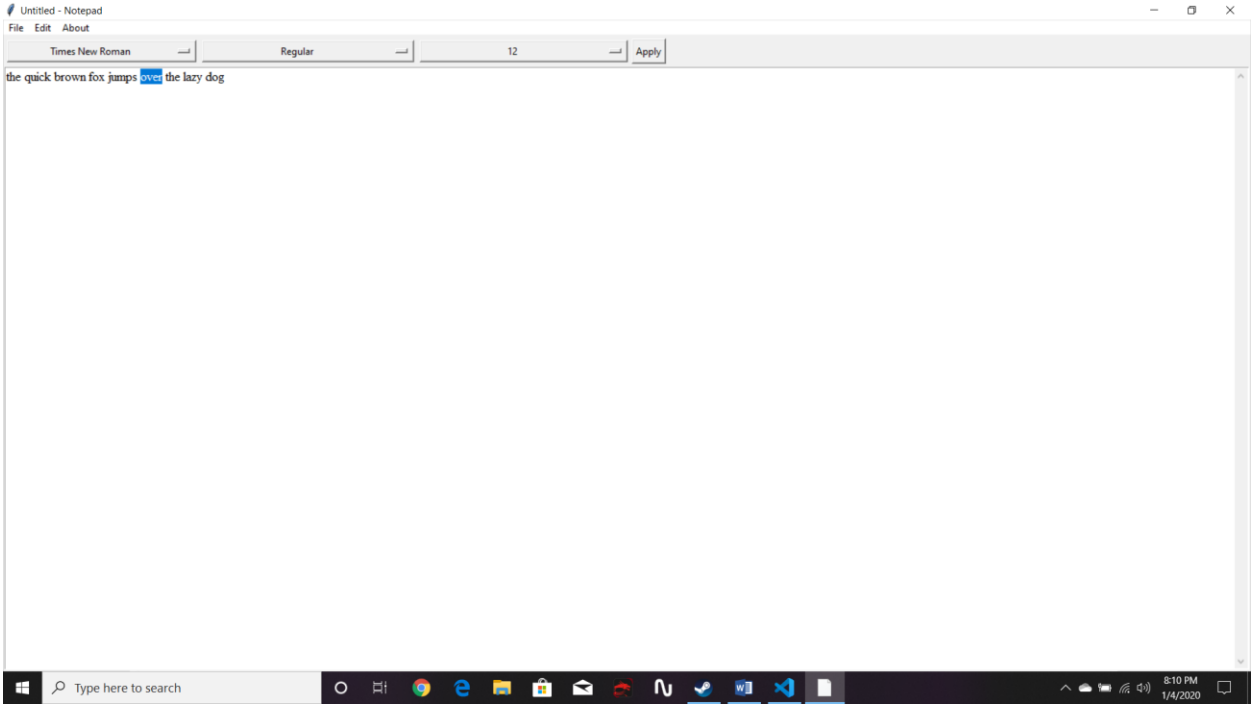
Screenshot of undoing text edit:

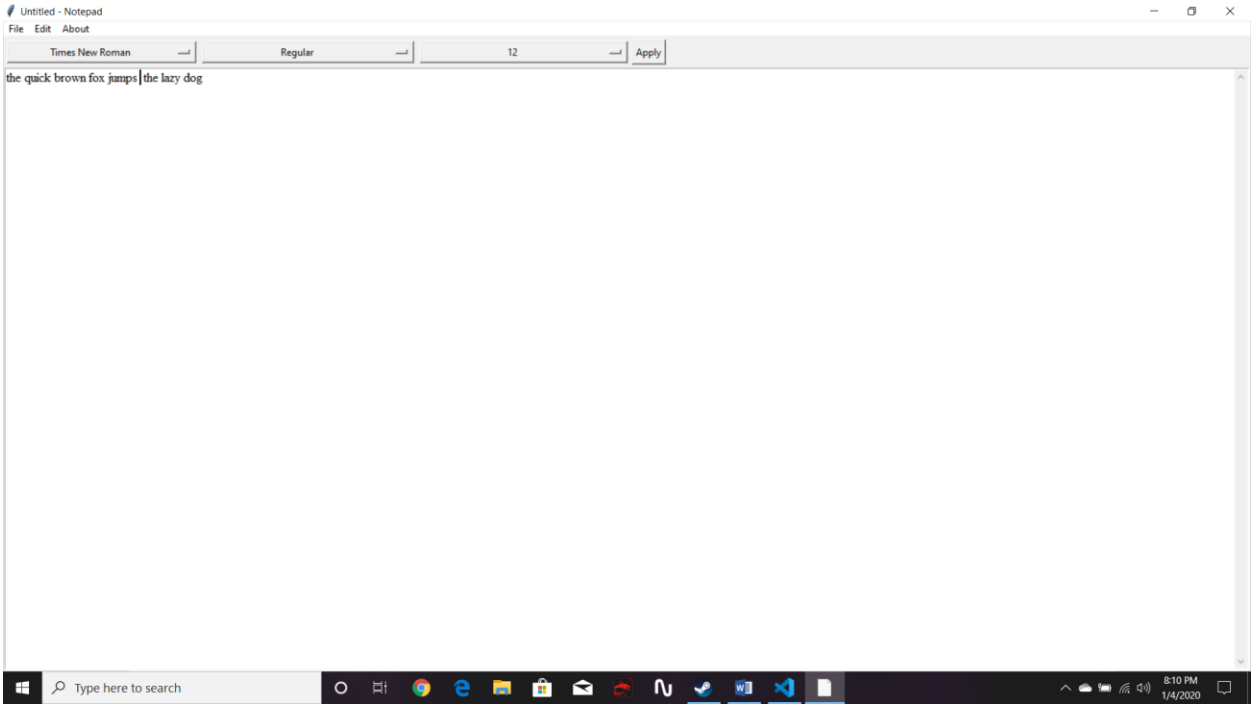


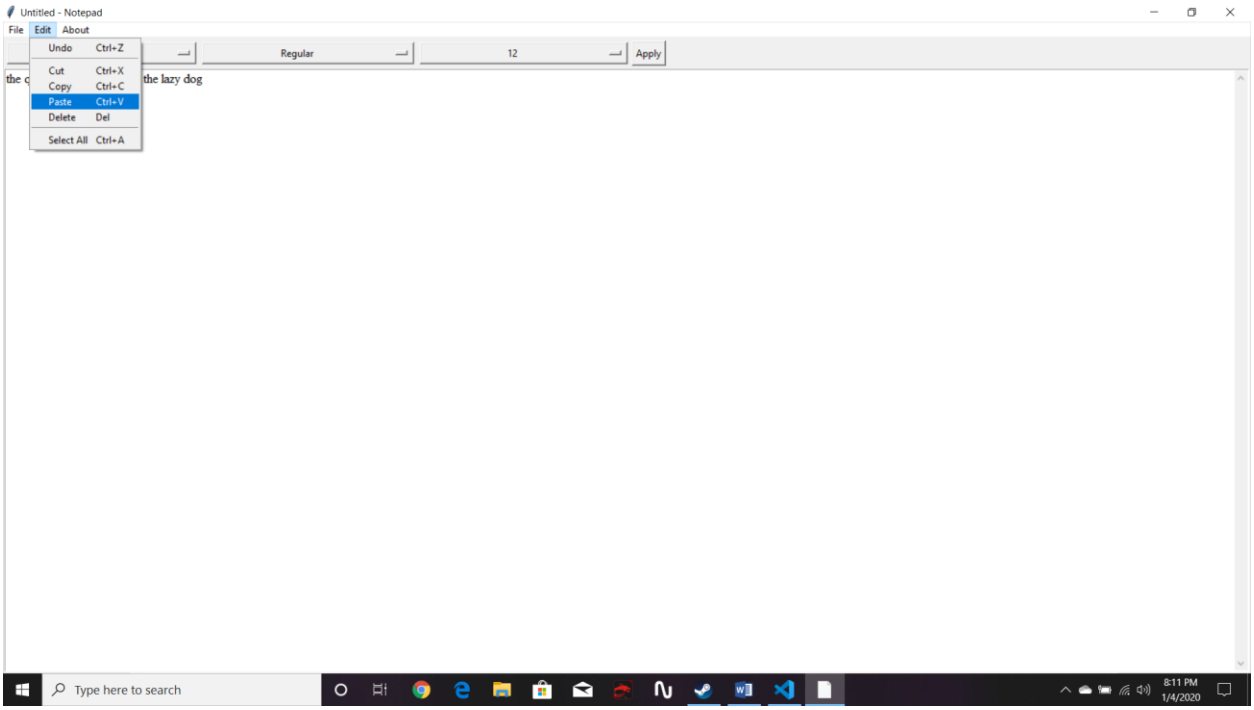

Untitled - Notepad
File Edit About
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Select All Ctrl+A
Regular
12
Apply
over the lazy dog
Type here to search
8:07 PM
1/4/2020

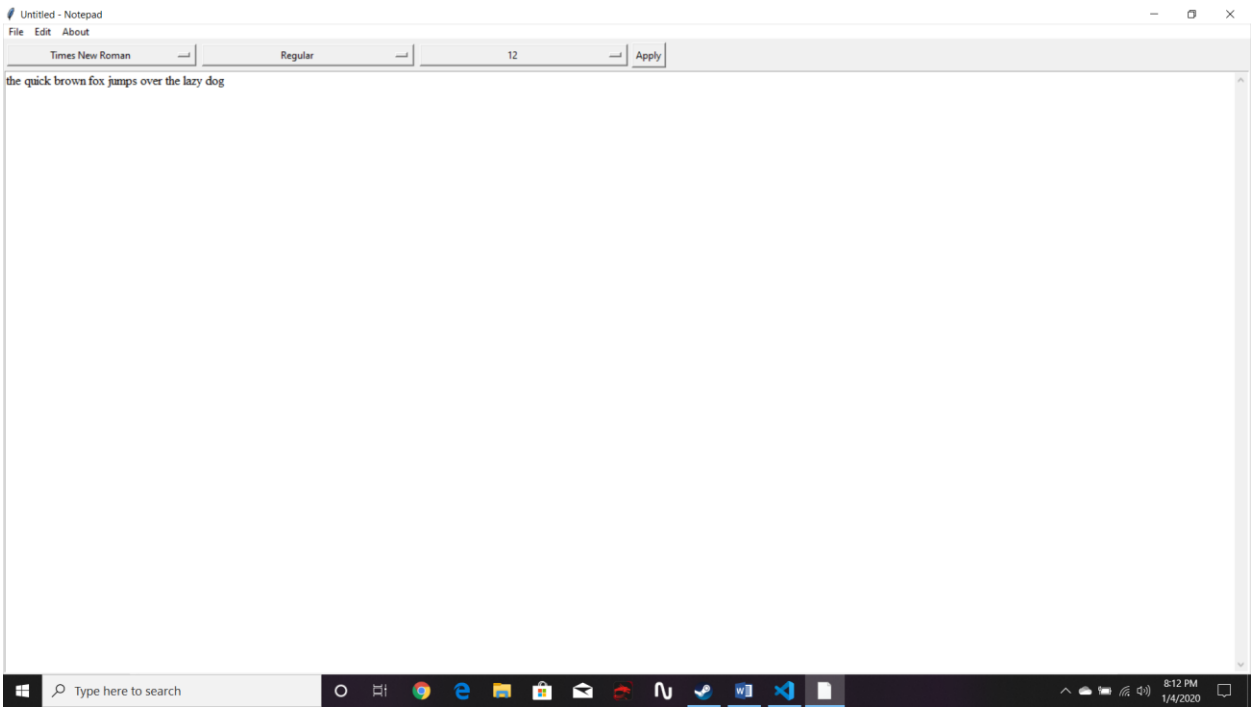

Untitled - Notepad
File Edit About
Times New Roman
Regular
12
Apply
Type here to search
8:08 PM
1/4/2020

Screenshot of cutting and pasting text:

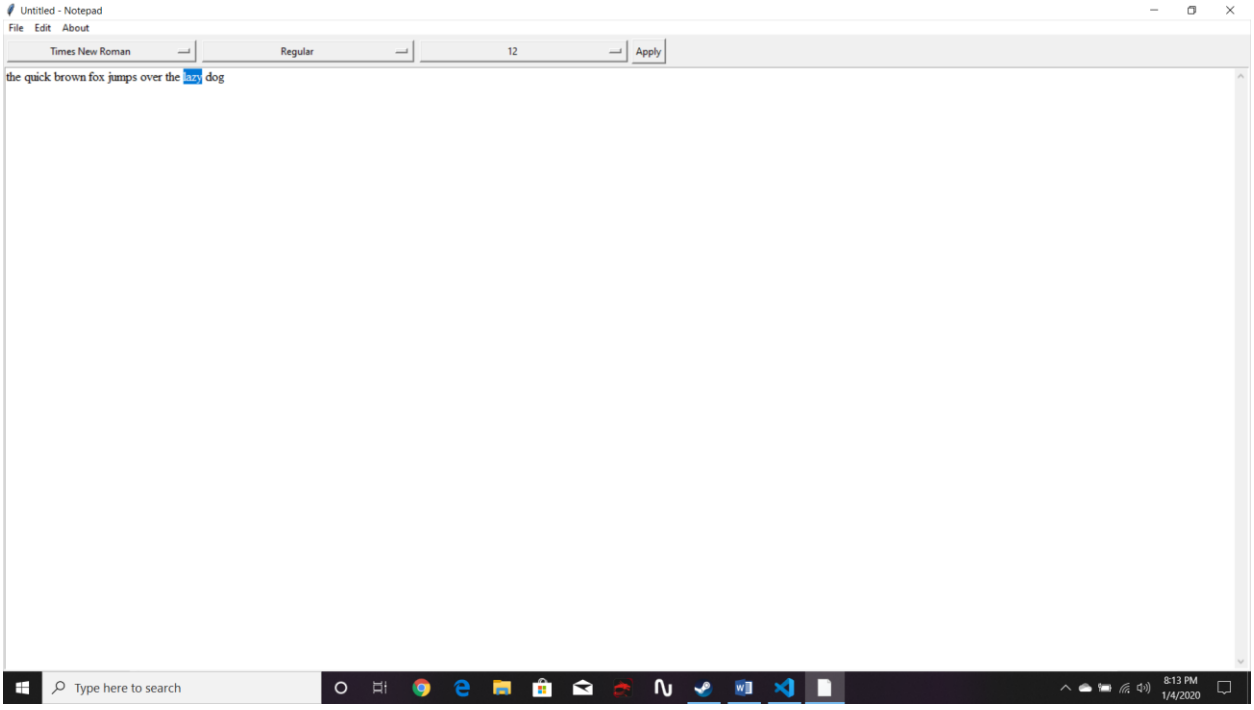


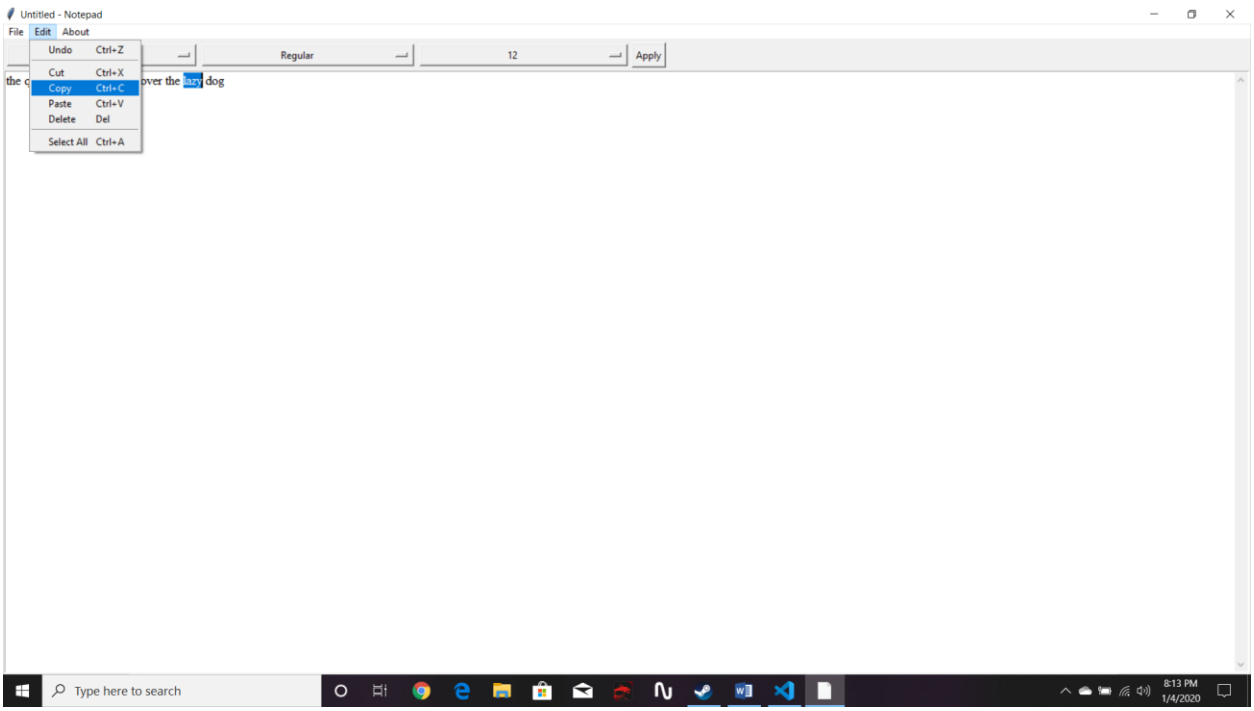

Untitled - Notepad
File Edit About
Times New Roman
Regular
12
Apply
the quick brown fox jumps the lazy dog
Type here to search
8:10 PM
1/4/2020

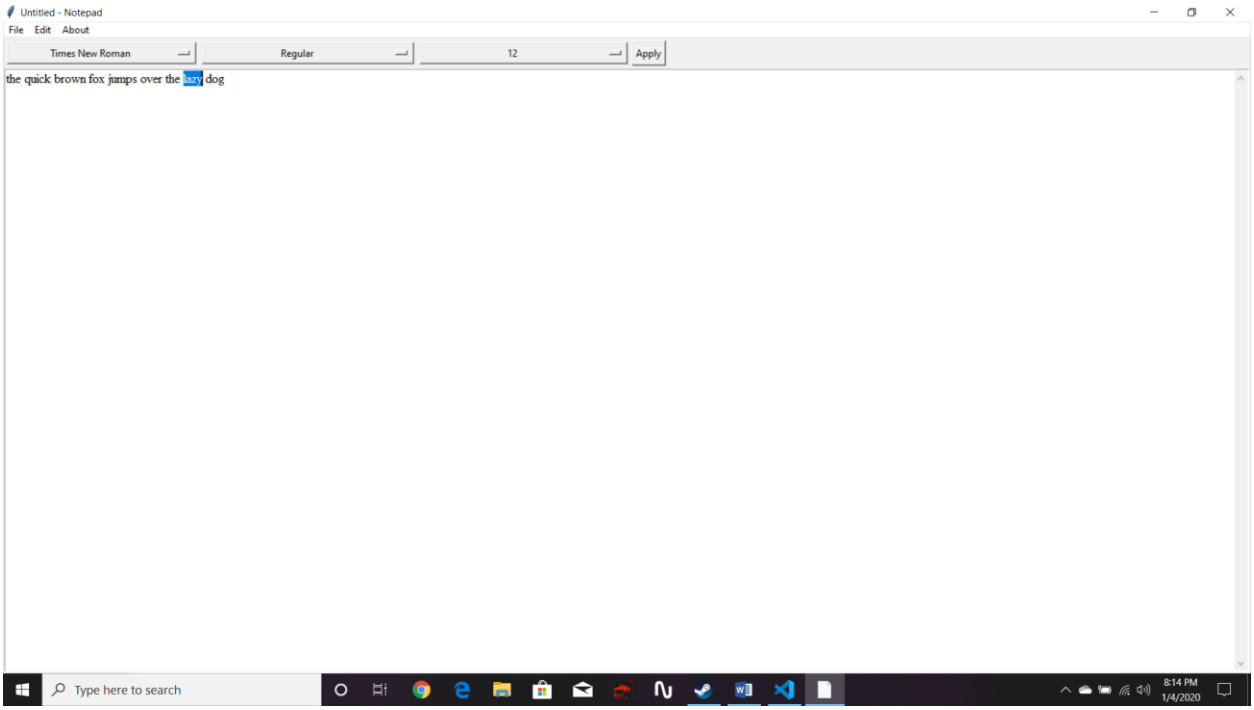

Untitled - Notepad
File Edit About
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Select All Ctrl+A
Regular
12
Apply
the lazy dog
Type here to search
8:11 PM
1/4/2020

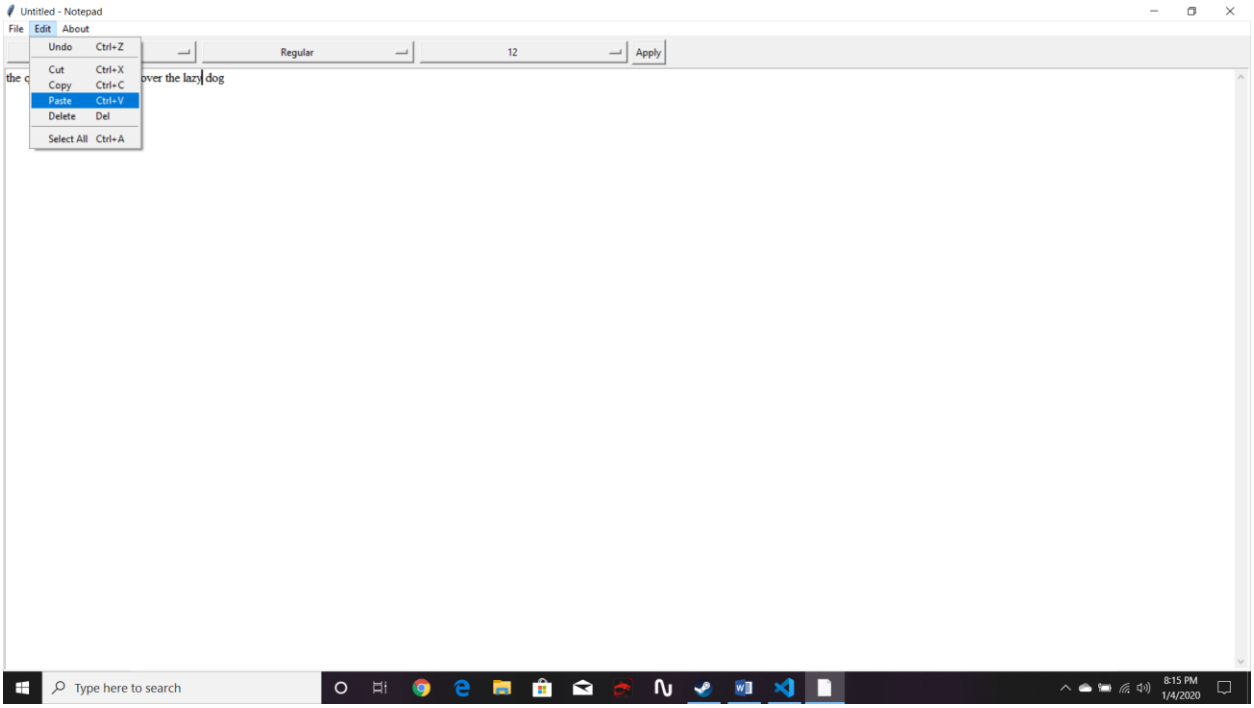


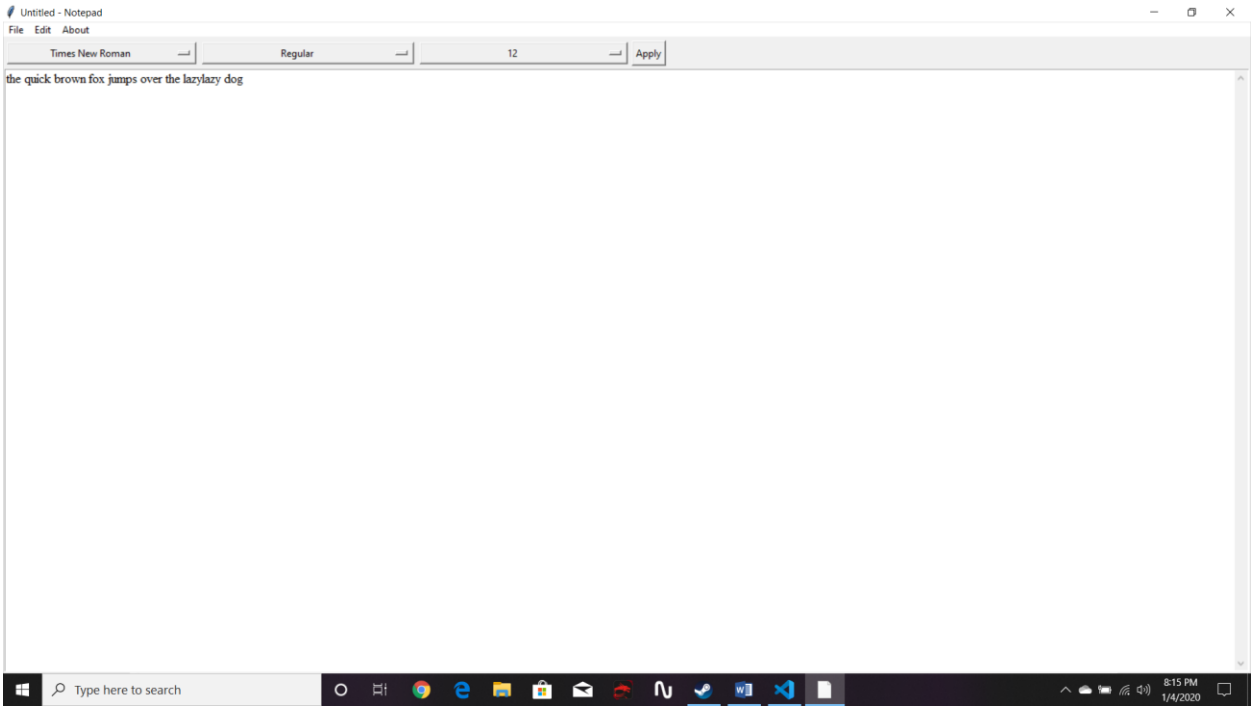
Screenshot of copying and pasting text:



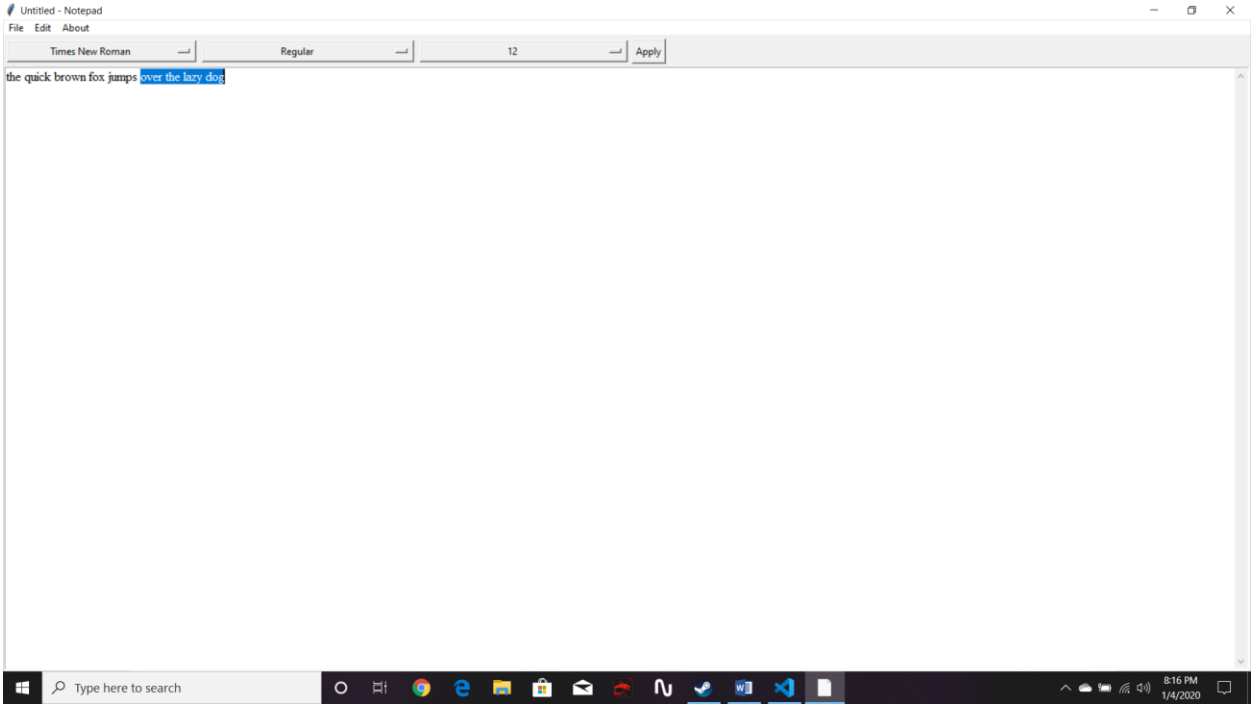

Untitled - Notepad
File Edit About
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Select All Ctrl+A
Regular
12
Apply
over the lazy dog
Type here to search
8:13 PM
1/4/2020

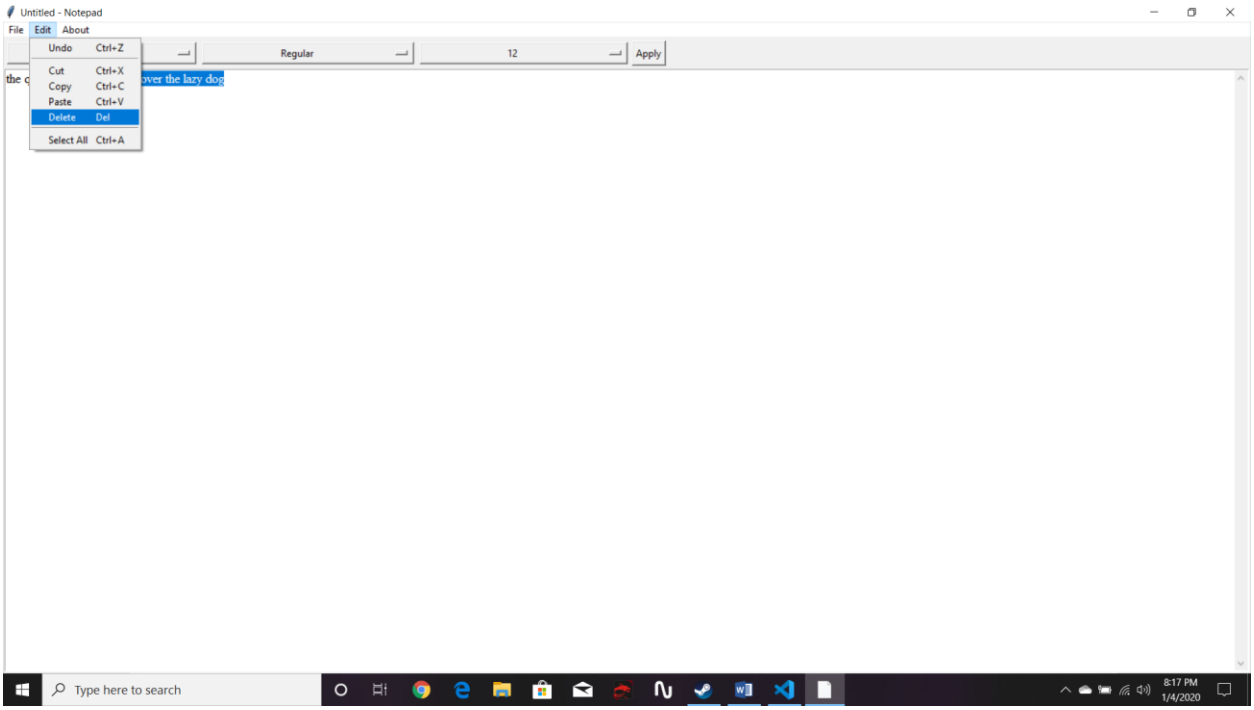

Untitled - Notepad
File Edit About
Times New Roman
Regular
12
Apply
the quick brown fox jumps over the lazy dog
Type here to search
8:14 PM
1/4/2020

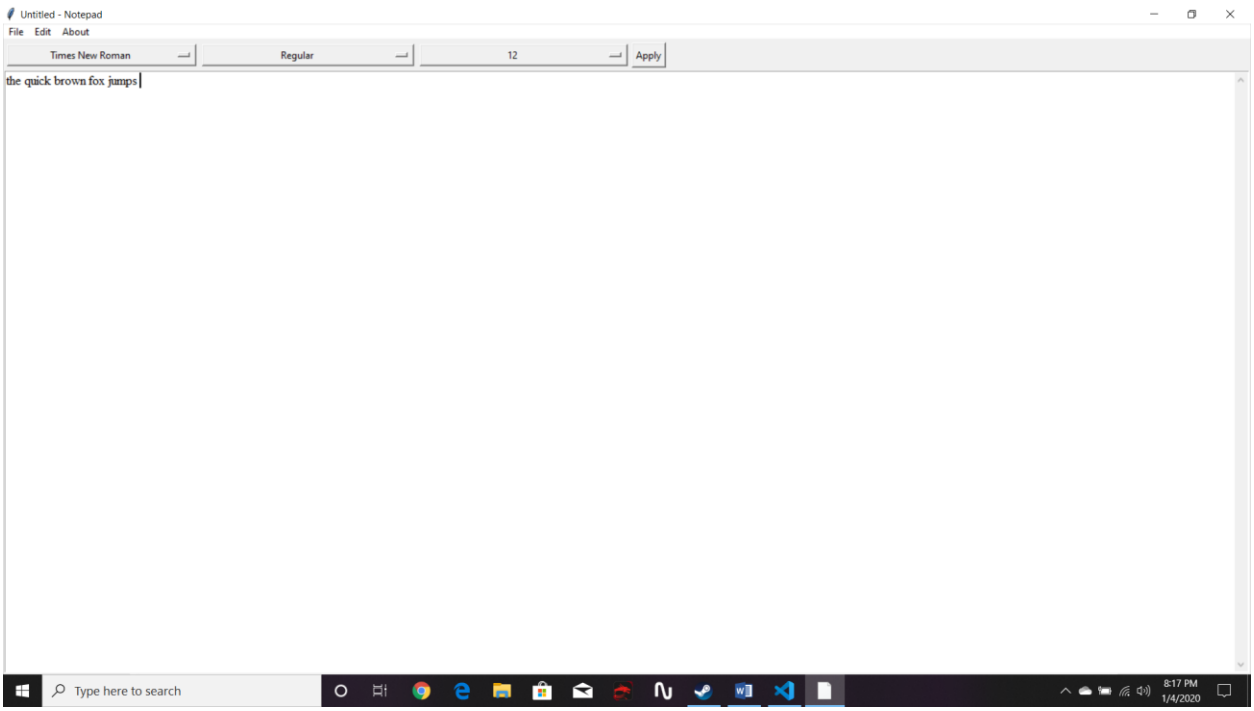

Untitled - Notepad
File Edit About
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Select All Ctrl+A
Regular
12
Apply
over the lazy dog
Type here to search
8:15 PM
1/4/2020


Untitled - Notepad
File Edit About
Times New Roman
Regular
12
Apply
the quick brown fox jumps over the lazylazy dog
Type here to search
8:15 PM
1/4/2020

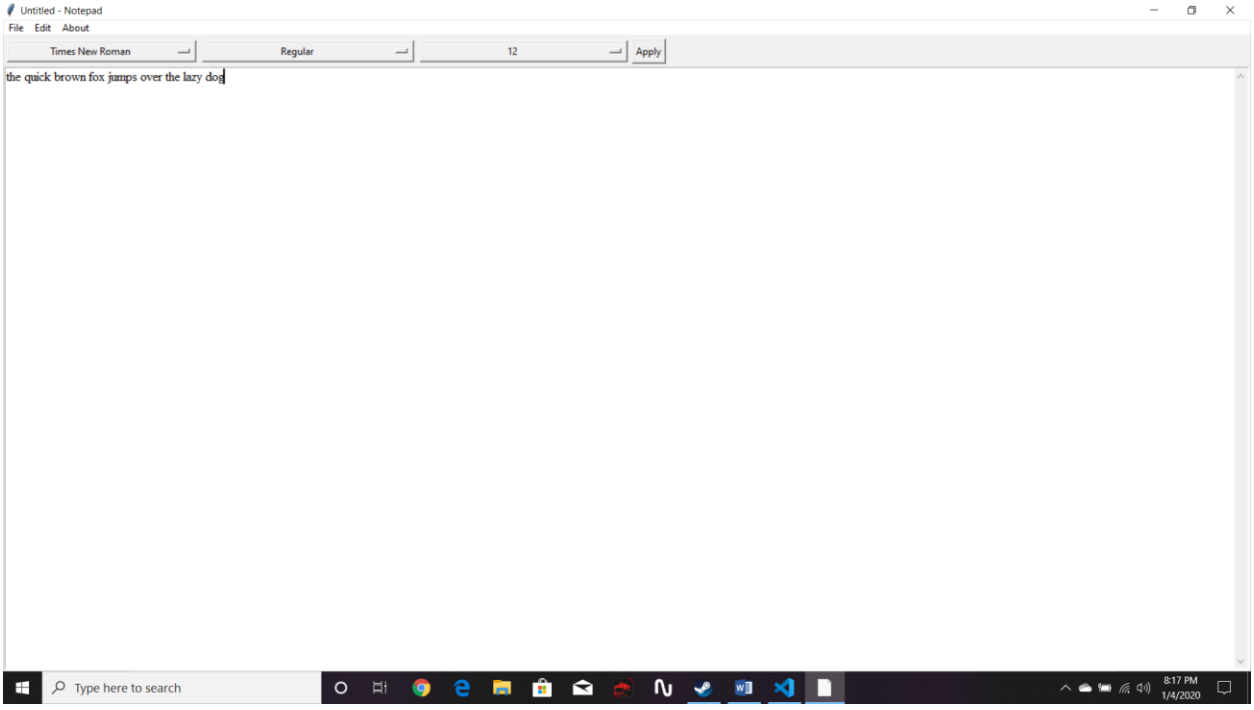
Screenshot of deleting text:

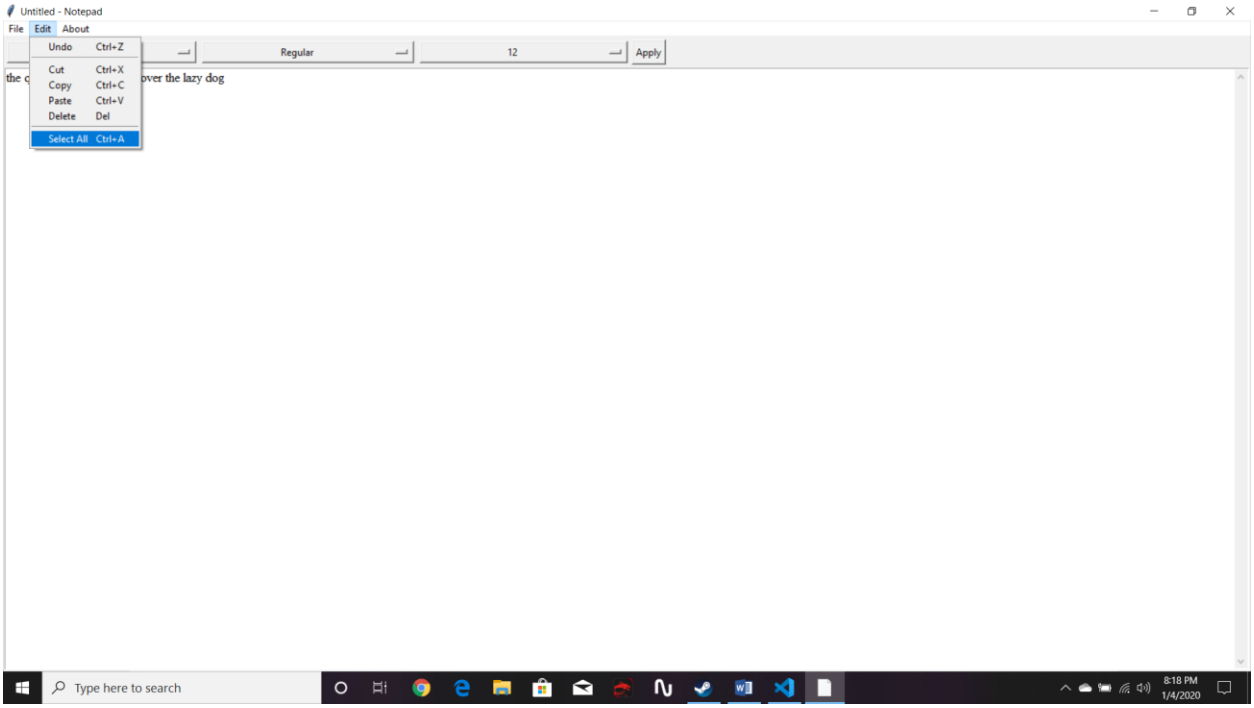


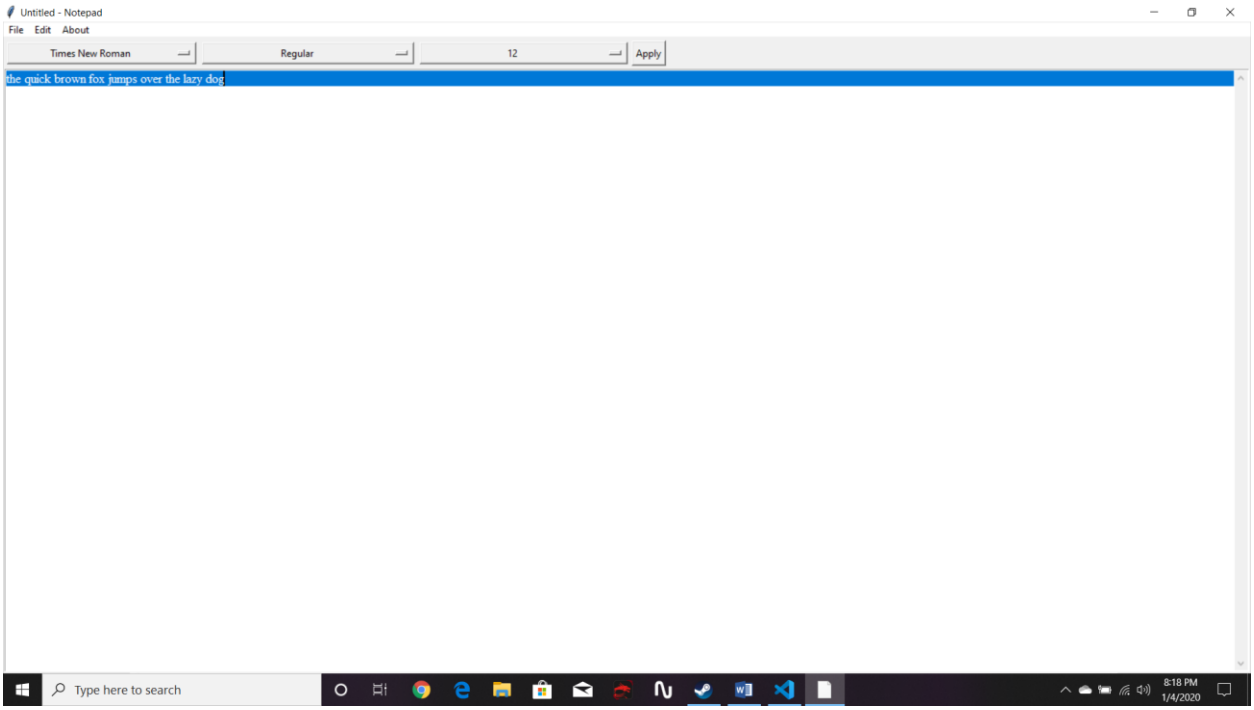




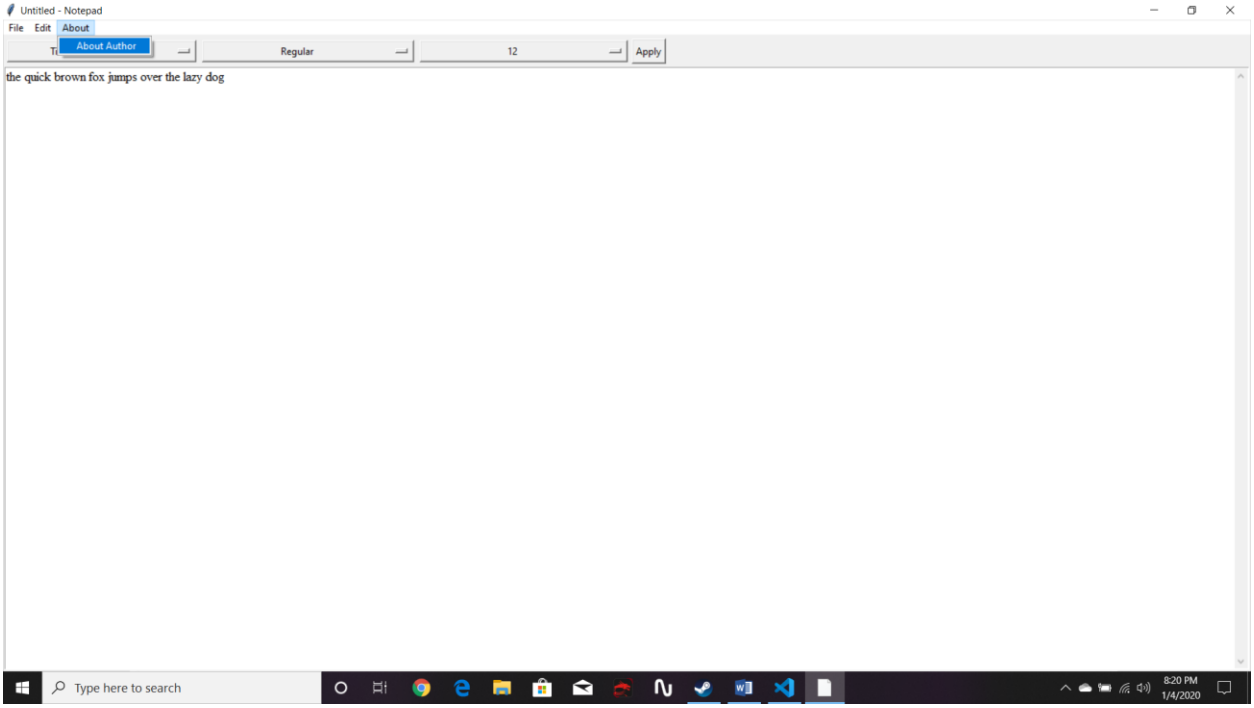
Screenshot of selecting all text:

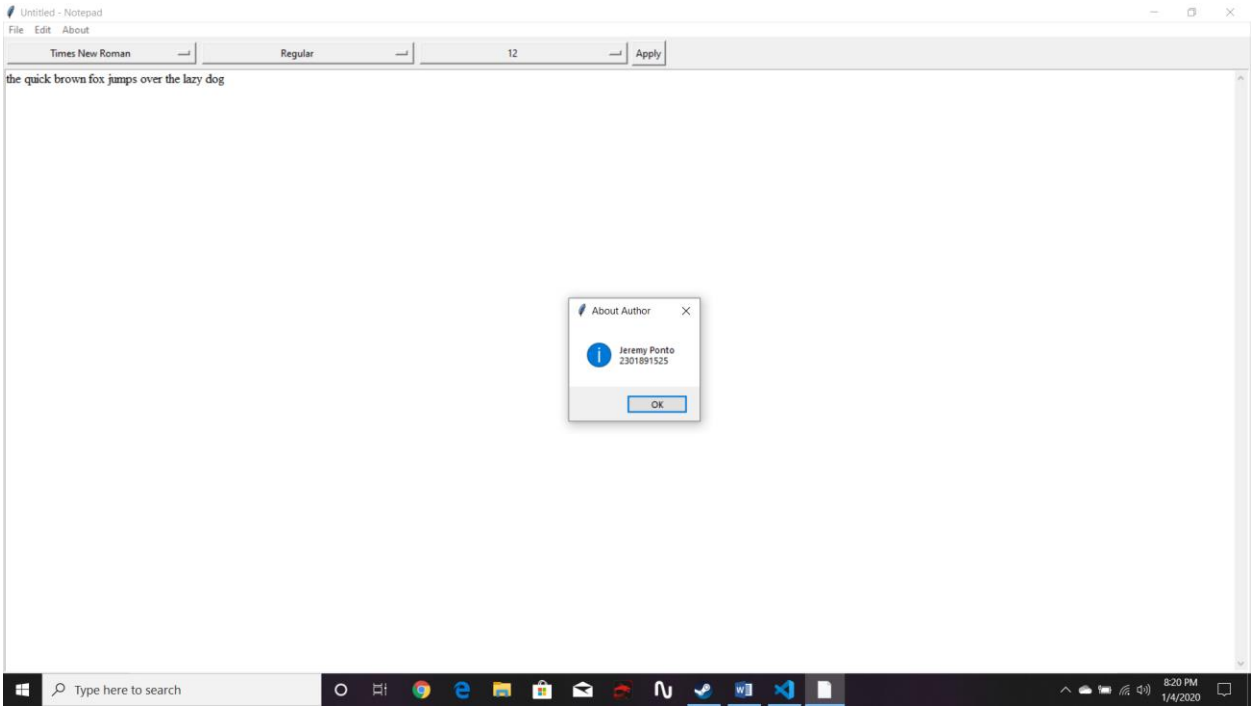



Untitled - Notepad
File Edit About
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Select All Ctrl+A
Regular
12
Apply
over the lazy dog
Type here to search
8:18 PM
1/4/2020


Untitled - Notepad
File Edit About
Times New Roman
Regular
12
Apply
the quick brown fox jumps over the lazy dog
Type here to search
8:18 PM
1/4/2020

Screenshot of seeing author's information:





REFERENCES

https://www.geeksforgeeks.org/make-notepad-using-tkinter/
https://stackoverflow.com/questions/3169344/undo-and-redo-features-in-a-tkinter-text-widget
https://www.youtube.com/watch?v=WZgJPOkTDnA
https://tkdocs.com/tutorial/fonts.html
https://www.youtube.com/watch?v=lysPSwjlvKI
https://stackoverflow.com/questions/46831855/getting-a-value-from-tkinter-optionmenu-in-python
https://www.tutorialspoint.com/python/tk_panedwindow.htm