CMPSCI 119
LAB #4 – Scroller
Professor William T. Verts

The goal of this Python programming assignment is to explore the graphical use of text, and to create a movie from that code. The next assignment will depend on the correct implementation of this assignment. In the JES environment, type in the program code from the last page of this document ***exactly as listed*** (or download a copy from the class site).

Wherever you see my name underlined in the comment code, replace it with *your own name* (two places). As always, be very careful about indentation and capitalization.

In JES click the Load Program button. At the >>> prompt, type `Main()` with the parentheses and press Enter↵. The program should run, but should not do anything visible at this time. Fix any syntax errors or other mistakes. We will not change `Main` while developing the program, but we will fill in code for the `Scroller` and `SaveFile` functions. (The `Sunrise` function will be filled in next time.)

The First Goal – Scrolling Text

The first goal is to create a scrolling line of "movie credits" that go from off the bottom of the screen to off the top. We must use two new JES functions: `addTextWithStyle` and `makeStyle`. The `addTextWithStyle` function uses a bunch of parameters, and is laid out as follows:

```
addTextWithStyle(Canvas, X, Y, Text, Style, Color)
```

The `X` and `Y` values represent the lower-left corner of where the string `Text` will be plotted on the `Canvas` using the value defined here as `Color`. Text will be plotted just above and to the right of coordinates <X,Y>. The `Style` parameter controls how the text should look on screen, and can be defined by the `makeStyle` function as follows:

```
Style = makeStyle(Typeface, Emphasis, Size)
```

The `Typeface` parameter is a string that can be values such as `"serif"` or `"sansSerif"`. `Emphasis` can be `plain`, `bold`, `italic`, or the sum `bold+italic`. `Size` is measured in points (1/72 of an inch), so size `24` represents characters that are ⅓ of an inch tall. You may use any reasonable settings you like for `Typeface` and `Emphasis`, but please restrict `Size` to 24.

Your `Scroller` function has the following arguments:

```
def Scroller (Canvas,Frames,NewText,BackgroundColor,TextColor):
```

`Canvas` is the image to draw on, as usual. `Frames` is an integer that tells `Scroller` how many independent frames to generate (each with the text in a slightly different position). `NewText` is a string that contains the text to plot on screen. `BackgroundColor` is the base color of the canvas. `TextColor` is the color of the text.

Therefore, if we call `Scroller` from within `Main` as follows:

```
Scroller (Canvas, 100, Message, blue, yellow)
```

then `Frames` will be 100, `NewText` will contain the string value passed in through `Message`, `BackgroundColor` will be `blue`, and `TextColor` will be `yellow`. You are **not** to use constants (such as `100`, `"quoted strings"`, `blue`, or `yellow`, for example) inside `Scroller`!

`Scroller` must loop for exactly `Frames` times (the value passed in through the parameter list), plotting the text on the canvas from well off the bottom to just off the top, pausing 0.02 seconds per frame. The text must be invisible at the start (completely below the bottom of the screen) and must be invisible at the end (completely above the top of the screen).

Start the Y coordinate of the text at the height of the canvas plus 20 pixels to give enough padding for the text to be completely invisible at the start, and continue until Y is -20. Make the X coordinate of the text equal to 10. .For each frame you must clear the canvas to the background color, compute where to plot the text, plot it, repaint the canvas, and then pause for 0.02 seconds.

You must compute the change in the Y coordinate per frame based on the height of the canvas plus padding above and below the canvas, and the number of frames passed in through `Frames`.

NOTE: Keep both Y and the change in Y as **floats** (not integers) in order to update the position correctly, and convert the value of Y to an integer only in the call to `addTextWithStyle`.

In `Main()`, you are allowed to change the values that will be passed in to `BackgroundColor` and `TextColor`, shown in the call to `Scroller` as `blue` for the background and `yellow` for the text, but you may **not** use explicit color constants for these values within `Scroller` itself. Note that the `chr(169)` function call in the text string assigned to `Message` will generate a copyright © symbol.

For Extra Credit

In the current instructions, `NewText` is a single string that scrolls up the screen. For 50% extra credit, write Scroller so that `NewText` is a ***list of strings***, all of which must be scrolled up the screen together as a block. You'll need to worry about spacing between lines (For 24-point text I suggest 25 pixels as the line separation). Based on the line spacing and the total number of lines in `NewText` you'll also need to adjust the amount of padding at the top of the screen so that all lines scroll off the top by the last frame of the animation.

To test this, make `NewText` a list containing your name and other related information, such as:

```
["Copyright "+chr(169)+"2017",
 "Dr. William T. Verts",
 "All rights reserved"]
```

When the program works properly, these three lines will scroll together up the screen. Your program should work with a list containing an arbitrary number of strings.

The Second Goal - Saving Each Frame as an Image

The `SaveFile` function is incomplete, but it contains comments that describe what should go there. ***Replace those comments with working code*** to perform the described actions. The intent is that the first time `SaveFile` is called it will prompt you for a folder in which to store saved images, then save the current canvas in that folder as `SAVE00000.jpg` (using the `writePictureTo` function). After that, each call to `SaveFile` saves the canvas to `SAVE00001.jpg`, `SAVE00002.jpg`, `SAVE00003.jpg`, and so on. In order to generate the digits in the file name, you must first create a string variable to hold the current value of `SequenceNumber`, then pad that string on the left with **0** characters until it is exactly five characters in length (think about loops here).

Adding the variables called `SequenceNumber` and `BaseFolder` *outside* the function but declaring them as global variables *inside* the function makes their values persistent from call to call. In other words, when `SaveFile` adds one to `SequenceNumber` that new value will be available the next time `SaveFile` is called.

Add a call to `SaveFile` at the appropriate place inside `Scroller` to create a sequence of frames, create a special folder to hold those frames, then run the program to save the 100 `.jpg` files there.

The Third Goal – Making a Movie

The function `MakeMovie` uses JES functions to stitch the individual frame files together into a single `.mov` (Apple QuickTime) movie. After the `Main` function has been used to call `Scroller` to generate movie frames, run the `MakeMovie` function. Once the `.mov` file has been created (it will be quite large), it may be played in any QuickTime player, independent of JES. Test this and make certain that you can successfully create a 100-frame QuickTime movie.

Finishing Up

That is enough information for you to figure out how to fill out the `SaveFile` and `Scroller` functions and run the program. Try changing the size of the screen and the number of frames passed to `Scroller`.

When you are finished and everything runs correctly, submit the assignment through the on-line form as Lab #4.

We will grade this assignment based on (a) whether or not the text is properly scrolled up the screen, (b) whether or not the individual frame files are saved with the correct names, and (c) whether or not we can create a QuickTime movie from the result.

```
# William T. Verts - Lab #4 - Scroller

SequenceNumber = 0
BaseFolder = ""

def SaveFile (Canvas):
    global SequenceNumber,BaseFolder
    if (BaseFolder == ""): BaseFolder = pickAFolder()

    # Create a new string from SequenceNumber, with enough
    #     leading 0s to make it 5 characters in length.
    # Create a new file name from:
    #    BaseFolder
    #    "SAVE"
    #    Your 5-digit string based on SequenceNumber
    #    ".jpg"
    # Save the Canvas to that new file name.
    # Add 1 to SequenceNumber.

    return

def Scroller (Canvas,Frames,NewText,BackgroundColor,TextColor):

    # Stub to be filled in in this assignment (see instructions)

    return

def Sunrise (Canvas,Frames):

    # Stub to be completed in the next assignment

    return

def Main():
    global SequenceNumber, BaseFolder
    SequenceNumber = 0
    BaseFolder = ""
    Message = "Copyright "+chr(169)+"2017 Dr. William T. Verts"
    Canvas = makeEmptyPicture(640,480)
    Sunrise (Canvas, 100)
    Scroller (Canvas, 100, Message, blue, yellow)
    return

def MakeMovie():
    global BaseFolder
    if (BaseFolder == ""): BaseFolder = pickAFolder()
    MyMovie = makeMovieFromInitialFile(BaseFolder + "SAVE00000.jpg")
    writeQuicktime(MyMovie, BaseFolder + "Scroller.mov", 16)
    return
```