HW3: Cassandra Cabrera and Mike Menendez **Due:** March 9, 2020

Summary:

Overall this homework was a lot of fun to do. Applying the gui knowledge we have learned so far was cool. Searching with the key terms seemed a bit tricky at first, but was easy enough to understand once we figured it out and we were able to make sure it would not be case sensitive. The trouble we had was applying the sepia filter because we had to test around with different values until we found the right shade.

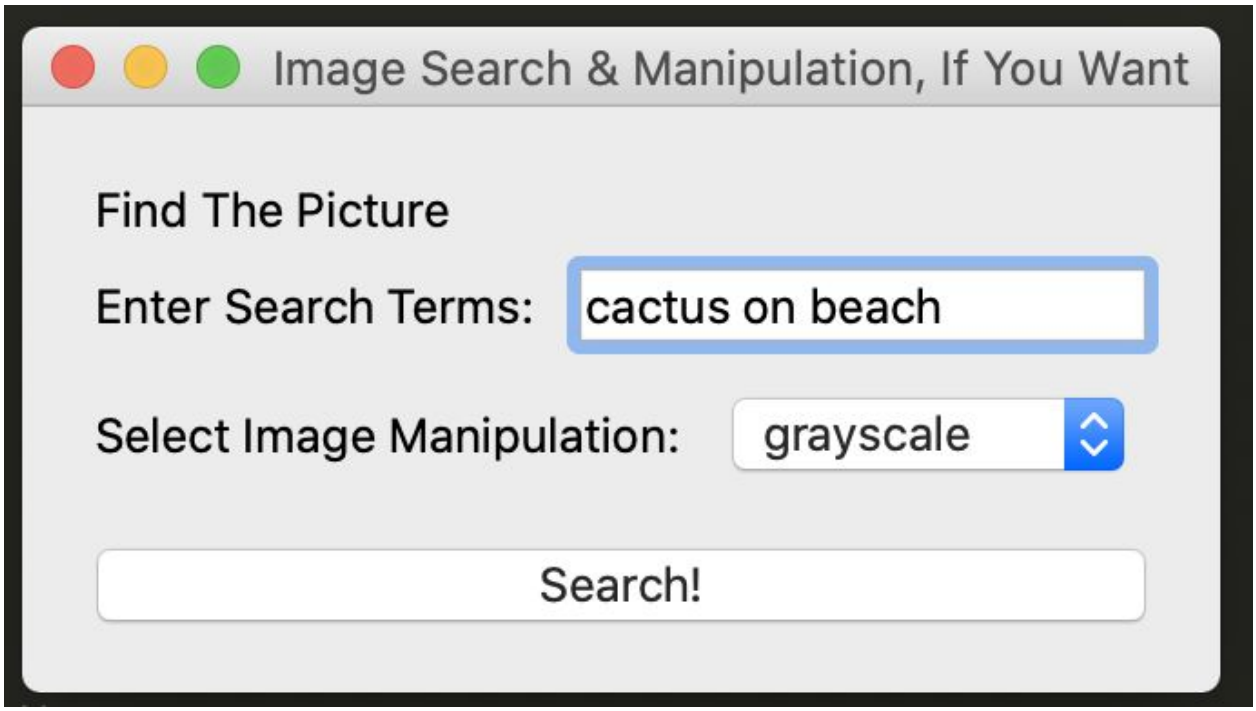
Code:

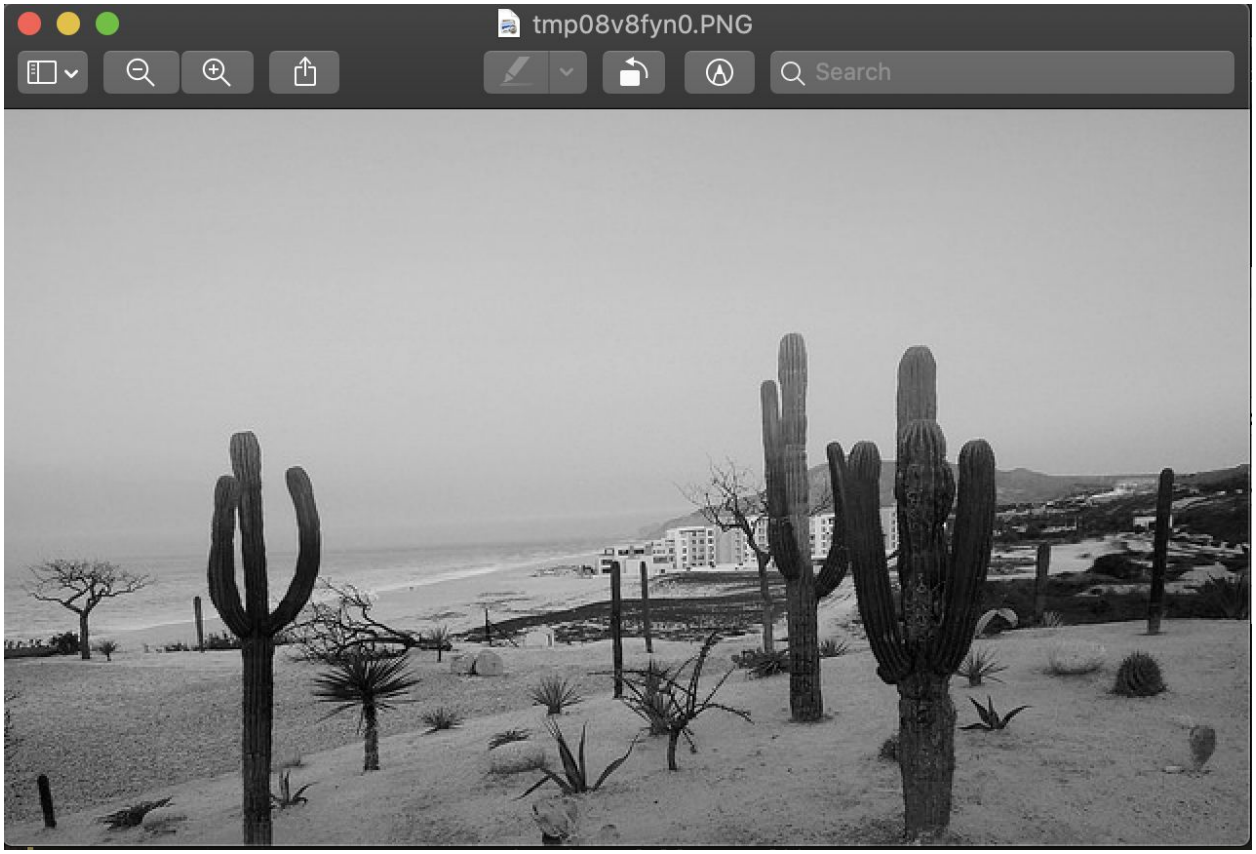
```
'''
Authors: Cassandra Cabrera & Mike Menendez
Date: March 9, 2020
Purpose: Homework 3 for CST205
    - allow user to enter search terms
    - allow user to select type of filter to apply to image
    - display image with filter to user
'''

import sys
import glob
from dict import image_info as info

from PIL import Image, ImageOps
from PyQt5.QtWidgets import (QApplication, QWidget, QLabel, QPushButton,
                             QLineEdit, QHBoxLayout, QVBoxLayout, QComboBox)
from PyQt5.QtCore import pyqtSlot, Qt
from PyQt5.QtGui import QPainter, QColor, QPen
from PyQt5.QtGui import QIcon

manipulations = ['please select','sepia','negative','grayscale','thumbnail','none']

#sets up the main window
class Search(QWidget):
    #initialize the search gui
    def __init__(self):
        super().__init__()

        self.line_edit = QLineEdit(self)
        self.combo_box = QComboBox()
        self.combo_box.addItems(manipulations)
        self.btn = QPushButton("Search!", self)
        self.title = QLabel("Find The Picture")
        self.search = QLabel("Enter Search Terms: ")
        self.change = QLabel("Select Image Manipulation: ")

        self.btn.clicked.connect(self.on_click)
        vbox = QVBoxLayout()
        hbox1 = QHBoxLayout()
        hbox2 = QHBoxLayout()

        hbox1.addWidget(self.search)
        hbox1.addWidget(self.line_edit)

        hbox2.addWidget(self.change)
        hbox2.addWidget(self.combo_box)

        vbox.addWidget(self.title)
        vbox.addLayout(hbox1)
        vbox.addLayout(hbox2)
        vbox.addWidget(self.btn)

        self.setLayout(vbox)

        self.setWindowTitle("Image Search & Manipulation, If You Want")

    #helper for calculating the sepia filter
    def sepia_list(self,p):
        if p[0] < 63:
            r,g,b = int(p[0] * 1.5), int(p[1]), int(p[2] * .6)
```

```
            r,g,b = int(p[0] * 1.5), int(p[1]), int(p[2] * .6)
        if p[0] > 62 and p[0] < 192:
            r,g,b = int(p[0] * 1.75), int(p[1]), int(p[2] * 0.40)
        else:
            r = int(p[0] * 1.08)
            if r > 255:
                r = 255
            g,b = p[1], int(p[2] * 0.5)
        return (r, g, b)

    #sets sepia filter over image
    def sepia(self,img):
        lst = [self.sepia_list(p) for p in img.getdata()]
        img.putdata(lst)
        img.show()
        return

    #sets negative filter over image
    def negative(self,img):
        ImageOps.invert(img).show()
        return

    #sets grayscale filter over image
    def grayscale(self,img):
        new_list = map(lambda a : (int((a[0]+a[1]+a[2])/3),) * 3, img.getdata())
        img.putdata(list(new_list))
        img.show()
        return

    #creates thumbnail of image
    def thumbnail(self,img):
        img.thumbnail((100, 100))
        img.show()
        return


    #the on click listener for the button
    @pyqtSlot()
    def on_click(self):
        terms = (self.line_edit.text()).split()

        #checks for key words
        maxIndex = 0
        maxFound = 0
        exists = False
        for f in range(len(info)):
            found = 0
            for i in range(len(info[f]["tags"])):
                for t in terms:
                    if (info[f]["tags"][i]).lower() == t.lower():
                        found += 1
                        exists = True
            if maxFound < found:
                maxFound += 1
                maxIndex = f

        #check for existing keywords
        if(not exists):
            self.btn.setText("No Matches")
            return
```

```
        if(not exists):
            self.btn.setText("No Matches")
            return

        self.btn.setText("Search")
        img = Image.open("images/"+info[maxIndex]["id"]+".jpg")

        if(self.combo_box.currentText() == "sepia"):
            self.sepia(img)
        elif(self.combo_box.currentText() == "negative"):
            self.negative(img)
        elif(self.combo_box.currentText() == "grayscale"):
            self.grayscale(img)
        elif(self.combo_box.currentText() == "thumbnail"):
            self.thumbnail(img)
        else:
            img.show()


app = QApplication(sys.argv)
main = Search()
main.show()
sys.exit(app.exec_())
```

Results of a negative example:




tmp08v8fyn0.PNG
Search