```python
# coding: utf-8

'''
This is a fairly advanced example of using the `objc_util` module to access CoreImage APIs.
It uses CIDetector to detect (possibly skewed) rectangles in a photo, straighten them using a perspective correction filter, and apply contrast enhancement.

You can use this script to convert photos of receipts etc. to a "scanned" page, more suitable for archiving.
'''

import photos
import console
from objc_util import *

CIFilter, CIImage, CIContext, CIDetector, CIVector = map(ObjCClass, ['CIFilter', 'CIImage', 'CIContext', 'CIDetector', 'CIVector'])

def take_photo(filename='.temp.jpg'):
  img = photos.capture_image()
  if img:
    img.save(filename)
    return filename

def pick_photo(filename='.temp.jpg'):
  img = photos.pick_image()
  if img:
    img.save(filename)
    return filename

def load_ci_image(img_filename):
  data = NSData.dataWithContentsOfFile_(img_filename)
  if not data:
    raise IOError('Could not read file')
  ci_img = CIImage.imageWithData_(data)
  return ci_img

def find_corners(ci_img):
  d = CIDetector.detectorOfType_context_options_('CIDetectorTypeRectangle', None, None)
  rects = d.featuresInImage_(ci_img)
  if rects.count() == 0:
    return None
  r = rects.firstObject()
  return (r.topRight(), r.bottomRight(), r.topLeft(), r.bottomLeft())

def apply_perspective(corners, ci_img):
  tr, br, tl, bl = [CIVector.vectorWithX_Y_(c.x, c.y) for c in corners]
  filter = CIFilter.filterWithName_('CIPerspectiveCorrection')
  filter.setDefaults()
  filter.setValue_forKey_(ci_img, 'inputImage')
  filter.setValue_forKey_(tr, 'inputTopRight')
  filter.setValue_forKey_(tl, 'inputTopLeft')
  filter.setValue_forKey_(br, 'inputBottomRight')
  filter.setValue_forKey_(bl, 'inputBottomLeft')
  out_img = filter.valueForKey_('outputImage')
  return out_img

def enhance_contrast(ci_img):
  filter = CIFilter.filterWithName_('CIColorControls')
  filter.setDefaults()
  filter.setValue_forKey_(2.0, 'inputContrast')
  filter.setValue_forKey_(0.0, 'inputSaturation')
  filter.setValue_forKey_(ci_img, 'inputImage')
  ci_img = filter.valueForKey_('outputImage')
  filter = CIFilter.filterWithName_('CIHighlightShadowAdjust')
  filter.setDefaults()
  filter.setValue_forKey_(1.0, 'inputShadowAmount')
  filter.setValue_forKey_(1.0, 'inputHighlightAmount')
  filter.setValue_forKey_(ci_img, 'inputImage')
  ci_img = filter.valueForKey_('outputImage')
  return ci_img

def write_output(out_ci_img, filename='.output.jpg'):
  ctx = CIContext.contextWithOptions_(None)
  cg_img = ctx.createCGImage_fromRect_(out_ci_img, out_ci_img.extent())
  ui_img = UIImage.imageWithCGImage_(cg_img)
  c.CGImageRelease.argtypes = [c_void_p]
  c.CGImageRelease.restype = None
  c.CGImageRelease(cg_img)
  c.UIImageJPEGRepresentation.argtypes = [c_void_p, CGFloat]
  c.UIImageJPEGRepresentation.restype = c_void_p
  data = ObjCInstance(c.UIImageJPEGRepresentation(ui_img.ptr, 0.75))
  data.writeToFile_atomically_(filename, True)
  return filename

def main():
  console.clear()
  i = console.alert('Info', 'This script detects a printed page (e.g. a receipt) in a photo, and applies perspective correction and contrast enhancement filters automatically.\n\nThe result is a "scanned" black&white image that you can save to your camera roll.\n\nFor best results, make sure that the page is evenly lit.', 'Take Photo', 'Pick from Library')
  if i == 1:
    filename = take_photo()
  else:
    filename = pick_photo()
  if not filename:
    return
  ci_img = load_ci_image(filename)
  corners = find_corners(ci_img)
  if not corners:
    print('Error: Could not find a rectangle in the photo. Please try again with a different image.')
    return
  out_img = apply_perspective(corners, ci_img)
  out_img = enhance_contrast(out_img)
  out_file = write_output(out_img)
  console.show_image(out_file)
  print('Tap and hold the image to save it to your camera roll.')

if __name__ == '__main__':
  main()

```