Generated by Cython 0.29.32

Yellow lines hint at Python interaction.
Click on a line that starts with a "+" to see the C code that Cython generated for it.

Raw output: cython_filters.c

```
+01: """Cython implementation of filter functions"""
 02: 
+03: import numpy as np
 04: cimport numpy as np
 05: 
 06: """  # IGNORE
 07: # BEGIN PROMPT
 08: def cython_color2gray(image):
 09: # END PROMPT
 10: """  # IGNORE
 11: # BEGIN SOLUTION NO PROMPT
+12: def cython_color2gray(np.ndarray[np.uint8_t, ndim=3] image):
 13: # END SOLUTION
 14:     """Convert rgb pixel array to grayscale
 15: 
 16:     Args:
 17:         image (np.array)
 18:     Returns:
 19:         np.array: gray_image
 20:     """
 21:     # BEGIN SOLUTION
+22:     cdef int height=image.shape[0], width=image.shape[1]
+23:     cdef np.ndarray[np.float64_t, ndim=3] gray_image = np.empty(shape=(height, width, 3), dtype=np.float64)
 24:     cdef int i, j
 25: 
+26:     for i in range(height):
+27:         for j in range(width):
+28:             gray_image[i,j,0] =  gray_image[i,j,1] = gray_image[i,j,2] = 0.21 * image[i,j,0] + 0.72 * image[i,j,1] +  0.07 * image[i,j,2]
+29:     return gray_image.astype("uint8")
 30:     # END SOLUTION
 31: 
 32: """  # IGNORE
 33: # BEGIN PROMPT
 34: def cython_color2sepia(image):
 35: # END PROMPT
 36: """  # IGNORE
 37: # BEGIN SOLUTION NO PROMPT
+38: def cython_color2sepia(np.ndarray[np.uint8_t, ndim=3] image):
 39: # END SOLUTION
 40:     """Convert rgb pixel array to sepia
 41: 
 42:     Args:
 43:         image (np.array)
 44:     Returns:
 45:         np.array: gray_image
 46:     """
 47:     # BEGIN SOLUTION
+48:     cdef np.ndarray[np.uint8_t, ndim=3] sepia_image = np.empty_like(image)
+49:     cdef int height=image.shape[0], width=image.shape[1]
 50:     cdef int i, j
 51:     cdef float r, g, b
 52: 
+53:     for i in range(height):
+54:         for j in range(width):
+55:             r = image[i,j,0] * 0.393 + image[i,j,1] * 0.769 + image[i,j,2] * 0.189
+56:             g = image[i,j,0] * 0.349 + image[i,j,1] * 0.686 + image[i,j,2] * 0.168
+57:             b = image[i,j,0] * 0.272 + image[i,j,1] * 0.534 + image[i,j,2] * 0.131
+58:             if r > 255:
+59:                 sepia_image[i,j,0] = 255
 60:             else:
+61:                 sepia_image[i,j,0] = int(r)
+62:             if g > 255:
+63:                 sepia_image[i,j,1] = 255
 64:             else:
+65:                 sepia_image[i,j,1] = int(g)
+66:             if b > 255:
+67:                 sepia_image[i,j,2] = 255
 68:             else:
+69:                 sepia_image[i,j,2] = int(b)
+70:     return sepia_image
 71:     # END SOLUTION
```