In [1]:

```
import numpy as np
```

In [2]:

```
A = np.arange(4).reshape(2,2)
A
```

Out[2]:

```
array([[0, 1],
       [2, 3]])
```

In [3]:

```
B = np.arange(10, 16).reshape(2,3)
B
```

Out[3]:

```
array([[10, 11, 12],
       [13, 14, 15]])
```

In [5]:

```
np.hstack((A, B))
```

Out[5]:

```
array([[ 0,  1, 10, 11, 12],
       [ 2,  3, 13, 14, 15]])
```

In [10]:

```
np.vstack((A, B.T))
```

Out[10]:

```
array([[ 0,  1],
       [ 2,  3],
       [10, 13],
       [11, 14],
       [12, 15]])
```

In [ ]:

```

```

In [ ]:

```

```

What will be the output of following code?

```
a = np.array([[1], [2], [3]])
b = np.array([[4], [5], [6]])
np.vstack((a, b))
```

A.
```
array([1, 2, 3, 4, 5, 6])
```

B.
```
array([[1, 4],
       [2, 5],
       [3, 6]])
```

C.
```
array([[1],
       [2],
       [3],
       [4],
       [5],
       [6]])
```

D. Error

In [11]:

```
a = np.array([[1], [2], [3]])
a
```

Out[11]:

```
array([[1],
       [2],
       [3]])
```

In [12]:

```
b = np.array([[4], [5], [6]])
b
```

Out[12]:

```
array([[4],
       [5],
       [6]])
```

In [14]:

```
c = np.vstack((a, b))
c
```

Out[14]:

```
array([[1],
       [2],
       [3],
       [4],
       [5],
       [6]])
```

In [15]:

```
c.shape
```

Out[15]:

```
(6, 1)
```

In [18]:

```
c.flatten()
```

Out[18]:

```
array([1, 2, 3, 4, 5, 6])
```

In [ ]:

```
# np.concatenate()
```

In [24]:

```
a = np.array([2, 30, 41, 7, 17, 52])
a
```

Out[24]:

```
array([ 2, 30, 41,  7, 17, 52])
```

In [28]:

```
np.sort(a)
```

Out[28]:

```
array([ 2,  7, 17, 30, 41, 52])
```

In [33]:

```
# descending order

# np.flip()

# [::-1]
```

In [38]:

```
-np.sort(-a)
```

Out[38]:

```
array([52, 41, 30, 17,  7,  2])
```

In [22]:

```
np.ndarray.sort(a)
```

In [23]:

```
a
```

Out[23]:

```
array([ 2,  7, 17, 30, 41, 52])
```

Argsort

In [29]:

```
a = np.array([2, 30, 41, 7, 17, 52])
a
```

Out[29]:

```
array([ 2, 30, 41,  7, 17, 52])
```

In [30]:

```
np.sort(a)
```

Out[30]:

```
array([ 2,  7, 17, 30, 41, 52])
```

In [31]:

```
np.argsort(a)
```

Out[31]:

```
array([0, 3, 4, 1, 2, 5])
```

In [ ]:

```

```

Broadcasting

In [39]:

```
A = np.arange(12).reshape(3, 4)
A
```

Out[39]:

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [50]:

```
v = np.array([1,2,3])
v
```

Out[50]:

```
array([1, 2, 3])
```

In [51]:

```
A*v
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call
 last)
<ipython-input-51-9bd99444c218> in <module>
----> 1 A*v

ValueError: operands could not be broadcast together with shapes (3,4)
(3,)
```

In [52]:

```
v*A
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call
 last)
<ipython-input-52-cb1e8bf78ac1> in <module>
----> 1 v*A

ValueError: operands could not be broadcast together with shapes (3,)
 (3,4)
```

In [53]:

```
A+v
```

```
---------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
 last)
<ipython-input-53-3a4a475150f8> in <module>
----> 1 A+v

ValueError: operands could not be broadcast together with shapes (3,4)
(3,)
```

In [47]:

```
# np.expand_dims
# np.newaxis
```

In [ ]:

```
# v.reshape(-1,1)
```

In [58]:

```
v = np.array([1,2,3,4])
v
```

Out[58]:

```
array([[1, 2, 3, 4]])
```

In [56]:

```
A
```

Out[56]:

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [57]:

```
A + v
```

Out[57]:

```
array([[ 1,  3,  5,  7],
       [ 5,  7,  9, 11],
       [ 9, 11, 13, 15]])
```

In [ ]:

```
```

In [ ]:

```
```

In [62]:

```
v
```

Out[62]:

```
array([1, 2, 3, 4])
```

In [66]:

```
v.reshape(1, -1)
```

Out[66]:

```
array([[1, 2, 3, 4]])
```

In [64]:

```
v[np.newaxis, :]
```

Out[64]:

```
array([[1, 2, 3, 4]])
```

In [65]:

```
v.reshape(-1, 1)
```

Out[65]:

```
array([[1],
       [2],
       [3],
       [4]])
```

In [63]:

```
v[:, np.newaxis]
```

Out[63]:

```
array([[1],
       [2],
       [3],
       [4]])
```

In [68]:

```
A
```

Out[68]:

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [69]:

```
B = np.array([1,2,3]).reshape(-1, 1)
B
```

Out[69]:

```
array([[1],
       [2],
       [3]])
```

In [70]:

```
A+B
```

Out[70]:

```
array([[ 1,  2,  3,  4],
       [ 6,  7,  8,  9],
       [11, 12, 13, 14]])
```

In [79]:

```
A+ (B.T)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call
 last)
<ipython-input-79-e409f072c4f2> in <module>
----> 1 A+ (B.T)

ValueError: operands could not be broadcast together with shapes (3,4)
(1,3)
```

In [ ]:

```

```

In [80]:

```
A = np.arange(4).reshape(1,4)
A
```

Out[80]:

```
array([[0, 1, 2, 3]])
```

In [81]:

```
A.T
```

Out[81]:

```
array([[0],
       [1],
       [2],
       [3]])
```

In [82]:

```
A
```

Out[82]:

```
array([[0, 1, 2, 3]])
```

In [83]:

```
np.dot(A, A.T)
```

Out[83]:

```
array([[14]])
```

In [84]:

```
np.dot(A.T, A)
```

Out[84]:

```
array([[0, 0, 0, 0],
       [0, 1, 2, 3],
       [0, 2, 4, 6],
       [0, 3, 6, 9]])
```

In [85]:

```
A*A.T
```

Out[85]:

```
array([[0, 0, 0, 0],
       [0, 1, 2, 3],
       [0, 2, 4, 6],
       [0, 3, 6, 9]])
```

In [ ]:

```
```

In [86]:

```
A
```

Out[86]:

```
array([[0, 1, 2, 3]])
```

In [87]:

```
A.T
```

Out[87]:

```
array([[0],
       [1],
       [2],
       [3]])
```

In [ ]:

```
# (1, 4)

# (4,1)
```

In [78]:

```
np.array([[2,2,2,2]])@np.array([[2],[2],[2],[2]])
```

Out[78]:

```
array([[16]])
```

In [77]:

```
np.array([[2],[2],[2],[2]])
```

Out[77]:

```
array([[2],
       [2],
       [2],
       [2]])
```

In [ ]:

```
```

Random Number

In [103]:

```
np.random.randint(1, 10, size=(3,3))
```

Out[103]:

```
array([[8, 6, 2],
       [5, 6, 7],
       [4, 2, 7]])
```

In [107]:

```
np.random.rand(3)
```

Out[107]:

```
array([0.77330055, 0.21983434, 0.36999826])
```

In [129]:

```
# 50-75

25*np.random.rand() + 50
```

Out[129]:

```
59.9842930406216
```

In [ ]:

```

```

In [155]:

```
marks = np.random.normal(loc=70, scale=15, size=(100,1))
```

In [156]:

```
# np.average()
np.mean(marks)
```

Out[156]:

```
70.25329596652699
```

In [157]:

```
np.std(marks)
```

Out[157]:

```
13.994807788954416
```

In [ ]:

```
```

Shallow Copy Vs. Deep copy

In [158]:

```
a = np.arange(4)
a
```

Out[158]:

```
array([0, 1, 2, 3])
```

In [159]:

```
b  = a.reshape(2,2)
b
```

Out[159]:

```
array([[0, 1],
       [2, 3]])
```

In [160]:

```
b[0,0] = 100
b
```

Out[160]:

```
array([[100,   1],
       [  2,   3]])
```

In [161]:

```
a
```

Out[161]:

```
array([100,   1,   2,   3])
```

In [164]:

```
id(b), id(a)
```

Out[164]:

```
(140435922117040, 140435921940080)
```

In [162]:

```
b.base is a
```

Out[162]:

```
True
```

In [163]:

```
b.flags.owndata
```

Out[163]:

```
False
```

In [165]:

```
a.flags.owndata
```

Out[165]:

```
True
```

In [166]:

```
c = a.copy()
```

In [ ]:

```
c.reshape
```

In [171]:

```
c[0] = 200
```

In [172]:

```
c
```

Out[172]:

```
array([200,   1,   2,   3])
```

In [173]:

```
a
```

Out[173]:

```
array([100,   1,   2,   3])
```

In [ ]:

```

```

In [ ]:

```

```

uFunc

In [175]:

```
a = np.array([0,1,2,3])
b = np.arange(4)
```

In [176]:

```
a+b
```

Out[176]:

```
array([0, 2, 4, 6])
```

In [177]:

```
np.add(a,b)
```

Out[177]:

```
array([0, 2, 4, 6])
```

In [178]:

```
np.mean(a)
```

Out[178]:

```
1.5
```

In [179]:

```
np.sum(a)
```

Out[179]:

```
6
```

In [182]:

```
A = np.arange(12).reshape(3,4)
A
```

Out[182]:

```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

In [183]:

```
np.sum(A)
```

Out[183]:

```
66
```

In [184]:

```
np.sum(A, axis=1)
```

Out[184]:

```
array([ 6, 22, 38])
```

In [185]:

```
np.sum(A, axis=0)
```

Out[185]:

```
array([12, 15, 18, 21])
```

In [194]:

```
np.mean(A, axis=1).reshape(-1, 1)
```

Out[194]:

```
array([[1.5],
       [5.5],
       [9.5]])
```

In [191]:

```
np.mean(A, axis=1).reshape(1, -1)
```

Out[191]:

```
array([[1.5, 5.5, 9.5]])
```

In [ ]:

```

```

- np.any()
- np.all()

In [195]:

```
a = np.array([0,1,2,4])
a
```

Out[195]:

```
array([0, 1, 2, 4])
```

In [196]:

```
np.any(a)
```

Out[196]:

```
True
```

In [199]:

```
a = np.array([0,0,0,0])
np.any(a)
```

Out[199]:

```
False
```

In [201]:

```
a = np.array([1,2,3,4])
b = np.array([4,3,2,1])

np.any(a>b)
```

Out[201]:

```
True
```

In [202]:

```
a = np.array([1,2,3,4])
b = np.array([4,3,10,20])

np.any(a>b)
```

Out[202]:

```
False
```

In [203]:

```
a = np.array([1,2,3,4])
b = np.array([4,3,2,1])

np.all(a>b)
```

Out[203]:

```
False
```

In [208]:

```
A = np.random.randint(-1, 2, size=(3, 4))
A
```

Out[208]:

```
array([[ 0,  0,  0,  0],
       [ 1,  0, -1,  0],
       [-1,  0,  0,  0]])
```

In [209]:

```
np.any(A, axis=1)
```

Out[209]:

```
array([False,  True,  True])
```

In [210]:

```
np.any(A, axis=0)
```

Out[210]:

```
array([ True, False,  True, False])
```

In [ ]:

```

```

In [217]:

```
def distance(p1, p2):

#     d = ((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)**0.5

    return np.sum(np.square(p1 - p2))**0.5
```

In [218]:

```
p1 = np.array([3,4]).reshape(-1, 1)
p2 = np.array([7,2]).reshape(-1, 1)
```

In [219]:

```
distance(p1, p2)
```

Out[219]:

```
4.47213595499958
```

In [212]:

```
p1
```

Out[212]:

```
array([[3],
       [4]])
```

In [213]:

```
p2
```

Out[213]:

```
array([[7],
       [2]])
```

In [ ]:

```

```

In [ ]:

```

```

Images-> 3D

In [282]:

```
A = np.random.randint(1, 10, size=(2,3,4))
A.shape
```

Out[282]:

```
(2, 3, 4)
```

In [233]:

```
A
```

Out[233]:

```
array([[[8, 8, 1, 8],
        [3, 8, 5, 9],
        [1, 1, 3, 4]],

       [[7, 6, 7, 7],
        [2, 6, 1, 1],
        [3, 4, 5, 6]]])
```

In [234]:

```
A[0]
```

Out[234]:

```
array([[8, 8, 1, 8],
       [3, 8, 5, 9],
       [1, 1, 3, 4]])
```

In [235]:

```
A[0,0,0]
```

Out[235]:

```
8
```

In [236]:

```
A[1,1,1]
```

Out[236]:

```
6
```

In [ ]:

```

```

In [237]:

```
!pip install matplotlib
```

```
Requirement already satisfied: matplotlib in /Users/mohit/opt/anaconda
3/lib/python3.8/site-packages (3.3.4)
Requirement already satisfied: cycler>=0.10 in /Users/mohit/opt/anacon
da3/lib/python3.8/site-packages (from matplotlib) (0.10.0)
Requirement already satisfied: pillow>=6.2.0 in /Users/mohit/opt/anaco
nda3/lib/python3.8/site-packages (from matplotlib) (8.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /Users/mohit/opt/a
naconda3/lib/python3.8/site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: numpy>=1.15 in /Users/mohit/opt/anacond
a3/lib/python3.8/site-packages (from matplotlib) (1.20.1)
Requirement already satisfied: python-dateutil>=2.1 in /Users/mohit/op
t/anaconda3/lib/python3.8/site-packages (from matplotlib) (2.8.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.
3 in /Users/mohit/opt/anaconda3/lib/python3.8/site-packages (from matp
lotlib) (2.4.7)
Requirement already satisfied: six in /Users/mohit/opt/anaconda3/lib/p
ython3.8/site-packages (from cycler>=0.10->matplotlib) (1.15.0)
```

In [238]:

```
import matplotlib.pyplot as plt
```

In [255]:

```
img = plt.imread("fruits.png")
type(img)
```

Out[255]:

```
numpy.ndarray
```

In [256]:

```
img.shape
```

Out[256]:

```
(1333, 2000, 3)
```

In [257]:

```
img = img*255
```

In [258]:

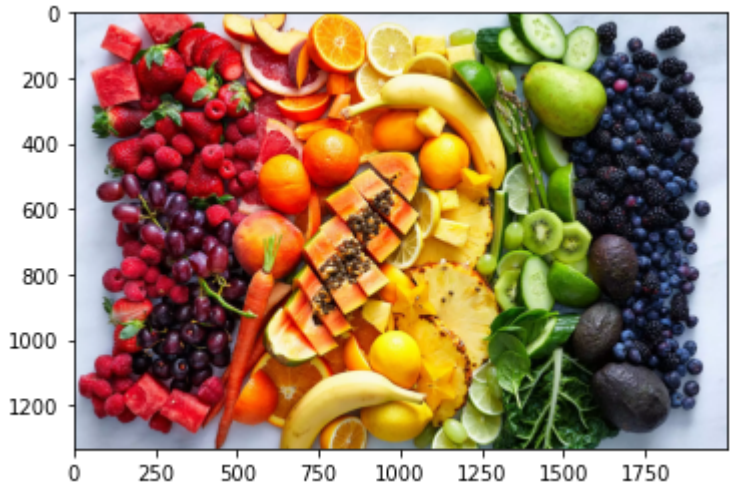
```
img = img.astype('int')
```

In [259]:

```
plt.imshow(img)
```

Out[259]:

```
<matplotlib.image.AxesImage at 0x7fb9c941f8e0>
```



In [267]:

```
img_r = img.copy()
img_g = img.copy()
img_b = img.copy()
```

In [265]:

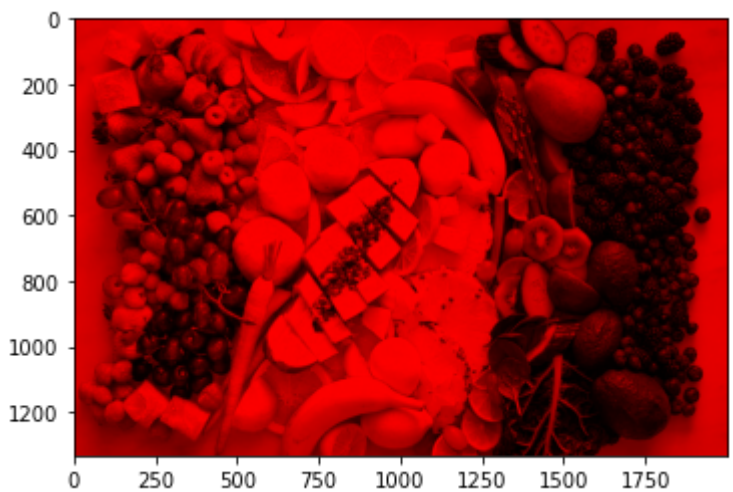
```
# making green and blue channel as 0
img_r[: , : , [1,2]] = 0
```

In [266]:

```
plt.imshow(img_r)
```

Out[266]:

```
<matplotlib.image.AxesImage at 0x7fb9c8a34700>
```



In [268]:

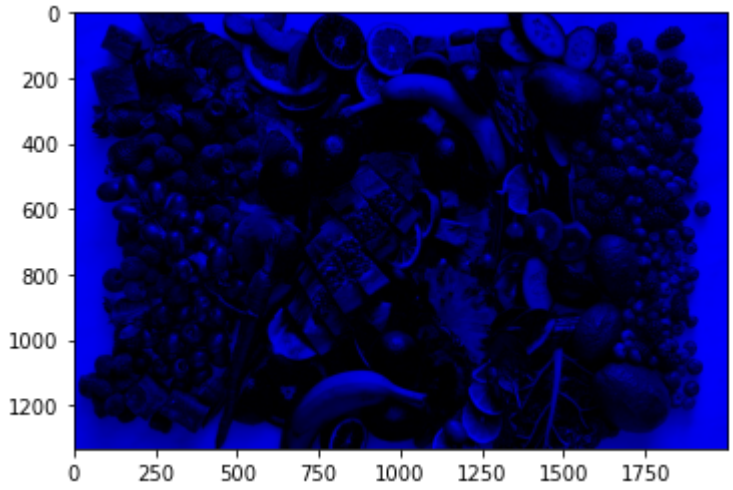
```
# making red and green channel as 0
img_b[: , : , [0,1]] = 0
```

In [269]:

```
plt.imshow(img_b)
```

Out[269]:

```
<matplotlib.image.AxesImage at 0x7fb9780ac310>
```



In [270]:

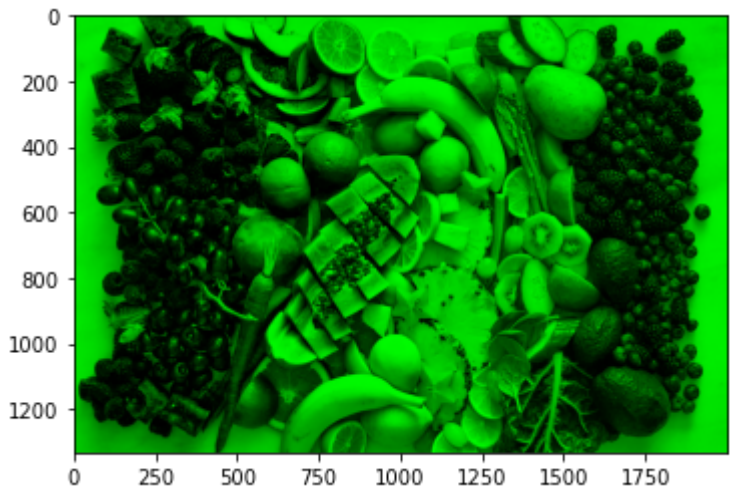
```
# making red and green channel as 0
img_g[: , : , [0,2]] = 0
```

In [271]:

```
plt.imshow(img_g)
```

Out[271]:

```
<matplotlib.image.AxesImage at 0x7fb9b8711ac0>
```



In [ ]:

```
```

In [284]:

```
img.shape
```

Out[284]:

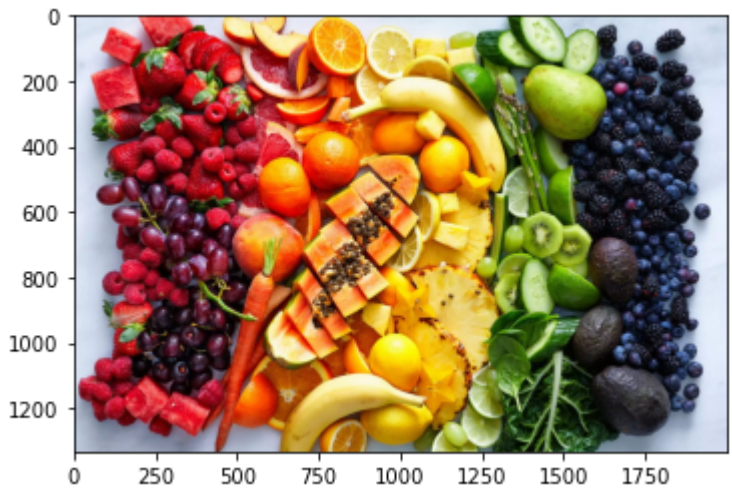
```
(1333, 2000, 3)
```

In [287]:

```
plt.imshow(img)
```

Out[287]:

```
<matplotlib.image.AxesImage at 0x7fb9a037dd30>
```



In [290]:

```
papaya = img[ 400:1100, 510:1100 , : ]
papaya.shape
```

Out[290]:

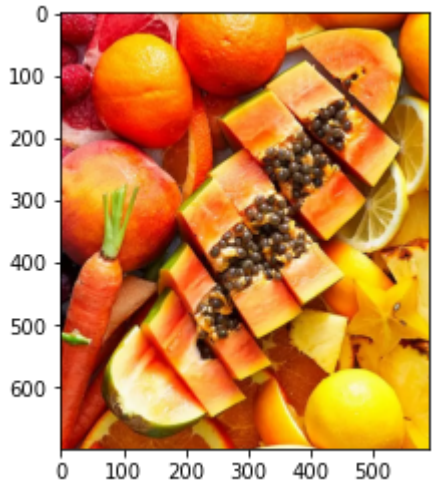
```
(700, 590, 3)
```

In [291]:

```
plt.imshow(papaya)
```

Out[291]:

```
<matplotlib.image.AxesImage at 0x7fb989b2d430>
```



In [ ]:

```

```

In [299]:

```
img.shape
```

Out[299]:

```
(1333, 2000, 3)
```

In [296]:

```
fruit_rotate = np.transpose(img, [1,0,2])
```

In [297]:

```
fruit_rotate.shape
```

Out[297]:

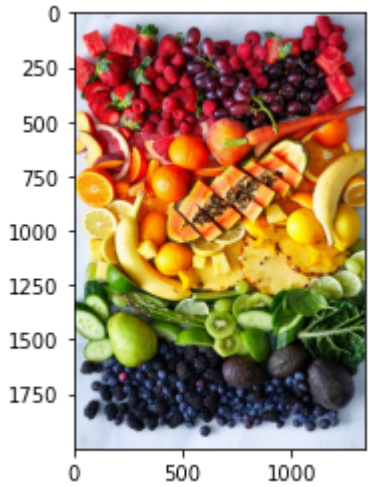
```
(2000, 1333, 3)
```

In [298]:

```
plt.imshow(fruit_rotate)
```

Out[298]:

```
<matplotlib.image.AxesImage at 0x7fb981b3be50>
```



In [ ]:

```

```

In [302]:

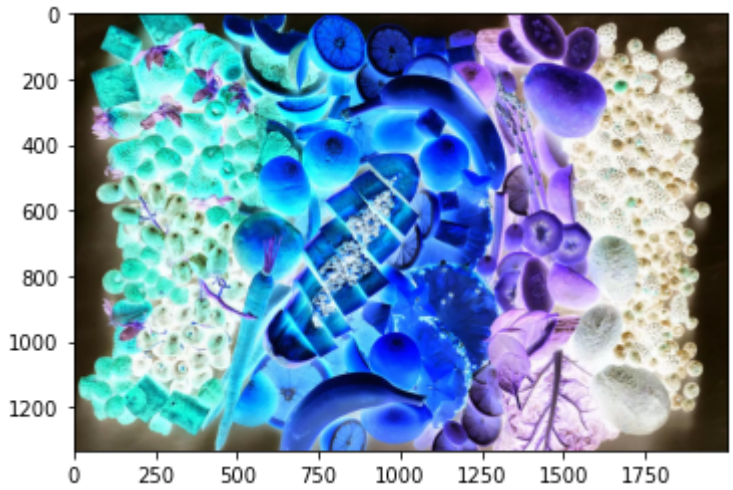
```
fruits_neg =  255- img
```

In [303]:

```
plt.imshow(fruits_neg)
```

Out[303]:

```
<matplotlib.image.AxesImage at 0x7fb98a2e49a0>
```



In [ ]:

```
```

In [316]:

```
black = np.zeros((30,2000, 3), dtype='int')
black.shape
```

Out[316]:

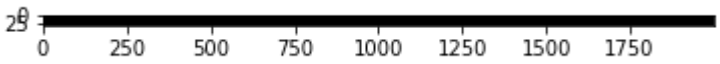
```
(30, 2000, 3)
```

In [317]:

```
plt.imshow(black)
```

Out[317]:

```
<matplotlib.image.AxesImage at 0x7fb981b452b0>
```



In [318]:

```
new_img = np.vstack((black, img, black))
```

In [319]:

```
plt.imshow(new_img)
```

Out[319]:

```
<matplotlib.image.AxesImage at 0x7fb9a03d09d0>
```

In [ ]:

```
```