Contours in openCV

- use Binary Images
- background should be black and objects white
- contours modifies the original image so save a copy first.

In [23]:

```
import numpy as np
import cv2

img = cv2.imread('images/pic1.png')
imgray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
ret,thresh = cv2.threshold(imgray,127,255,0)
contours, hierarchy = cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)

print(len(contours))
```

using drawContours fucntion to draw all the contours

In [22]:

```
import numpy as np
import cv2

img = cv2.imread('images/pic1.png')
imgray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
ret,thresh = cv2.threshold(imgray,127,255,0)
contours, hierarchy = cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)


img = cv2.drawContours(img, contours, -1, (0,255,0), 1)
cv2.imshow("image",img)
cv2.waitKey(5000)
cv2.destroyAllWindows()
```

Contour Approximation methods

- `cv2.CHAIN_APPROX_NONE`
  - Stores all the points in the contour (inefficient)
- `cv2.CHAIN_APPROX_SIMPLE`
  - removes all the redundant points found (efficient)

Contour area

To find the contour area we can use the `cv2.contourArea()` function

Contour Perimeter

To find the contour area we can use the `cv2.arcLength(cnt,True)` function

Contour Approximation

Contour approximation can be used when we are tying to detect a particular shape but can't because of some irruegularities.

We can use the `cv2.approxPolyDP()` fucntion to do the approximation for us The fucntion requires a epsilon argument which has to chosen wisely fot the approximation to work as expected.Usually it is taken as a percentage of the perimeter of the contour

```
epsilon = 0.1*cv2.arcLength(cnt,True)
approx = cv2.approxPolyDP(cnt,epsilon,True)
```

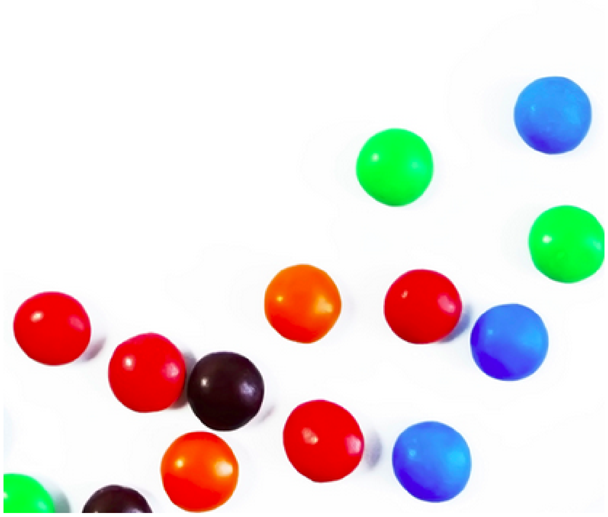
The function returns another contour which will mostly have lesser number of edges. Ideally we will us this approximated contour to draw bounding box.

Bounding Rectangle

We can use the `cv2.boundingRect()` fucntion to draw a bounding box around the detected contour. The fucntion returns the (x,y) coordinates along with the height and the width of the rectangle. Later we can use this information to draw the rectangle.

Home Work

1) calculate the number of circles in this image using contours



In [ ]: