



# WORKSHOP COMPUTER VISION



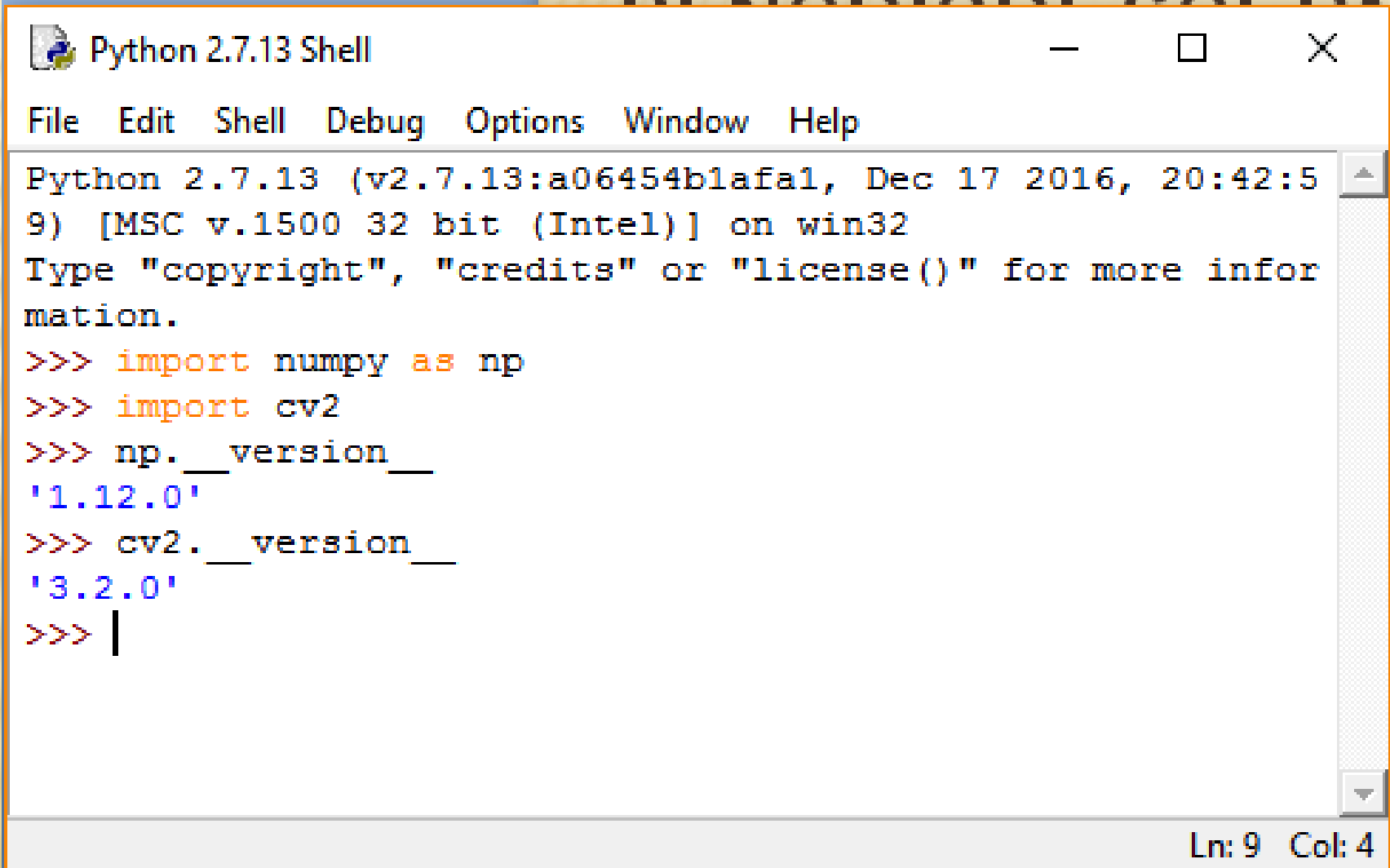
# WORKSHOP COMPUTER VISION

- ✖ 2 dagen
- ✖ 1 ec
- ✖ Opdrachten maken
- ✖ Geen toets, wel controle op deelname
- ✖ Elo: opleiding informatica -> jaar 2 -> computer vision

# BENODIGDE SOFTWARE

- ✗ Python versie 2.7.\*
- ✗ <https://www.python.org/downloads/>
- ✗ Vink aan dat Python in het pad komt
  
- ✗ Voor windows vanuit console:
- ✗ `pip install opencv-python`
  
- ✗ Voor raspberry pi vanuit console :
- ✗ `sudo apt-get install libopencv-dev python-opencv`
  
- ✗ Zie appendix voor verschillen opencv 2.\* en opencv 3.\*

# TEST: IDLE (PYTHON GUI)



The screenshot shows a window titled "Python 2.7.13 Shell" with a standard menu bar (File, Edit, Shell, Debug, Options, Window, Help). The main text area contains the following text and code:

```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

>>> import numpy as np
>>> import cv2
>>> np.__version__
'1.12.0'
>>> cv2.__version__
'3.2.0'
>>> |
```

The status bar at the bottom right indicates "Ln: 9 Col: 4".

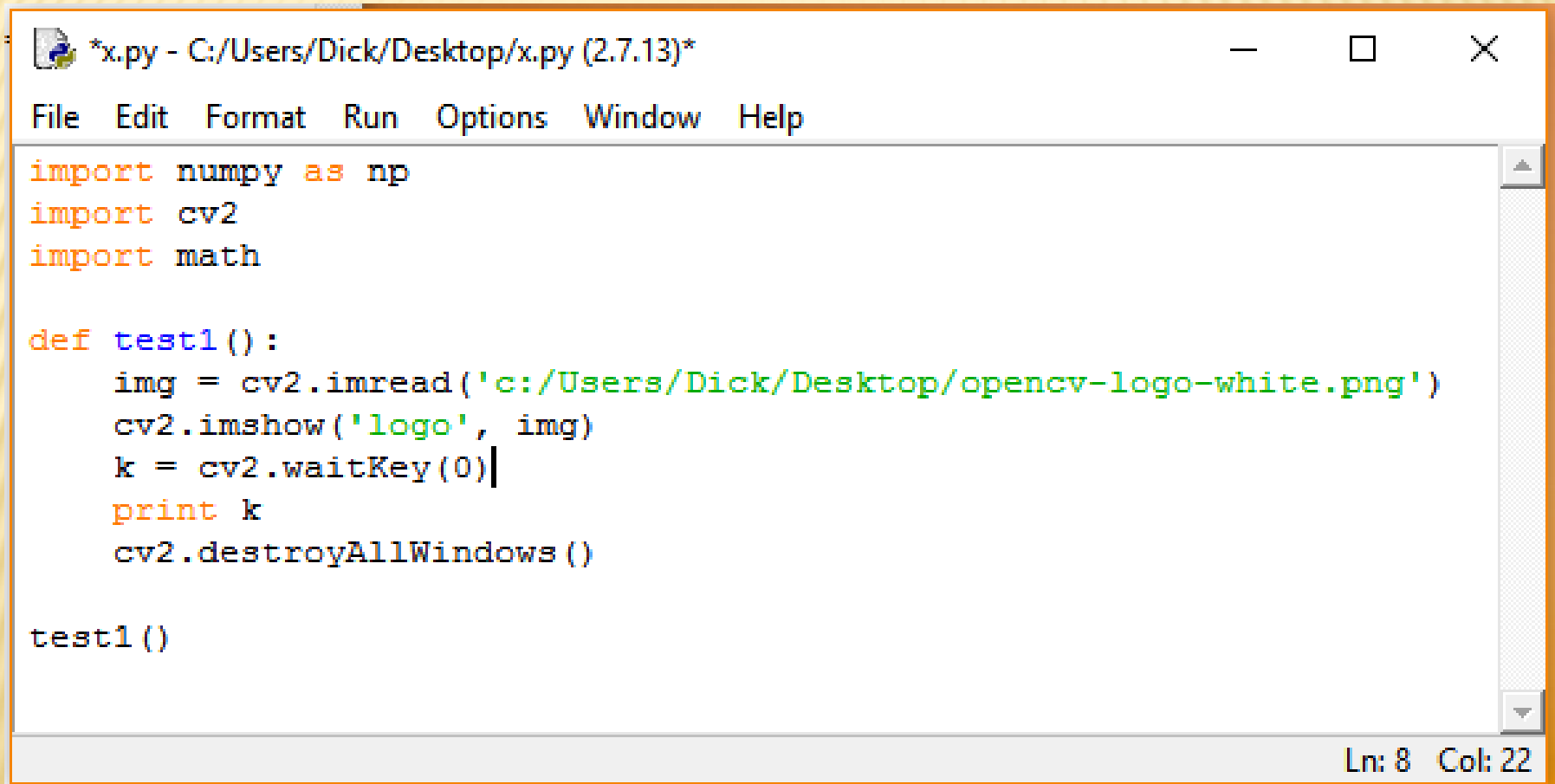


# PYTHON

---

- ✗ Download materiaal van Elo
- ✗ Open voorbeeld.py met Idle
- ✗ Kies vanuit het menu run -> Run Module
- ✗ Na de prompt >>> mag je opdrachten geven
- ✗ >>> `sum(x**2 for x in range(10))`

# LOGO TONEN



A screenshot of a Python IDE window titled "\*x.py - C:/Users/Dick/Desktop/x.py (2.7.13)\*". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The code editor contains the following Python code:

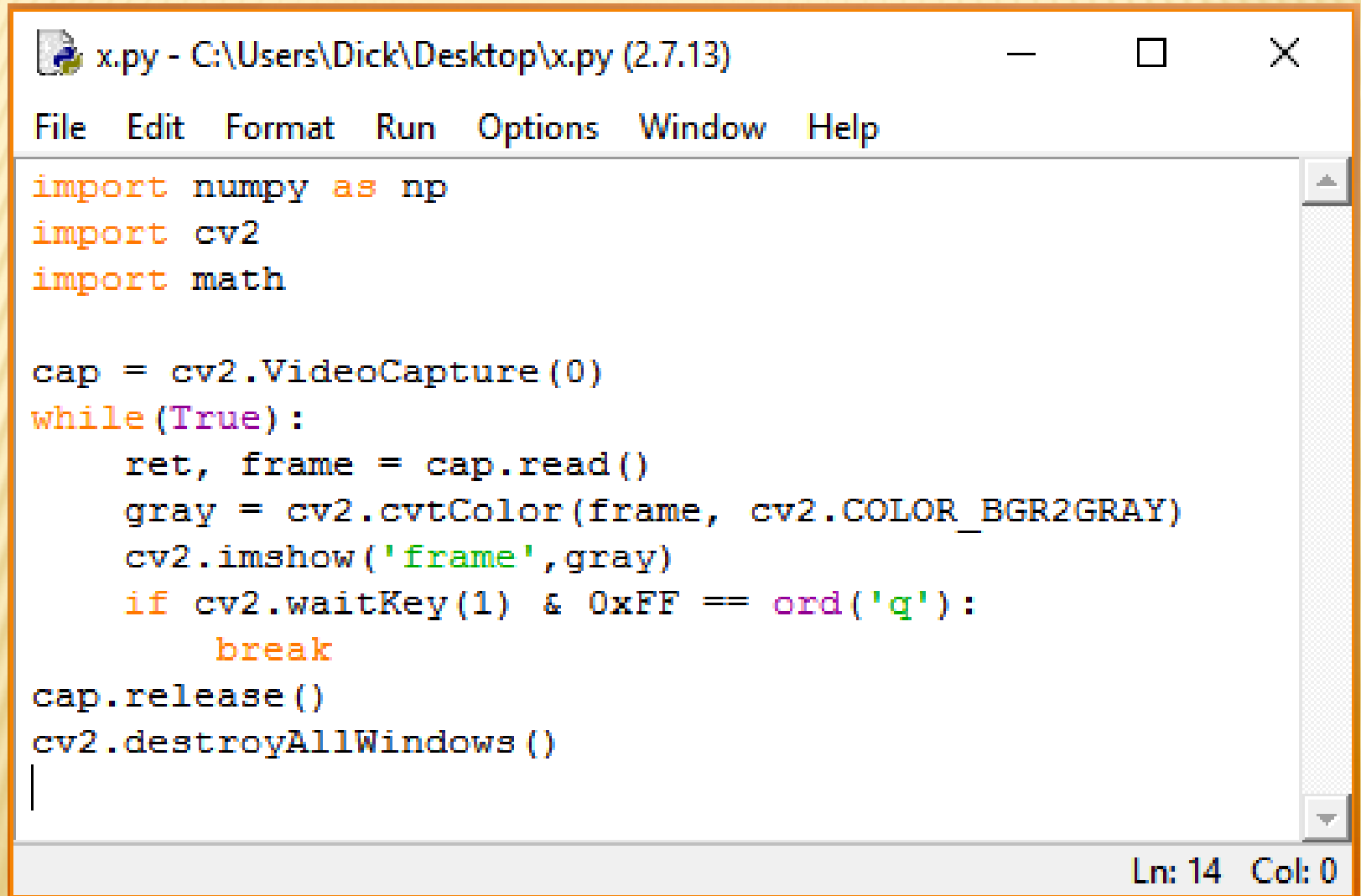
```
import numpy as np
import cv2
import math

def test1():
    img = cv2.imread('c:/Users/Dick/Desktop/opencv-logo-white.png')
    cv2.imshow('logo', img)
    k = cv2.waitKey(0)
    print k
    cv2.destroyAllWindows()

test1()
```

The status bar at the bottom right indicates "Ln: 8 Col: 22".

# VIDEO



The image shows a screenshot of a Python IDE window titled "x.py - C:\Users\Dick\Desktop\x.py (2.7.13)". The window has a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains the following Python code:

```
import numpy as np
import cv2
import math

cap = cv2.VideoCapture(0)
while(True):
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    cv2.imshow('frame',gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
|
```

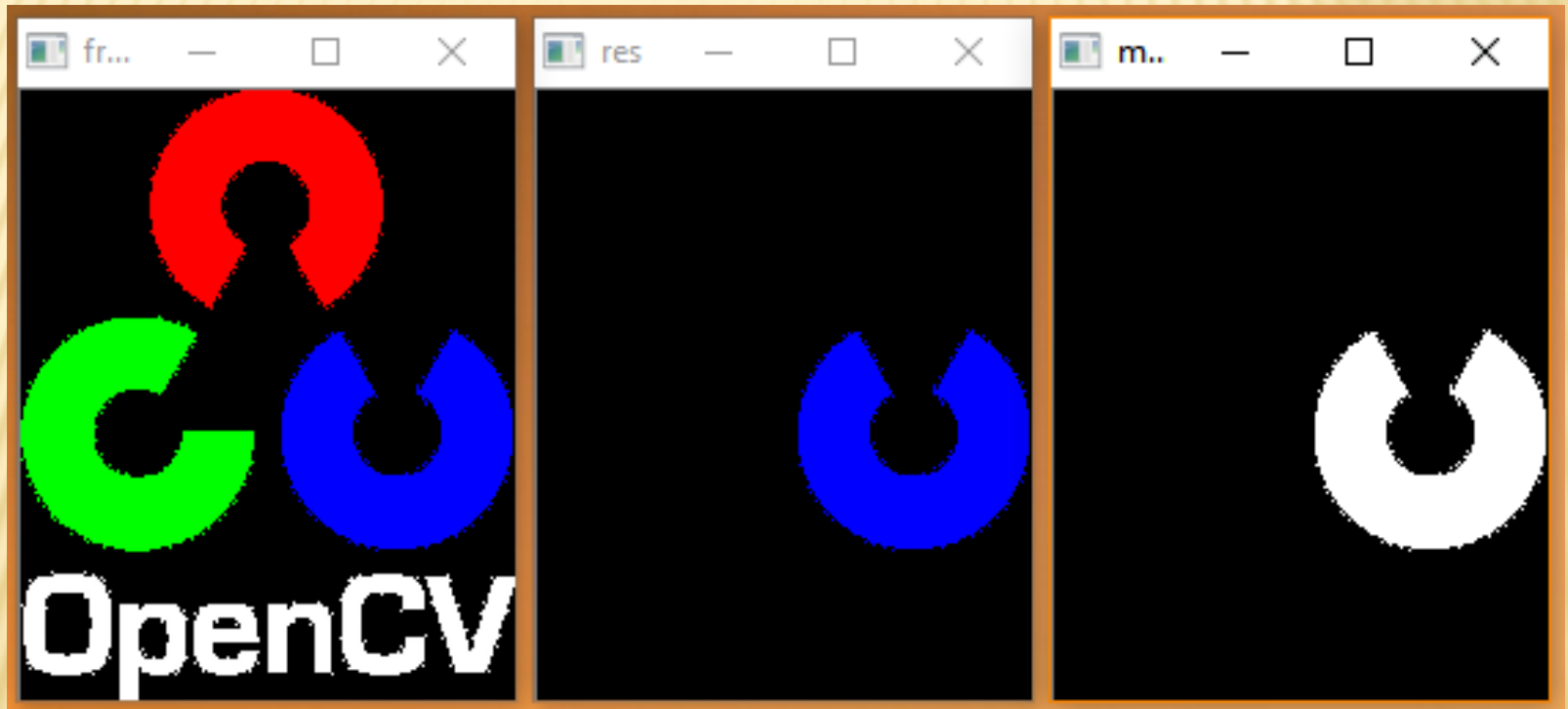
The status bar at the bottom right of the window displays "Ln: 14 Col: 0".

# OPDRACHT 1

- ✖ <https://opencv-python-tutroals.readthedocs.io/en/latest/>
- ✖ <http://docs.opencv.org/3.1.0/index.html>
  
- ✖ Zoek in de eerste tutorial naar
  - + OpenCV-Python Tutorials
  - + Image Processing in OpenCV
  - + Changing Colorspaces
  - + Object Tracking
- ✖ Pas de code zodanig aan dat je het blauwe deel van het OpenCV logo los kan laten zien
- ✖ Doe dit ook voor het groene en het rode deel



# RESULTAAT



# OPDRACHT 2



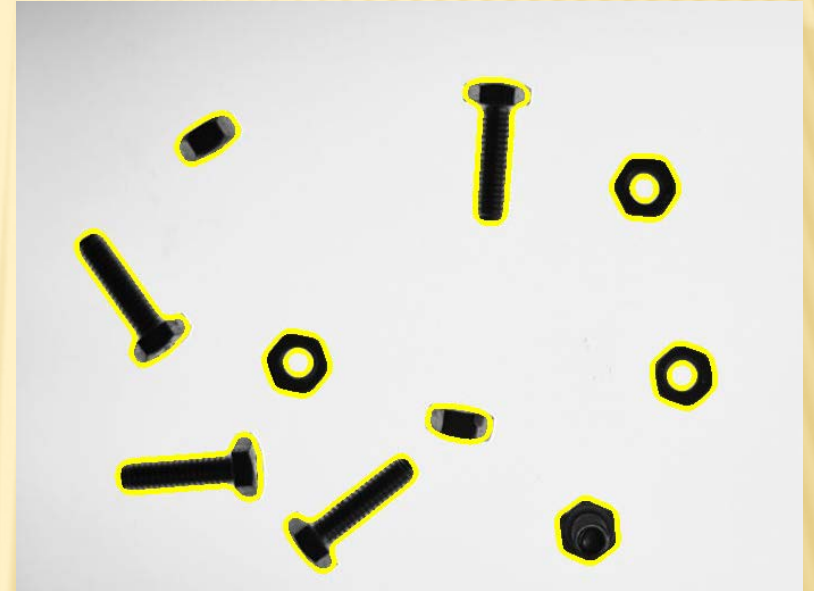
Convex hull

Centroid

# AANWIJZINGEN

- ✖ Lees: Image Processing in OpenCV
  - + Contours : Getting Started
    - ✖ cv2.findContours()
    - ✖ cv2.drawContours()
  - + Contour Features
    - ✖ Moments (pas deze toe op contours[0])
    - ✖ Convex Hull (pas deze toe op contours[0])
- ✖ Lees: Gui Features in OpenCV
  - + Drawing Functions in OpenCV
    - ✖ Drawing Circle

# OPDRACHT3



Vind de belangrijkste contouren



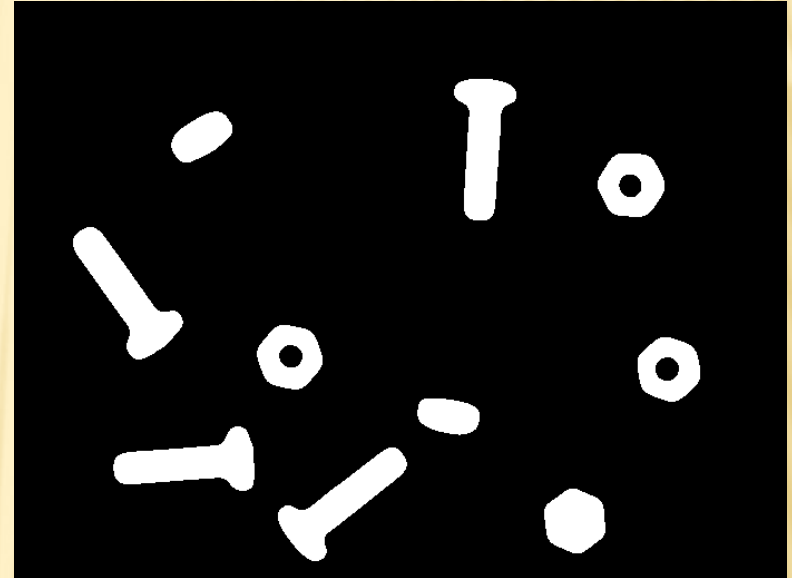
# AANWIJZINGEN



Converteer naar gray-scale mbv `cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)`  
Blur mbv `cv2.GaussianBlur(img,(5,5),0)`

Zie: Image Processing in OpenCV -> Smoothing Images -> Gaussian Filtering

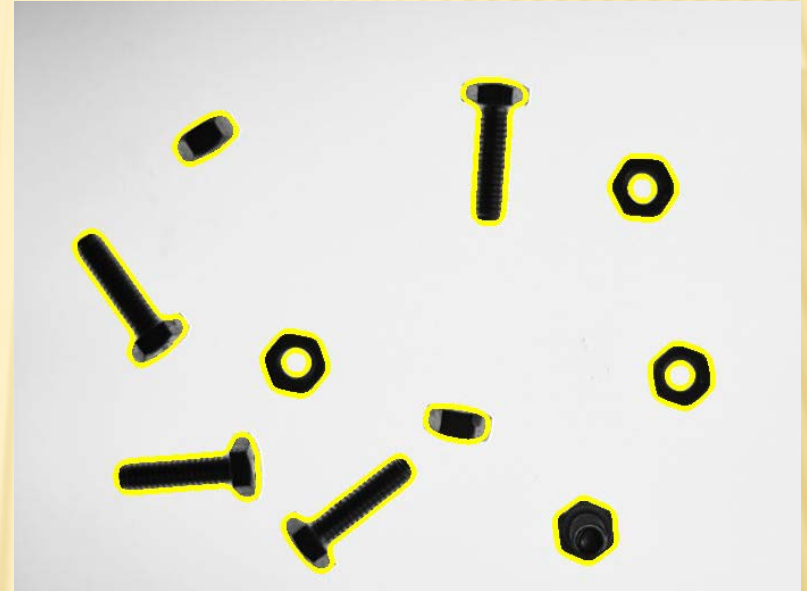
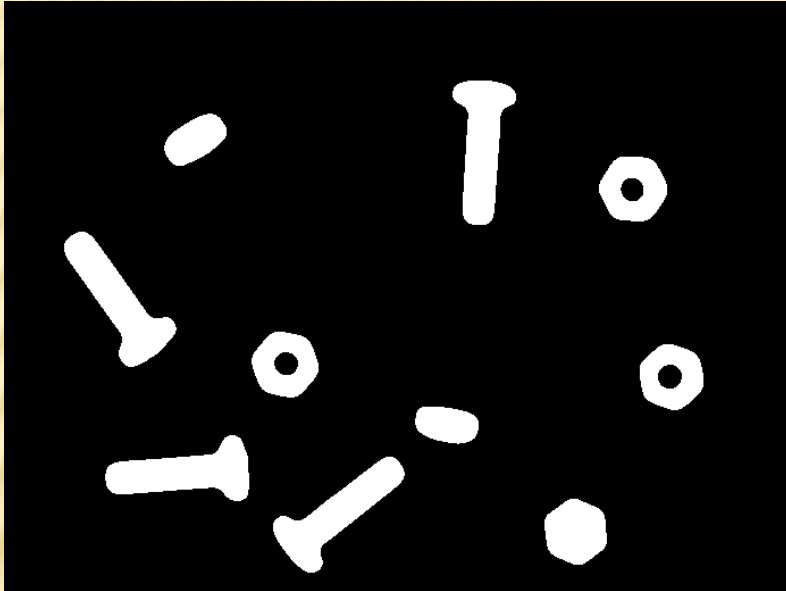
# AANWIJZINGEN



Vind blobs mbv `ret, th = cv2.threshold(img,180,255,cv2.THRESH_BINARY_INV)`

Zie: Image Processing in OpenCV -> Image Thresholding

# AANWIJZINGEN



Vind coutouren mbv

image, contours, hierarchy

```
= cv2.findContours(th,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
```

Teken ze met cv2.drawContours()

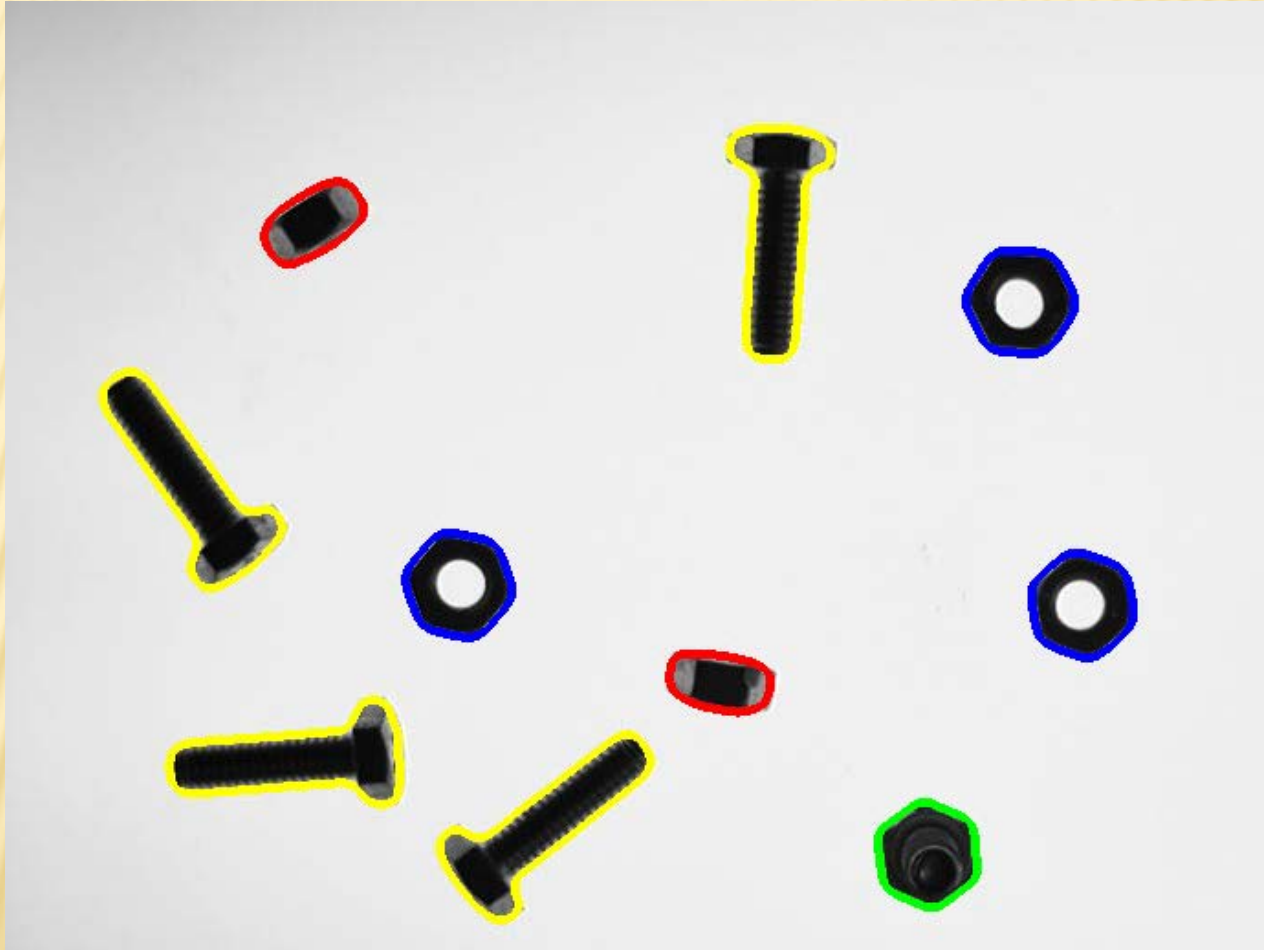


# AANWIJZINGEN

- ✗ `gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)`
- ✗ `gray = cv2.GaussianBlur(gray, (25,25), 0)`
- ✗ `ret, th = cv2.threshold(gray, 180, 255, cv2.THRESH_BINARY_INV)`
- ✗ `image, contours, hierarchy =  
cv2.findContours(th, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)`
  
- ✗ `for cnt in contours:`
- ✗  `area = cv2.contourArea(cnt)`
- ✗  `if area > 100:`
- ✗  `img = cv2.drawContours(img, [cnt], -1, (0,255,255), 3)`
  
- ✗ Zo teken je alleen contouren met een oppervlakte groter dan 100



# OPDRACHT 4



# AANWIJZINGEN

- ✖ De vormfactor van een contour geeft aan hoe rond een contour is
- ✖ Een cirkel heeft vormfactor 1
- ✖ Minder ronde contouren een vormfactor  $< 1$
- ✖ Hij wordt berekend als  $4 * \pi * opp / omtrek^2$
- ✖ Hiermee kun je bouten, staande moeren en liggende moeren onderscheiden

# AANWIJZINGEN

---

- ✗ Liggende moeren hebben een groot gat
- ✗ Staande bouten niet
- ✗ Via de contour hierarchy kun je ze onderscheiden
- ✗ Zie: Contours Hierarchy
- ✗ Dit is een lastig onderdeel!



# AANWIJZINGEN

```
✖ hierarchy = hierarchy[0]

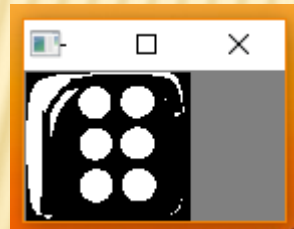
✖ for cnr in range(len(contours)):
✖     cnt = contours[cnr]
✖
✖     area = cv2.contourArea(cnt)
✖     perimeter = cv2.arcLength(cnt, True)
✖     factor = 4 * math.pi * area / perimeter**2
✖
✖     holes = 0
✖     child = hierarchy[cnr][2]
✖     while child >= 0:
✖         holes += cv2.contourArea(contours[child])
✖         child = hierarchy[child][0]
✖
✖     print area, factor, holes
```



# OPDRACHT 4 VERVOLG

- ✖ Maak een tekening van de hierarchy
- ✖ Leg voor jezelf uit hoe de berekening van “holes” werkt
- ✖ Zorg dat je alle typen van een andere kleur contour voorziet

# OPDRACHT 5



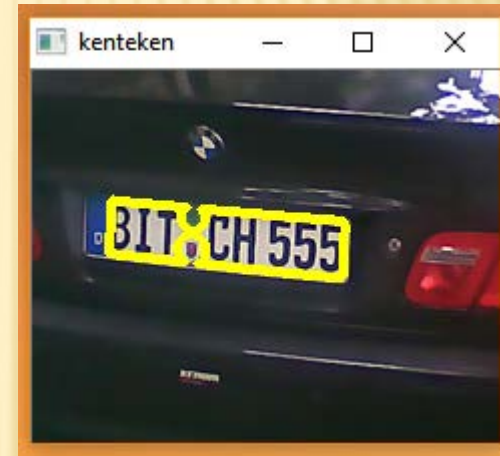
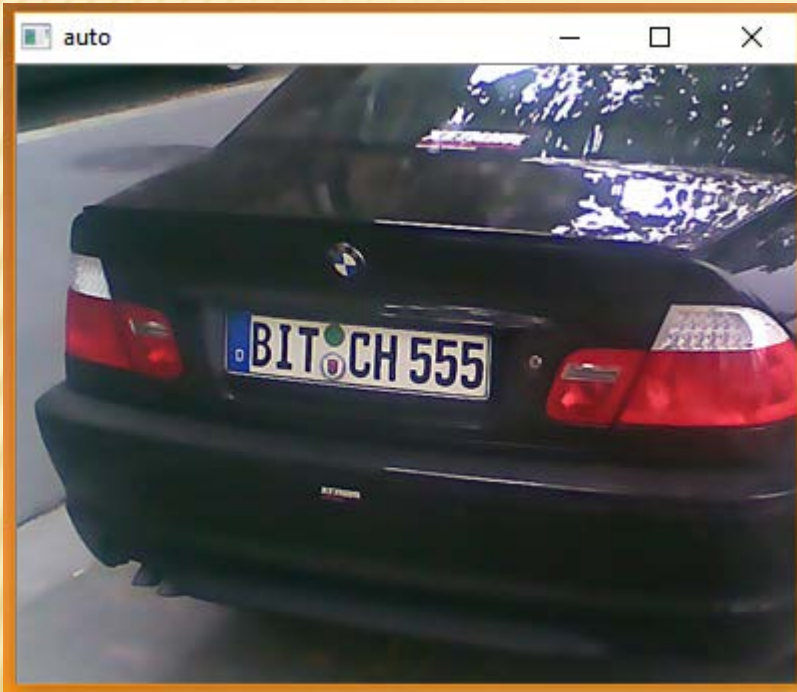
[3, 3, 3, 3, 4, 4, 5, 5, 6, 6, 6, 6]

# AANWIJZINGEN

- ✗ Zoek eerst de dobbelstenen via grayscale en threshold
- ✗ Doorloop de contouren en bereken ROI
  - ✗ `for cnr in range(len(contours)):`
  - ✗ `x,y,w,h = cv2.boundingRect(contours[cnr])`
  - ✗ `die = gray[y:y+h, x:x+w]`
- ✗ Voor elke “die” ga je thresholden en contouren berekenen. De ronde contouren zijn de “ogen”.
- ✗ Denk aan de vormfactor
- ✗ Zie: Core Operations -> Basic Operations



# OPDRACHT 6 (OPTIONEEL)



Neem een region of interest (ROI)  
Maak het blauwe en rode kanaal 0  
Teken de grootste contour  
Zie: Core Operations -> Basic Operations



# APPENDIX: VERSCHILLEN OPENCV 3.\* / 2.\*

## ✖ OpenCV 3.\*:

```
✖ image, contours, hierarchy
✖ = cv2.findContours(th,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
✖ img = cv2.drawContours(img, contours, -1, (0,255,0), 3)
✖ img = cv2.circle(img,(cx,cy), 5, (0,255,255), -1)
```

## ✖ OpenCV 2.\*:

```
✖ contours, hierarchy
✖ = cv2.findContours(th,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
✖ cv2.drawContours(img, contours, -1, (0,255,0), 3)
✖ cv2.circle(img,(cx,cy), 5, (0,255,255), -1)
```