LAB # 01: Introduction to Python , OpenCV&Numpy

Lab Objective:

To introduce students with python programming language ,opencv and numpy.

Lab Description:

Installation:

Download and install Python interpreter and PyCharm IDE from the following links below.

Python interpreter: https://www.python.org/downloads/

PyCharm IDE: https://www.jetbrains.com/pycharm/download/download-thanks.html?platform=windows&code=PCC

Setup:

1. Create 'New Project'
2. Now Check 'Existing interpreter' and Add local python interpreter that you've installed, and click Create.
3. Now Right click on the folder and create new python file.
4. After writing your python code right click on the window

   and run the project.

Python is an interpreted high-level programming language for general-purpose programming.Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons.

Operation	Syntax	Explanation / Example	Fill the outputs of all the print statements

Comment	#	# This is a comment.	
Print (is used to display anything in console window)	print( )	print (10) print (12+6) print (5, end=" ") print ("END")	
Operators	+ plus - minus * multiply ** power / divide // divide and floor % modulus	print(5**2) print (5//2) print (5/2) print (5%2)	
Variables (Python automatically guess the data type. There is no need to explicitly define data type in python)	x=5 y, z = 3.14, 17.2	right side is evaluated first and then assigned to left side with corresponding values print (x + y) print(type(x)) print(type(y))	
Strings	a="py" b='charm'	both single and double quotes worksexactly same print (a + b)	

		print (a, b) print (a*5) print (“hello’\”world\” ”)	
<u>Lists</u> List in memory stores references to objects. Each memory location is a pointer to an object. There is no obligation of similardata types	x=int (input (“Enter a number”) y = 3.14 z = "HELLO" li = [x, y, z, 4]	input ( ) function always input string, int ( ) function is used to convert string to int print(li)	
<u>List Indexing</u> # From left to right: 0 → 1 → 2 # From right to left: -1 → -2 → -3	Q = li [2] [-4]	print (Q)	
<u>List Slicing</u> list [start: end: step size]start is inclusive and end is exclusive	R = li [1:3]	print (li [1:3]) print (li [0:4:2]) print (li [:]) print (li [0:])	

defaults values are list [0 : end : 1]		print (li [:3]) print (li [2] [1:4])	
<u>Copy</u> The assignment copies the reference to the original list while slicing creates a new list	w = [1, 2, 3, 4] x = w y = w [:]	x [0] = 6 y [1] = 9 print(w)	
<u>Indentation</u>	x =2 if x==10: print("inside”) print("inside") print("outside")	Whitespace (spaces and tabs) at the beginning of the logical line is used to determine the indentation level of the logical line, which in turn is used to determine the grouping of statements.	
<u>Boolean operations</u>	< (less than) >(greater than) <=(less than/equal to) >=(greaterthan/equal) == (equal to) ! = (not equal to)		

	not (Boolean NOT)
	and (Boolean AND)
	or (Boolean OR)
		if not x:
		if x==2 and y>4:
		print (2==4)	

<u>if</u>
If the Boolean expression evaluates to true, then the if block of code will be executed
	number = 23
	if number == 24:
	print ('equal')
	elif number<24:
	print ('less')
	else:
	print ('greater')
		If statement does not include brackets and ends with colons. The if block is determine by indentation level	

<u>while</u> Repeats a statement or group of statements while a given condition is true	number = 23 while number<30: print(number) number+=1		

<u>for</u>
Execute a sequence of statements multiple times and abbreviates the code that
	for i in [2,3,4,1,5]:
	print(i)
	for j in range(1,9,2):
	print(j)
		Range function is used to generate a list of numbers, which is generally used to iterate over with for loops	

manages the loop variable			

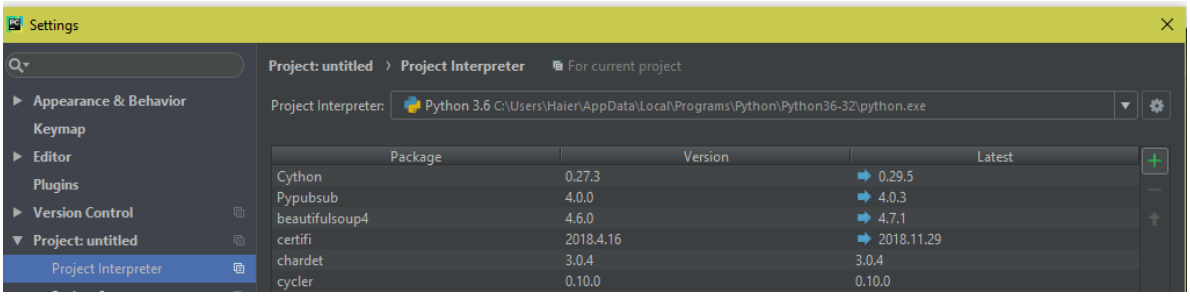
Functions
 	
A function is a block of code which only runs when it is called
	def max (x, y):
	
	if x > y:
	
	return x
	
	else:
	
	return y
	
	m=max (3,5)
	
	print(m)		

OpenCV:

Open Source **C**omputer **V**ision Library (**OpenCv**)is an open source software library written in C++ for machine learning and computer vision applications.

To install packages in PyCharm Click, Files > Settings > Project: Name > Project Interpreter

Now click on '+' icon to install new packages



Now search for following packages and Install them by clicking 'Install Package'

- opencv-python
- numpy // Used for numerical operations
- matplotlib // Used to plotting graphs

Why OpenCv?

OpenCv is comprehensive collection of more than 2500 machine learning and computer vision algorithms that can be used from something as simple detecting faces in images to project augmented reality overlaid with scenery.

Another area in which OpenCv excels is its superior support for multiple interfaces and all the major operating systems as compared to other solutions. OpenCv supports Windows, Linux, OS X and Android and provides interfaces for C, C++, Python, Java and MATLAB.

Some of the applications that can be accomplished easily with OpenCv are: identifying objects, tracking camera movements, stitching images together, finding similar images in a database using an image, face detection and tracking moving objects in a video feed etc.

Some Useful Commands:

1. To slice a 2D array:

x = y [row_start: row_ end, col_start: col_end]

2. To create a 2D array of zeros using NumPy:

my_array = **numpy.zeros ((row, columns), dtype=numpy.uint8)**

3. To create a 2D array of ones using NumPy:

my_array = **numpy.ones ((row, columns), dtype=numpy.uint8)**

4. To check the size of a 2D array: size = **numpy.shape(my_array)**

5. To join a sequence of arrays along an existing axis: F = **np.concatenate ((a, b), axis=0);**

6. To assemble an array from nested lists of blocks.

img = np.block **([[np.ones ((2, 2)), np.zeros ((2, 3))], [np.zeros ((2, 2)), np.ones ((2, 3))]])**

Note: all the input array dimensions except for the concatenation axis must match exactly

7. Reading an image using OpenCv: my_image = **cv2.imread("test_image.jpg",0)**

The second argument determines whether the image is read as a grayscale image or a colored image. **0** is used for reading an image as grayscale and while **1** is used for reading in color. If no argument is passed then the image is read as is.

8. Displaying an image using OpenCv: **cv2.imshow ("Title of the window", my_image**).

   Two more commands that need to be used while displaying an image are:
   cv2.waitKey(x)

   cv2.destroyAllWindows (). The waitKey () function waits for a key being pressed for x number of milliseconds. If **0** is passed to waitKey () as an argument, it will wait indefinitely for a key press. **cv2.destroyAllWindows ()** closes are the open image windows.

9. Writing an image to disk: **cv2.imwrite("image_name.jpg", my_image)**

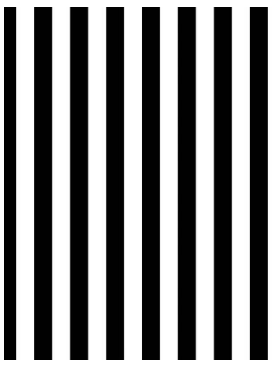
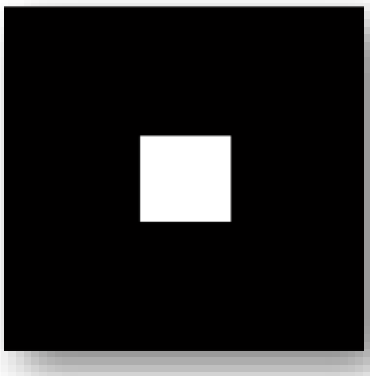
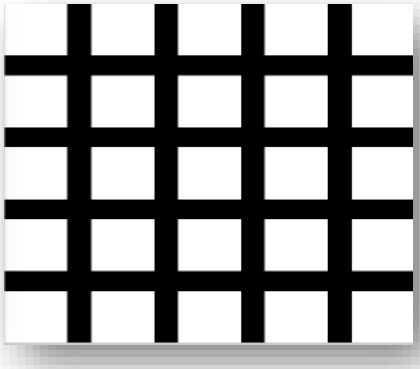
<u>Lab Tasks:</u>

1. Write a Python function that takes a list as input and performs the min-max normalization on the list. The function should return the scaled list.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

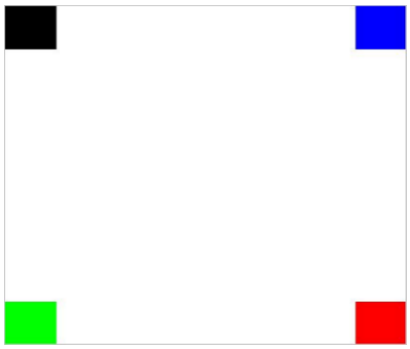
2. Create an r x c matrix of ones and pad 10 pixels wide border of zeros across each side of it, such that its order will become (5+500+5) x (5+500+5) = 510x510

3. Write 3 different Python functions that can create the images given below. Code them in such so that the size of the image itself and the boxes and lines can be changed.

4. Read an image and resize it to 512x512 using the appropriate function. Then down sample the image by 4 so that the final size of the image is 128x128. Display and save the image to the disk.

5. Write a function to create a white image of 500x500 (or any other size entered by the user) and then create 4 boxes of Red, Green, Blue and Black respectively on each corner of the image as shown below. The size of the colored boxes should be 1/8th the size of the image. (HINT: the arrays of ones and zeros can be in more than 2 dimensions).



6. Mirror the image that you have read at center i.e. the lower half of the image should be the copy of the upper half. (HINT: You can use nested loops)