

Python 入门基础和 Python 视觉应用实验报告

姓名：孔维凯 学号：23020007051

2024 年 9 月 13 日

目录

1	Python 入门基础	5
1.1	实例 1	5
1.1.1	练习内容	5
1.1.2	结果	5
1.1.3	解题感悟	5
1.2	实例 2	5
1.2.1	练习内容	5
1.2.2	结果	5
1.2.3	解题感悟	5
1.3	实例 3	6
1.3.1	练习内容	6
1.3.2	结果	6
1.3.3	解题感悟	6
1.4	实例 4	6
1.4.1	练习内容	6
1.4.2	结果	6
1.4.3	解题感悟	7
1.5	实例 5	7
1.5.1	练习内容	7
1.5.2	结果	8
1.5.3	解题感悟	8
1.6	实例 6	8
1.6.1	练习内容	8
1.6.2	结果	8
1.6.3	解题感悟	8
1.7	实例 7	9
1.7.1	练习内容	9
1.7.2	结果	9
1.7.3	解题感悟	9
1.8	实例 8	10
1.8.1	练习内容	10
1.8.2	结果	10
1.8.3	解题感悟	10
1.9	实例 9	10

1.9.1	练习内容	10
1.9.2	结果	10
1.9.3	解题感悟	11
1.10	实例 10	11
1.10.1	练习内容	11
1.10.2	结果	11
1.10.3	解题感悟	11
2	Python 视觉应用	12
2.1	实例 1	12
2.1.1	练习内容	12
2.1.2	结果	12
2.1.3	解题感悟	13
2.2	实例 2	13
2.2.1	练习内容	13
2.2.2	结果	13
2.2.3	解题感悟	13
2.3	实例 3	13
2.3.1	练习内容	13
2.3.2	结果	13
2.3.3	解题感悟	14
2.4	实例 4	14
2.4.1	练习内容	14
2.4.2	结果	14
2.4.3	解题感悟	15
2.5	实例 5	15
2.5.1	练习内容	15
2.5.2	结果	15
2.5.3	解题感悟	15
2.6	实例 6	15
2.6.1	练习内容	15
2.6.2	结果	15
2.6.3	解题感悟	16
2.7	实例 7	16
2.7.1	练习内容	16
2.7.2	结果	16

2.7.3	解题感悟	16
2.8	实例 8	17
2.8.1	练习内容	17
2.8.2	结果	17
2.8.3	解题感悟	17
2.9	实例 9	17
2.9.1	练习内容	17
2.9.2	结果	17
2.9.3	解题感悟	18
2.10	实例 10	18
2.10.1	练习内容	18
2.10.2	结果	18
2.10.3	解题感悟	18
3	github 链接	18

1 Python 入门基础

1.1 实例 1

1.1.1 练习内容

字符串连接

1.1.2 结果

比如输入 `print("hello"+"world"+"!")`, 加号会将 `hello` 和 `world` 和 `!` 连接成更长的字符串。

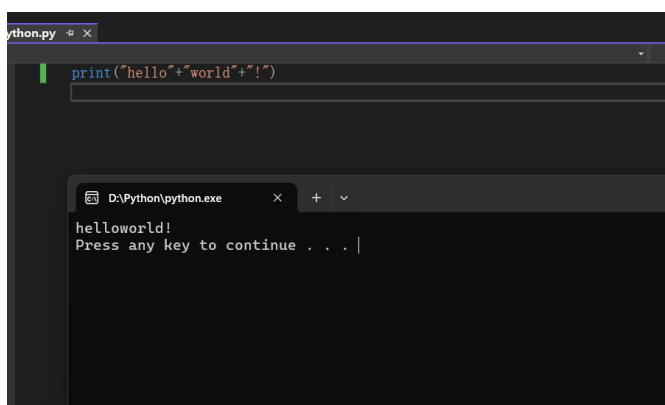


图 1: 例图 1

1.1.3 解题感悟

这种方式输入很有意思。

1.2 实例 2

1.2.1 练习内容

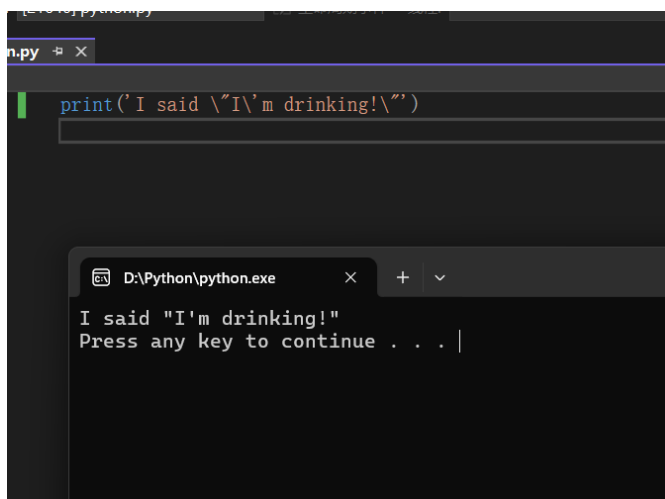
python 中 `"""` 和 `'''` 都可以用来输出字符, 但如果语句中既有 `"""` 又有 `'''` 该怎么做来达到预期效果

1.2.2 结果

在语句的 `'` 和 `"""` 前加上反斜杠提示系统即可

1.2.3 解题感悟

这样做就可以方便输出带有 `"""` 或 `'` 的语句, 方便有效。



```
print('I said \'I\'m drinking!\')
```

```
I said "I'm drinking!"  
Press any key to continue . . . |
```

图 2: 例图 2

1.3 实例 3

1.3.1 练习内容

如何换行

1.3.2 结果

第一种方法是加上斜杠 `n`

第二种方法是使用多个 `print` 语句，因为在 `python` 中 `print` 默认换行

第三种方法是使用 `'''` 或者 `"""` 将语句包裹，系统就明白下一行是想要换行而不会报错。

1.3.3 解题感悟

这三种方法可以便于用户换行，尤其是第三种方法便于用户在大量文字时换行。

1.4 实例 4

1.4.1 练习内容

如何使用三角函数

1.4.2 结果

首先使用 `import math` 引入 `math` 库，然后使用 `math.+ 函数名` 即可使用该函数，比如使用 `sin` 函数

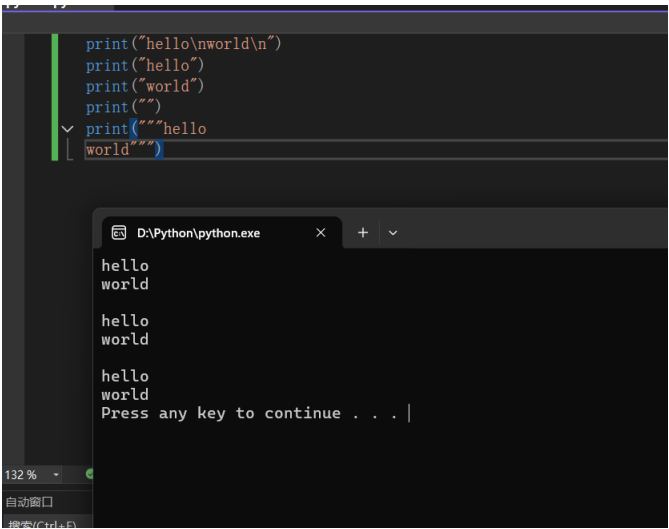


图 3: 例图 3

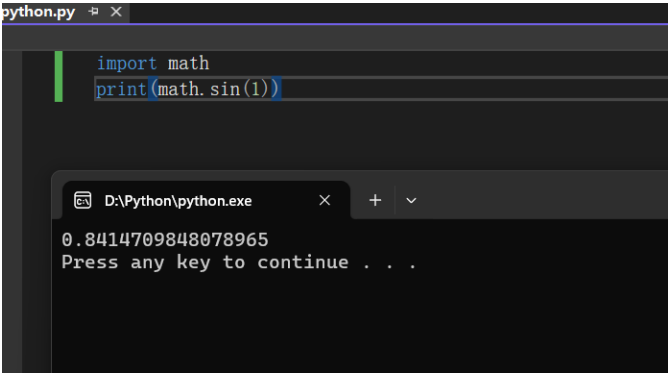


图 4: 例图 4

1.4.3 解题感悟

import 不仅可以引入 math 库还可以引入其他库，这方便用户使用现成函数，不需要自己重新写一个函数，方便快捷。

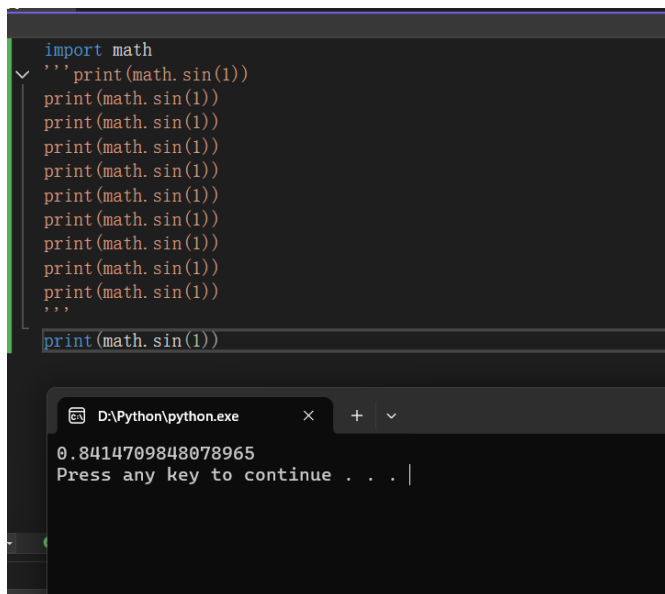
1.5 实例 5

1.5.1 练习内容

如何添加多行注释

1.5.2 结果

可以使用 `#` 一个一个添加注释，但是过于繁琐，一种更简便的方法是加入 `'''` 或 `"""` 将内容包裹就可以全变成注释。



```
import math
'''print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
print(math.sin(1))
'''
print(math.sin(1))
```

D:\Python\python.exe × + ▾

```
0.8414709848078965
Press any key to continue . . . |
```

图 5: 例图 5

1.5.3 解题感悟

使用这种方法便于用户快速添加多行注释。

1.6 实例 6

1.6.1 练习内容

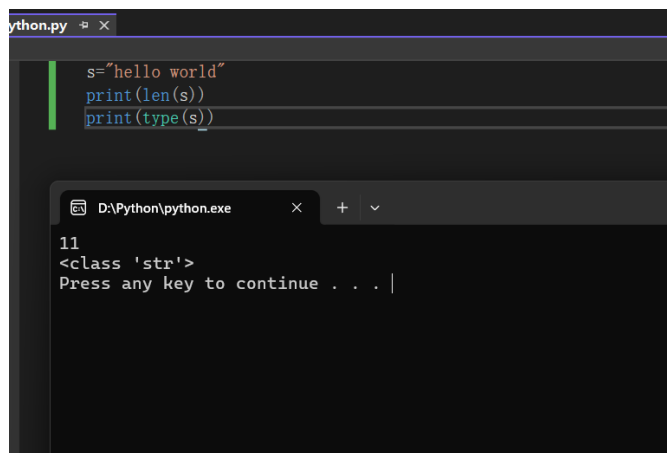
如何对字符串求长度和显示数据类型

1.6.2 结果

使用 `len` 函数就可以对字符串求长度，但是只是存在内存中，要 `print` 才能显示其长度，用 `type` 函数显示数据类型，也需要 `print` 才能显示

1.6.3 解题感悟

使用 `type` 函数可以知道数据类型，这样才能知道该数据可以用于什么函数中，防止报错
`len` 函数可以快速求出字符串长度。



```
python.py x
s="hello world"
print(len(s))
print(type(s))

D:\Python\python.exe x + v
11
<class 'str'>
Press any key to continue . . . |
```

图 6: 例图 6

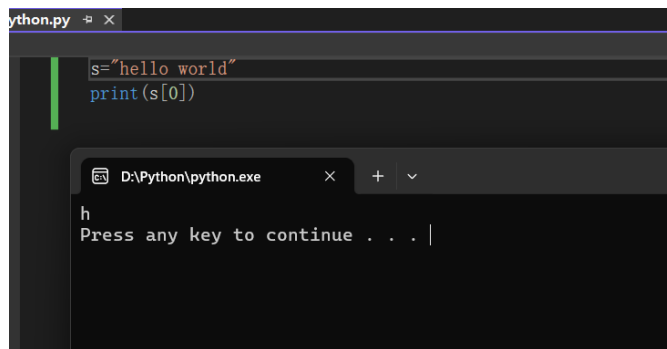
1.7 实例 7

1.7.1 练习内容

如何获取字符串中的单个字符

1.7.2 结果

只需要 `[]` 就可以得到某个字符，要注意索引从 0 开始，也就是说 0 对应第一个字符，依此类推。



```
python.py x
s="hello world"
print(s[0])

D:\Python\python.exe x + v
h
Press any key to continue . . . |
```

图 7: 例图 7

1.7.3 解题感悟

使用 `[]` 可以快速得到字符串中的某个字符。

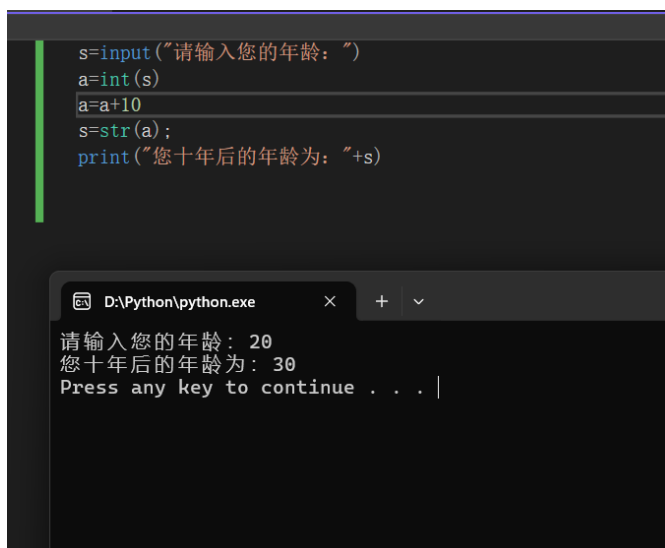
1.8 实例 8

1.8.1 练习内容

让用户输入数据，并转换数据类型

1.8.2 结果

使用 `input` 函数，但要注意 `input` 函数返回的数据类型是字符类型，如果想要进行一些不适合字符类型的操作要进行字符转化，如 `int` 可以将数据转换为整数类型。



```
s=input("请输入您的年龄：")
a=int(s)
a=a+10
s=str(a);
print("您十年后的年龄为："+s)
```

D:\Python\python.exe x + v

请输入您的年龄：20
您十年后的年龄为：30
Press any key to continue . . . |

图 8: 例图 8

1.8.3 解题感悟

这样就可以完成一些交互操作，而不单单是自说自话，数据类型转换方便数据在不同情况下的使用。

1.9 实例 9

1.9.1 练习内容

使用条件语句

1.9.2 结果

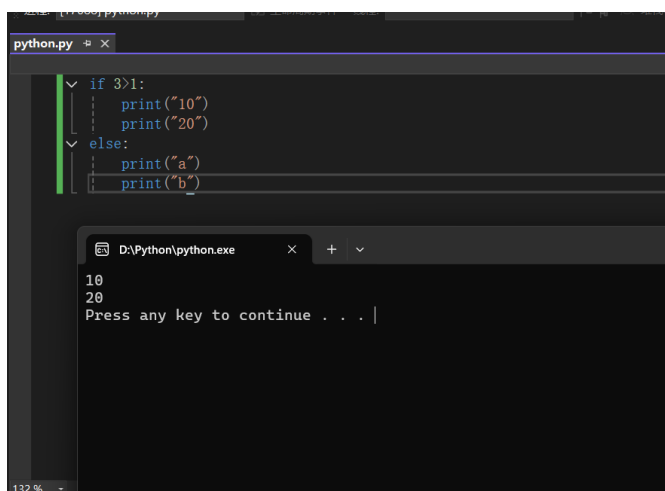
python 中条件语句的格式是
`if` 条件:

执行语句

else:

执行语句

注意执行语句需要有缩进，因为 python 会根据缩进判断这是 if 语句执行语句还是 else 语句执行语句亦或者不是条件语句的执行语句。



The screenshot shows a Python IDE with a file named 'python.py'. The code in the editor is:

```
if 3>1:
    print("10")
    print("20")
else:
    print("a")
    print("b")
```

Below the editor, a terminal window titled 'D:\Python\python.exe' shows the output of the program:

```
10
20
Press any key to continue . . . |
```

图 9: 例图 9

1.9.3 解题感悟

if 条件语句使得代码更加丰富，也使作用更加广泛。

1.10 实例 10

1.10.1 练习内容

使用 for 循环语句

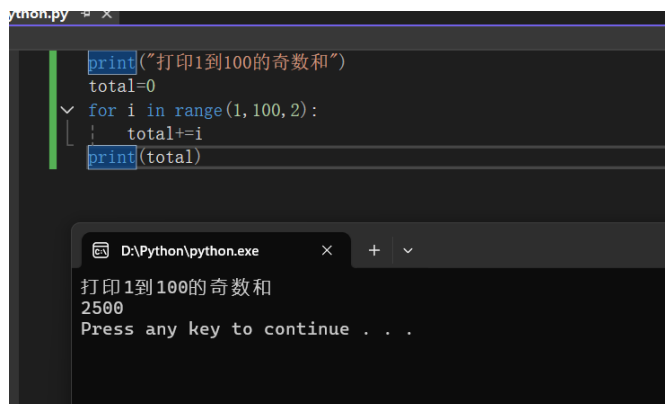
1.10.2 结果

格式为 for 变量名 in 可迭代对象:

执行语句，这样就可以进行循环操作

1.10.3 解题感悟

循环操作极大地简便了用户的操作，也使代码变得简洁。



```
python.py x
print(~打印1到100的奇数和")
total=0
for i in range(1,100,2):
    total+=i
print(total)
```

D:\Python\python.exe x + v

打印1到100的奇数和
2500
Press any key to continue . . .

图 10: 例图 10

2 Python 视觉应用

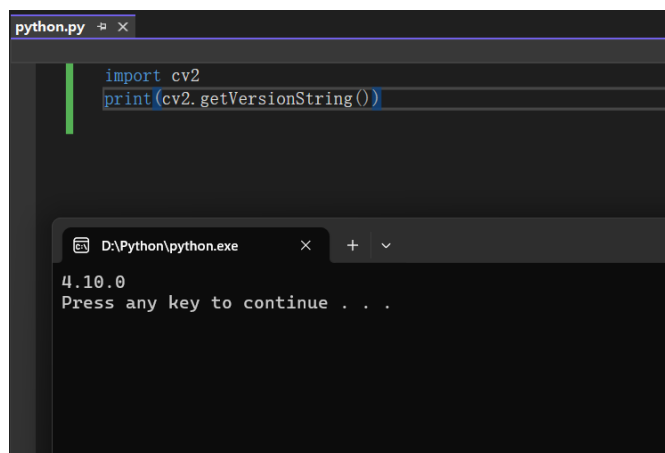
2.1 实例 1

2.1.1 练习内容

显示 OpenCV 的版本号

2.1.2 结果

首先引入 cv2 库，然后使用 cv2.getVersionString() 函数，用 print 函数将其输出即可得到版本号



```
python.py x
import cv2
print(cv2.getVersionString())
```

D:\Python\python.exe x + v

4.10.0
Press any key to continue . . .

图 11: 例图 11

2.1.3 解题感悟

使用这个函数可以查看 OpenCV 的版本号是否符合用户要求。

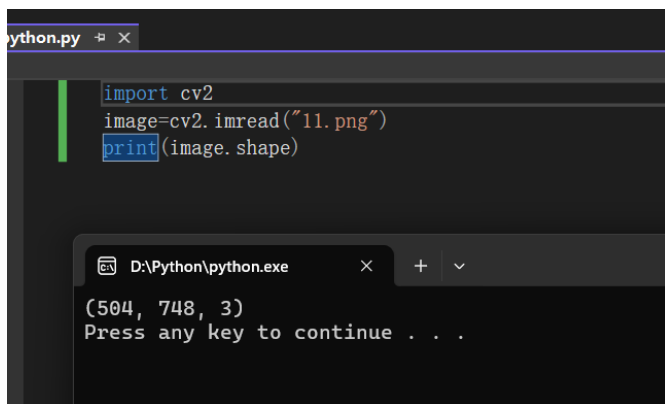
2.2 实例 2

2.2.1 练习内容

显示图片的维度

2.2.2 结果

首先引入 cv2 库，然后使用 cv2.imread 函数读取图片，再使用 shape 函数将图片维度显示出来，print 函数输出得到图片维度。

The image shows a screenshot of a code editor window titled 'python.py' and a terminal window. The code in the editor is:

```
import cv2
image=cv2.imread("11.png")
print(image.shape)
```

The terminal window shows the output:

```
(504, 748, 3)
Press any key to continue . . .
```

```
python.py
import cv2
image=cv2.imread("11.png")
print(image.shape)

D:\Python\python.exe
(504, 748, 3)
Press any key to continue . . .
```

图 12: 例图 12

2.2.3 解题感悟

使用这个函数，用户可以得到图片的维度数据。

2.3 实例 3

2.3.1 练习内容

如何显示图片并使图片停留

2.3.2 结果

使用 cv2.imshow 函数显示图片，但只是这样，图片只会出现一瞬，这时，你如果想要让图片停留，就需要使用 cv2.waitKey 函数。

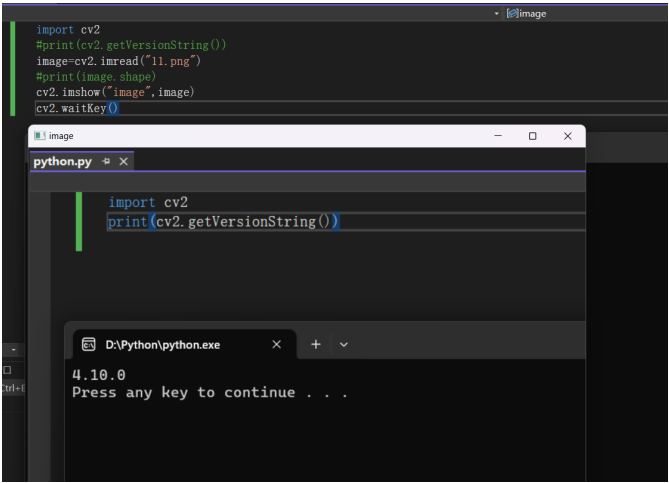


图 13: 例图 13

2.3.3 解题感悟

使用这些函数，用户可以显示图片并使其停留。

2.4 实例 4

2.4.1 练习内容

如何使用 OpenCV 提取红绿蓝三张灰度图

2.4.2 结果

使用 `cv2.imshow("red",image[:, :, 2])` `cv2.imshow("green",image[:, :, 1])` `cv2.imshow("blue",image[:, :, 0])` 函数，对应的红绿蓝分别对应 2,1,0，两者顺序相反。

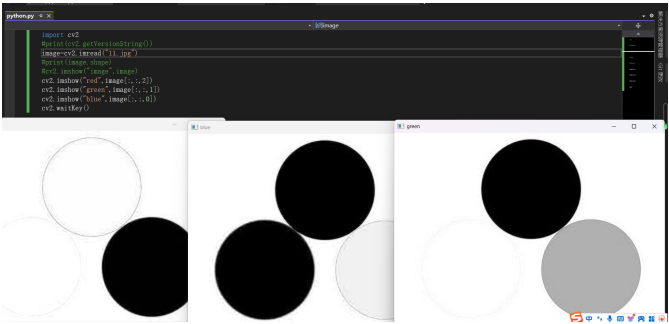


图 14: 例图 14

2.4.3 解题感悟

通过这种做法可以看到红绿蓝三张灰度图。

2.5 实例 5

2.5.1 练习内容

OpenCV 的彩色图像的灰度变换算法，显示一张 3 原色平均图

2.5.2 结果

首先使用 `gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)` 函数将三原色加权平均, 然后使用 `cv2.imshow("gray"` 函数就能显示一张 3 原色平均图。

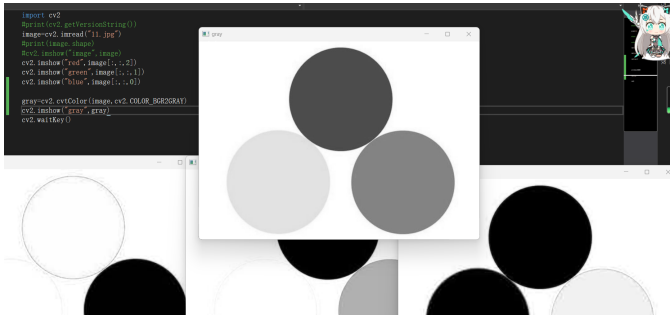


图 15: 例图 15

2.5.3 解题感悟

这样就可以显示三原色平均图

2.6 实例 6

2.6.1 练习内容

如何实现图像的裁剪

2.6.2 结果

使用 `crop=image[...]` 将图像进行裁剪, [...] 中是裁剪的区域, 然后使用 `cv2.imshow("crop",crop)` 将裁剪后的图像显示出来。

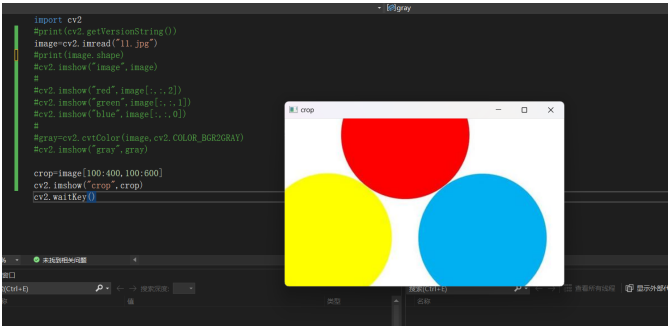


图 16: 例图 16

2.6.3 解题感悟

这样就可以精准地实现图片的裁剪。

2.7 实例 7

2.7.1 练习内容

如何使用 OpenCV 绘制

2.7.2 结果

首先要引入 numpy 库,然后使用 np.zeros() 函数来绘制,比如绘制一张黑色画布,使用 np.zeros([],dtype=np.uint8),其中 [] 中添加维度,黑色为 dtype,最后使用 cv2.imshow 函数显示黑色画布。

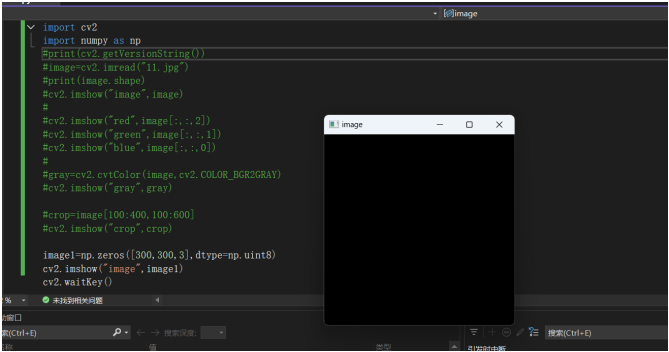


图 17: 例图 17

2.7.3 解题感悟

这是绘制的前置工作作为后续绘制做准备。

2.8 实例 8

2.8.1 练习内容

如何在画布上画一条蓝线

2.8.2 结果

使用 `cv2.line(,(),(),(),)` 函数第一个位置填入画布，第一个括号中填入线段起点，第二个括号填入线段终点，第三个括号填入线段颜色号码，最后一个位置填入像素大小。

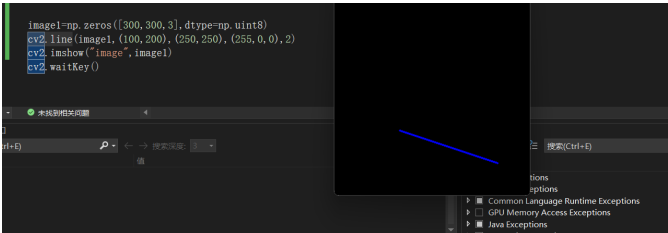


图 18: 例图 18

2.8.3 解题感悟

这样就进行了简单的绘制。

2.9 实例 9

2.9.1 练习内容

如何画一个矩形框，圆环

2.9.2 结果

使用 `cv2.rectangle(,(),(),(),)` 函数，第一个位置输入画布，第一个括号中输入第一个顶点坐标，第二个括号中输入对角顶点坐标，第三个括号输入颜色编号，最后输入像素大小，就可以得到一个矩形框。同理使用 `cv2.circle()` 函数可以画出一个圆环。



图 19: 例图 19

2.9.3 解题感悟

使用这个函数已经可以完成简单的绘制了。

2.10 实例 10

2.10.1 练习内容

如何来绘制字符串

2.10.2 结果

使用 `cv2.putText(, , , , , ,)` 函数，第一个位置输入画布，第二个位置输入要输入的字符串，第三个位置是坐标，后面分别是字体序号，字体缩放系数，然后是颜色编号，像素大小，线条类型。

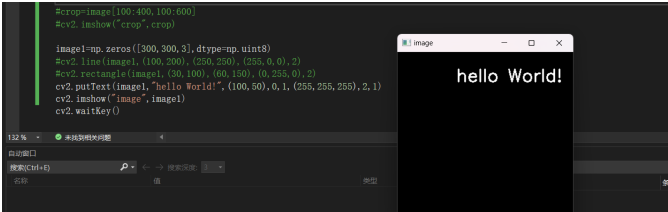


图 20: 例图 20

2.10.3 解题感悟

这样就可以绘制字符串，书写说明等内容。

3 github 链接