深 圳 大 学 实 验 报 告

课程名称：Python 程序设计

实验项目名称：实验 6：Python 实战

学院：计算机与软件学院

专业：软件工程

指导教师：潘浩源

报告人：郑彦薇 学号：2020151022

实验时间：2022/6/10~2022/6/19

实验报告提交时间：2022/6/14

教务部制

一、实验目的

用 python 语言编写解决问题的代码并运行。

利用 python 中 tkinter 开发 GUI 项目实现对问题的解决。

二、实验方法步骤

1、读题，对每个问题提出解决问题的思路

2、按照得到的思路，利用 python 语言编写解决问题的代码

3、运行代码，调试程序，直至程序可以正确输入输出

三、实验过程及内容

（一）解题思路和方法

1. 编写 GUI 程序--随机球：

1) 首先需要随机生成 10 个小球，调用 random 库，设置循环，随机生成点的位置坐标(x, y)，并设置圆半径。

2) 接着需要为圆随机生成颜色，同样使用 random 随机得到颜色的编号，返回得到的值，作为当前小球的颜色，进行填充

3) 主函数：定义窗口标题、提示语、画布以及按钮，并设置按钮功能。

代码及细节解释如下：

```
from tkinter import *
import random
def randomcolor():#让python随机生成颜色方法
    colorArr = ['1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F']
    color = ""
    for i in range(6):
        color += colorArr[random.randint(0, 14)]
    return '#' + color

def display():
    canvas.delete('all')#清空画布
    for i in range(10):
        x = random.randint(50, 450)
        y = random.randint(50, 250)#随机生成球的位置坐标
        R = 10#设定半径
        color = randomcolor()#获取随机颜色
        canvas.create_oval(x, y, x+R, y+R, fill=color, outline="black")

root = Tk()
root.title('Random Balls')
label = Label(root, text="click button to get balls")
label.pack()
canvas = Canvas(root, width=500, height=300)#添加画布
canvas.pack()
button = Button(root, text='Display', command=display)
button.pack()
root.mainloop()
```

随机生成 10 个球的方法

随机生成球的位置坐标，其中坐标不应超过设置的画布大小

2. 编写 GUI 程序--线连接两个半径为 20 的圆：

1) 初始化窗口信息：首先生成指定大小的空白画布，初始化两个圆的位置坐标及大小，并对初始信息进行显示。

2) 窗口信息展示：显示初始化的两个圆，生成一条直线连接两个圆，在直线上指定某一位置显示两圆之间的距离。

3) 更新圆的位置及距离：使用 canvas 库中的 bind 功能实现对鼠标的监听，然后根据鼠

标位置判断当前被挪动的是哪个圆，挪动圆，进行圆是否重合的判断。若重合，进行重合处理；若未重合，更新圆的位置坐标为鼠标的位置坐标，然后对当前信息（包括两圆、两圆之间的连线、两圆之间的距离）进行展示。

4) 圆重合的处理：当两圆的圆心重合（即两圆圆心距离为 0）时，使用 messagebox 功能显示错误信息框，然后将圆的位置恢复到初始状态并进行展示。

代码即细节解释如下：

初始化窗口信息：

```
def __init__(self):
    root = Tk()
    root.title("Circle")
    label = Label(root, text="drag circles to get dis")
    label.pack()
    self.canvas = Canvas(root, width=500, height=300)  # 添加画布
    self.canvas.pack()
    # 初始化两个圆的位置
    self.x1 = 100
    self.y1 = 100
    self.x2 = 200
    self.y2 = 200
    self.r = 20
    self.display_ele()
    self.canvas.bind("<B1-Motion>", self.MyMouse)
    root.mainloop()
```

监听鼠标

窗口信息展示：

```
def get_dis(self):
    return ((self.x1 - self.x2)**2 + (self.y1 - self.y2)**2)
def display_ele(self):
    self.canvas.delete('all')
    # 画布上显示两个圆
    self.canvas.create_oval(self.x1-self.r, self.y1-self.r, self.x1+self.r, self.y1+self.r, fill='red',
                            outline="black", tag='circle1')
    self.canvas.create_oval(self.x2-self.r, self.y2-self.r, self.x2 + self.r, self.y2 + self.r, fill='red',
                            outline="black", tag='circle2')
    # 两圆圆心确定一条直线
    self.canvas.create_line(self.x1, self.y1, self.x2, self.y2)
    # 线上显示两圆距离，距离在线上显示
    self.canvas.create_text((self.x1+self.x2)/2, (self.y1+self.y2)/2, text="{:.2f}".format(self.get_dis()))
```

计算距离，进行展示

距离信息显示位置

更新圆的位置：

```
def MyMouse(self, event):
    newX, newY = event.x, event.y
    # 圆1被移动
    if self.x1 - self.r < newX < self.x1 + self.r and self.y1 - self.r < newY < self.y1 + self.r:
        if self.get_dis() == 0:  # 如果两圆重合
            self.Dealcoincide()
            return
        # 圆没有重合，更新圆的新位置并进行展示
        self.x1 = newX
        self.y1 = newY
        self.display_ele()
    # 圆2被移动
    elif self.x2 - self.r < newX < self.x2 + self.r and self.y2 - self.r < newY < self.y2 + self.r:
        if self.get_dis() == 0:  # 如果两圆重合
            self.Dealcoincide()
            return
        # 圆没有重合，更新圆的新位置并进行展示
        self.x2 = newX
        self.y2 = newY
        self.display_ele()
```

鼠标位置记录

两圆重合，进行重合处理

圆的重合处理：

```
def Dealcoincide(self):
    # 显示错误信息框
    messagebox.showinfo("Error!", "Forbidden Coincide!")
    # 恢复初始状态
    self.x1 = 100
    self.y1 = 100
    self.x2 = 200
    self.y2 = 200
    self.display_ele()
```

3. 图片处理 1：

1) 首先获取即将进行拼接的图片的路径，将几张图片的路径存放在列表 paths 中（我这里进行了 5 张图像的拼接，因此路径存放设置的循环次数为 5）。

2) 打开待拼接图像，统一图像的大小为 741*986；以矩阵形式表示图像信息，借助 numpy 对图像进行水平或垂直拼接。

3) 将拼接完成的图像重新转换为图像类型，指定保存路径对图像进行保存。

代码及细节解释如下：

```
# 此处为路径，获取图像路径存放到paths中
img_path = "C:/Users/4334/Desktop/img_path"
paths = []
for i in range(5):
    paths.append(img_path + '/0' + str(i+1) + '.jpg')
#print(paths)
img_array = ''
img = ''
for i, v in enumerate(paths):
    if i == 0:
        img = Image.open(v)  # 打开图片
        # 此放处将单张图像进行缩为统一大小
        img = img.resize((741, 986), Image.ANTIALIAS)
        img_array = np.array(img)  # 转化为np array对象
    if i > 0:
        img = Image.open(v)
         # 此处将单张图像进行缩放为统一大小
        img = img.resize((741, 986), Image.ANTIALIAS)
        img_array1 = np.array(img)
        img_array = np.concatenate((img_array, img_array1), axis=1)  # 横向拼接
        #img_array = np.concatenate((img_array, img_array1), axis=0)# 纵向拼接
        img = Image.fromarray(img_array)

# 保存图片
img.save('C:/Users/4334/Desktop/img_path/final1.jpg')
#img.save('C:/Users/4334/Desktop/img_path/final2.jpg')
```

结果图像矩阵表示

结果图像

第一张图像，无需拼接

4. 图片处理 2：

1) 首先根据课件绘制简单验证码图片：创建指定大小的画布，设置为 RGB 模式；创建 Font 对象和 Drew 对象，绘制随机生成的四个字符，并使用 filter 方法进行简单模糊。

2) 进行简单的像素点运算，像素点位置的变换：该操作目的是“复制”上述得到的验证码图片，进行进一步的操作。随机得到 x 的偏移量 offset，重新得到 x 的位置，对 x 的新位置进行判断，由于画布的宽度为 240，因此 x 的新位置只能从 0~239。若小于 0，则将 x 新位置设置为 0；若大于 239，将 x 的新位置设置为 239。

3) 添加干扰噪点像素：随机生成点信息，并给点随机绘制颜色即可。

4) 添加干扰线条：随机生成两个点信息，作为线的起点和终点，并给线随机绘制颜色。

5) 添加干扰弧线：随机生成两个点信息，作为线的起止点，给线随机绘制颜色。

代码及细节解释如下：

库的调用和一些辅助常量和变量的定义：

```
from random import randint, choice
from PIL import Image, ImageDraw, ImageFont, ImageFilter
import string

# 随机字母
characters = string.ascii_letters + string.digits
def rndChar():
    return choice(characters)
# 生成随机颜色
def randomColor(start=0, end=255):
    return randint(start, end), randint(start, end), randint(start, end)
# 生成随机点坐标
def randomPoint():
    # 需要注意他们的取值范围
    return randint(0, img.width), randint(0, img.height)
# 生成两个随机点坐标
def randomPoints():
    return randomPoint(), randomPoint()
```

在绘制干扰线和干扰弧线时生成线的起止点

其他内容（作用见代码注释）：

```
if __name__ == '__main__':
    # 首先创建一个画布，选择RGB模式，设置图片尺寸，颜色为白色
    width = 60 * 4
    height = 60
    img = Image.new(mode="RGB", size=(width, height), color=(255, 255, 255))

# 创建Font对象
myfont = ImageFont.truetype("C:/Users/4334/Desktop/字体文件/kumo.ttf", size=50)

# 创建Draw对象
draw = ImageDraw.Draw(img, mode="RGB")
draw.point((200, 100), fill="black")

# 绘制验证码
total = 4  # 定义验证码长度
part = img.width // (total + 2)
pos = 0  # 相当于指针，指在哪个地方哪个地方就填入对应字符
res = []  # 存放验证码列表
for i in range(total):
    pos += part
    r = rndChar()
    res.append(r)
    draw.text((pos, 3 * img.height // 13), text=r, fill=randomColor(30, 200), font=myfont)
res = ''.join(res)

# 模糊一下
img = img.filter(ImageFilter.BLUR)
```

字体位置高度

```
imgFinal = Image.new(mode="RGB", size=(width, height), color=(255, 255, 255))
pixelsFinal = imgFinal.load()
pixelsTemp = img.load()
# 进行简单的像素点运算，像素点位置的变换
for y in range(0, height):
    offset = randint(-1, 1)
    for x in range(0, width):
        newx = x + offset
        # 边界处理
        if newx < 0:
            newx = 0
        elif newx > 239:
            newx = 239
        pixelsFinal[newx, y] = pixelsTemp[x, y]

draw = ImageDraw.Draw(imgFinal)
# 添加干扰噪点像素
for i in range(int(img.width * img.height * 0.01)):
    draw.point(randomPoint(), randomColor(150))

# 添加干扰线条
for i in range(4):
    draw.line(randomPoints(), fill=randomColor())

# 添加干扰弧线
for i in range(4):
    draw.arc(randomPoints(), 0, randint(0, 180), fill=randomColor())

imgFinal.show()
print(res)  # 列出验证码

with open(r"C:\Users\4334\Desktop\img_path\t.png", "wb") as fp:
    imgFinal.save(fp, format="png")
```

指定图片存放位置及格式

5. （选做）request 库的运用：

根据讲义中的步骤进行每一步操作，运行每一步得到相应的结果即可。

代码及细节解释如下：

调用 python 中解析 xml 的类库 elementTree：

```
# 调用python中解析XML的类库ElementTree加载得到xml文本
import xml.etree.ElementTree as ET
```

其他内容与讲义所提供的相同：

a. 无参数的 GET 请求

```python
# 无参数的get请求
def get_region_country_get():
    # 将url设置为接口的网址
    url = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx/getRegionCountry"
    # 由于该接口不需要进行传参，所以只把url传入
    resp = requests.get(url=url)
    # check the status code
    assert resp.status_code == 200
    text = resp.text
    root = ET.fromstring(text)
    # get the top
    country = root.find('{http://WebXml.com.cn/}string').text
    print(country)
    assert country == '阿尔及利亚,3320'

    # get all
    countries = root.findall('{http://WebXml.com.cn/}string')
    for country in countries:
        print(country.text)

    # check the length of the countries
    assert len(countries) == 79
    print(resp.text)
```

b. 带参数的 GET 请求

```python
# 带参数的GET的请求
def get_support_city_string_get():
    url = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx/getSupportCityString"
    # 根据接口要求，这里要传入地区编码，而地区编码可以通过getRegionProvince接口获得
    resp = requests.get(url=url, params={"theRegionCode": "3113"})
    assert resp.status_code == 200
    text = resp.text
    root = ET.fromstring(text)

    # get the top
    city = root.find('{http://WebXml.com.cn/}string').text
    assert city == '阿城,120'

    # get all
    cities = root.findall('{http://WebXml.com.cn/}string')
    for city in cities:
        print(city.text)
    assert len(cities) == 82
    print(resp.text)
```

c. 带正文（正文格式为 application/x-www-form-urlencoded ）的 POST 请求

```python
# c带正文的POST请求
def get_support_city_string_post():
    url = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx/getSupportCityString"
    resp = requests.post(url=url,
                         headers={"Content-Type": "application/x-www-form-urlencoded"},
                         data={"theRegionCode": "3114"})
    assert resp.status_code == 200
    text = resp.text
    root = ET.fromstring(text)

    # get the top
    city = root.find('{http://WebXml.com.cn/}string').text
    print(city)
    assert city == '安图,658'

    # get all
    cities = root.findall('{http://WebXml.com.cn/}string')
    for city in cities:
        print(city.text)

    assert len(cities) == 50
    print(resp.text)
```

d. 带正文（正文格式为 xml）的 POST 请求

```
# d带正文（正文格式为 xml的）的POST请求
def get_support_city_string_post_xml():
      url = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx"
      data = '''<?xml version="1.0" encoding="utf-8"?>
            <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSc
            <soap:Body>
            <getSupportCityString xmlns="http://WebXml.com.cn/">
            <theRegionCode>3115</theRegionCode>
            </getSupportCityString>
            </soap:Body>
            </soap:Envelope> '''
      resp = requests.post(url=url,
                           data=data,
                           headers={'Content-Type': 'text/xml'}
                           )

      assert resp.status_code == 200
      text = resp.text
      namespaces = {
                  'soap': 'http://schemas.xmlsoap.org/soap/envelope/',
                  'a': 'http://WebXml.com.cn/',
                  }
      root = ET.fromstring(text)

# 获取City的头节点
city = root.find('./soap:Body'
                 '/a:getSupportCityStringResponse'
                 '/a:getSupportCityStringResult'
                 '/a:string',
                 namespaces).text
assert city == '鞍山,724'

# 找到全部City
cities = root.findall('./soap:Body'
                 '/a:getSupportCityStringResponse'
                 '/a:getSupportCityStringResult'
                 '/a:string',
                 namespaces)
for city in cities:
      print(city.text)
assert len(cities) == 55
print(resp.text)
```

e. 不带正文的 POST 请求

```
# e不带正文的POST请求
def get_region_province_post():
      url = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx/getRegionProvince"
      resp = requests.post(url=url)

      assert resp.status_code == 200
      text = resp.text
      root = ET.fromstring(text)

      # get the top
      province = root.find('{http://WebXml.com.cn/}string').text
      print(province)
      assert province == '黑龙江,3113'

      # get all
      provinces = root.findall('{http://WebXml.com.cn/}string')
      for province in provinces:
            print(province.text)

      assert len(provinces) == 35
      print(resp.text)
```

（二）遇到的问题和收获

1. 在进行第一个问题的解决时，初次随机生成球的位置并没有注意球的位置坐标不能超出画布范围，导致点击 display 更新小球有时不能完全显示。通过对小球随机坐标

生成范围的限制，最终得到正确输出。

2. 在进行第三个问题的解决时，一开始会遇到以下警告，但能够成功获得拼接图像。

```
C:/Users/4334/PycharmProjects/pythonProject_test6/test6_3.py:17: DeprecationWarning: ANTIALIAS is deprecated and will be removed in Pillow 10
(2023-07-01). Use Resampling.LANCZOS instead.
  img = img.resize((741, 986), Image.ANTIALIAS)
```

通过资料查询得到解决方案如下：

```
import warnings
warnings.filterwarnings("ignore")
```

程序成功运行，得到拼接图像，且运行不报错。

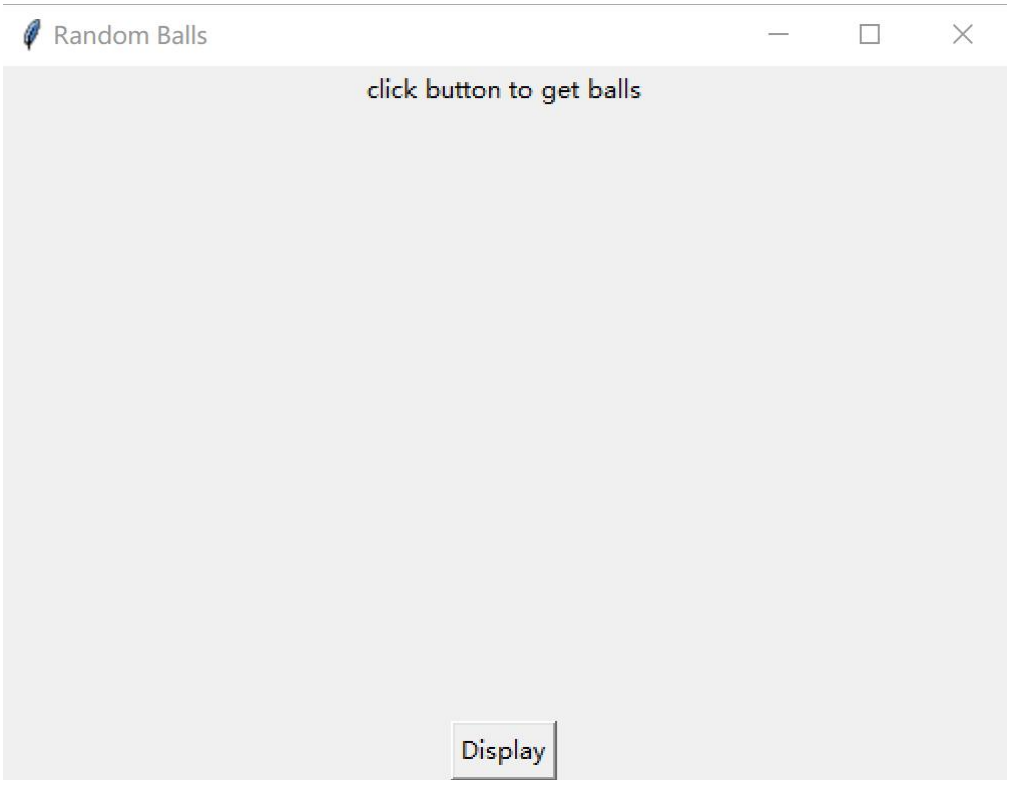
```
C:/Users/4334/PycharmProjects/pythonProject_test6/test6_3.py

进程已结束，退出代码为 0
```

3. 在进行第五题的学习时，根据讲义所提供的代码运行程序，一开始并没有对 ElementTree 进行正确调用，导致程序无法正常运行，通过资料查询加入了对 xml 类库的引用，得以正确运行程序。

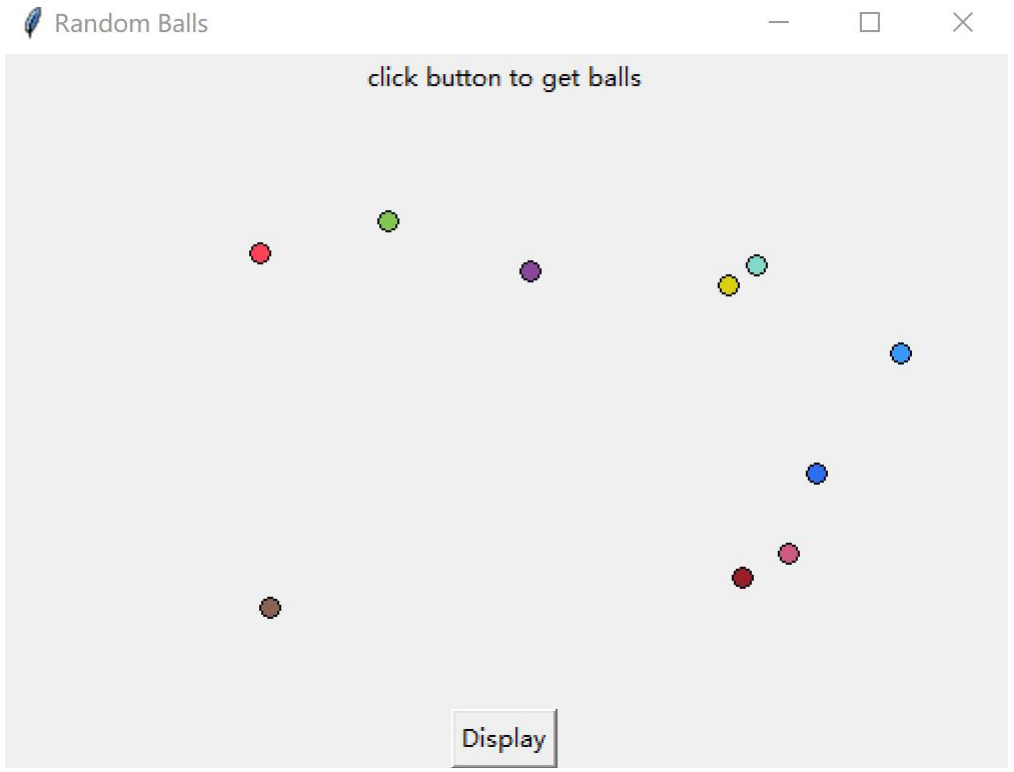
（三）代码运行结果展示

1. 编写 GUI 程序--随机球：

第一次单击前：



下面展示 2 次单击 display 按钮所得到的随机球：

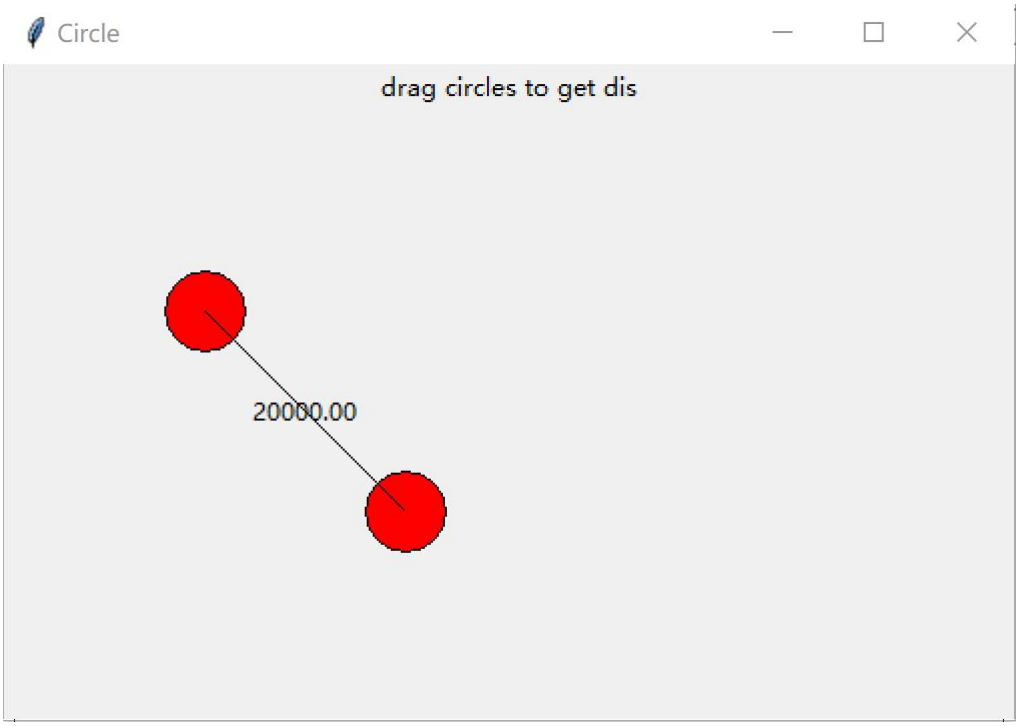


单击关闭结束程序运行：

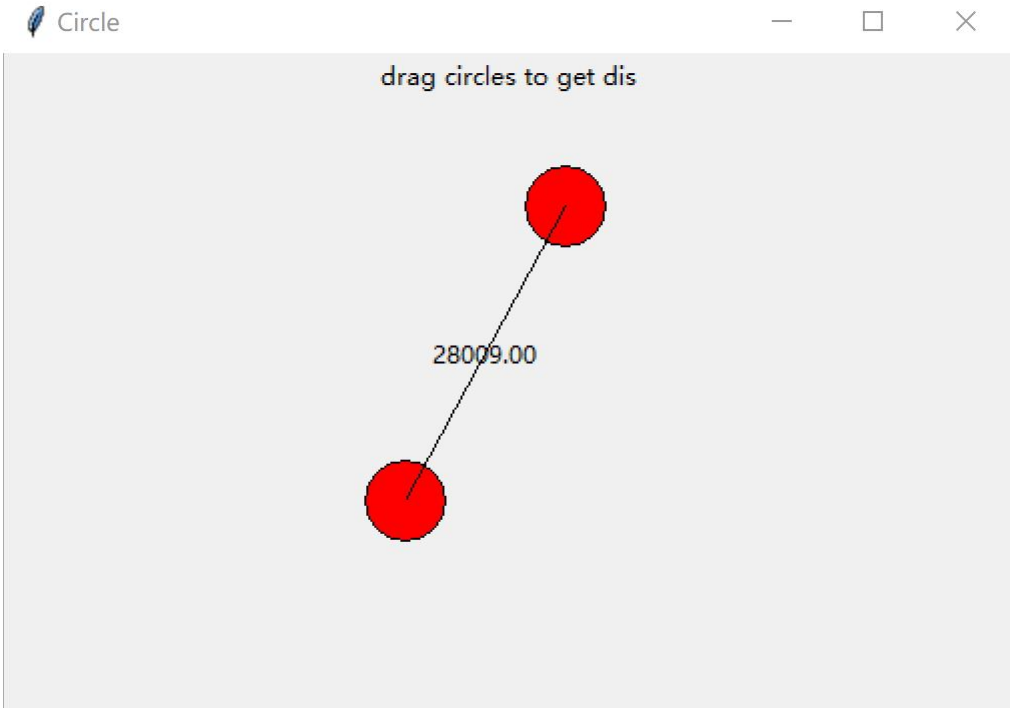
```
C:/Users/4334/PycharmProjects/pythonProject_test6/test6_1.py

进程已结束，退出代码为 0
```

2. 编写 GUI 程序--线连接两个半径为 20 的圆：

启动程序，得到初始状态：

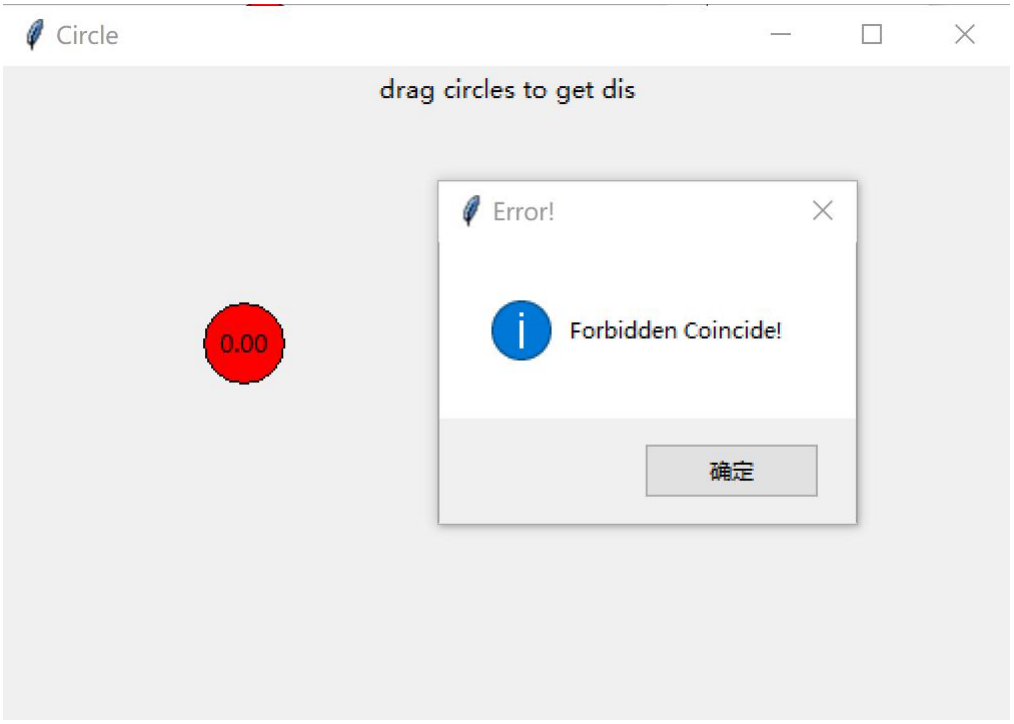


拖动第一个圆，信息更新：

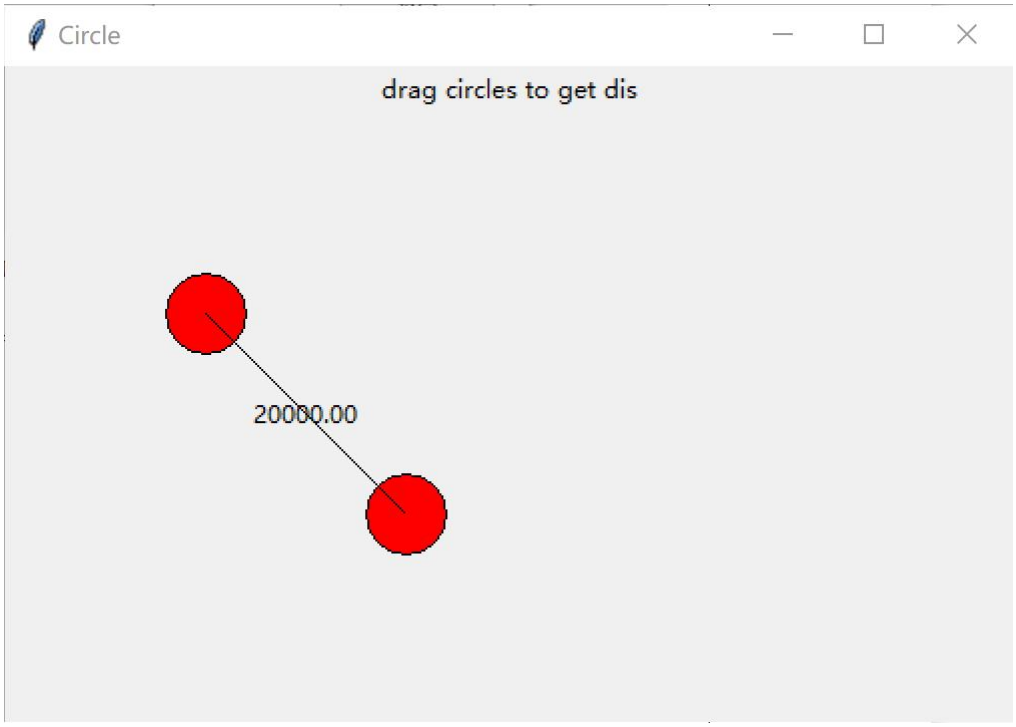


拖动第二个圆，信息更新：

两圆重合，弹出错误信息框：



单击确定或 X 关闭信息框，两圆位置回复初始状态：



单击关闭结束程序的运行：

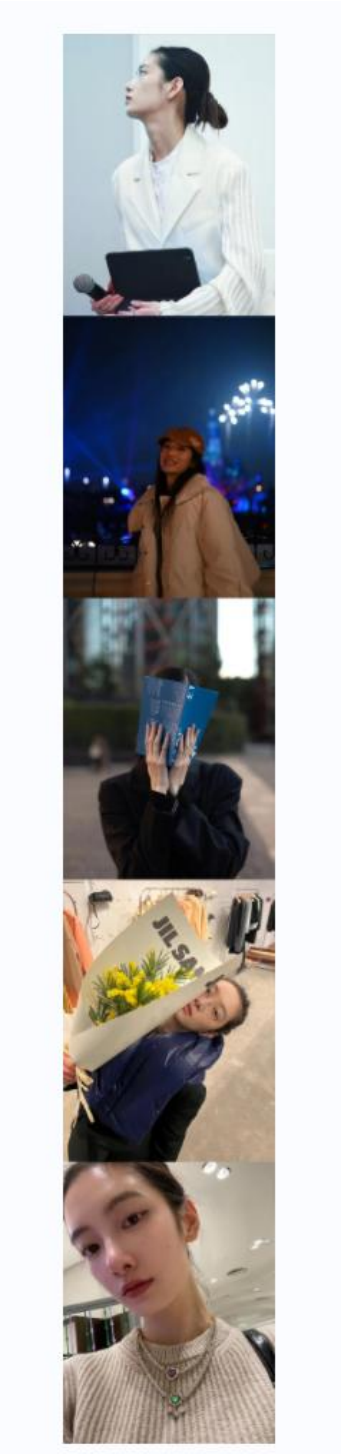
```
C:/Users/4334/PycharmProjects/pythonProject_test6/test6_2.py

进程已结束，退出代码为 0
```

3. 图片处理 1：

未拼接的 5 张图像信息：

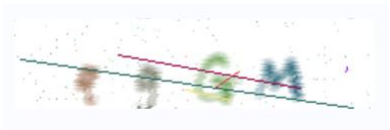
完成水平拼接：

完成垂直拼接：



4. 图片处理 2：

得到一张随机验证码图片：



验证码信息输出：

```
C:/Users/4334/PycharmProjects/pythonProject_test6/test6_4.py
qgGM

进程已结束，退出代码为 0
```

5. （选做）request 库的运用：

（说明：由于每一部分运行结果较长，下面只做部分运行结果展示，完整运行结果以文本文档形式存放在上交的文件中）

a. 无参数的 GET 请求：

```
匈牙利,3234
叙利亚,3174
牙买加,343
伊朗,3171
意大利,3231
印度,3168
印度尼西亚,3152
英国,3247
约旦,3173
越南,3156
智利,3523
<?xml version="1.0" encoding="utf-8"?>
<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://WebXml.com.cn/">
  <string>阿尔及利亚,3320</string>
  <string>阿根廷,3522</string>
  <string>阿曼,3170</string>
  <string>阿塞拜疆,3176</string>
  <string>埃及,3317</string>
  <string>埃塞俄比亚,3314</string>
  <string>爱尔兰,3246</string>
  <string>奥地利,3237</string>
  <string>澳大利亚,368</string>
  <string>巴基斯坦,3169</string>
  <string>巴西,3580</string>
  <string>保加利亚,3232</string>
```

b. 带参数的 GET 请求：

```
伊春,72
依安,67
依兰,96
肇东,89
肇源,117
肇州,115
友谊,3568
<?xml version="1.0" encoding="utf-8"?>
<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://WebXml.com.cn/">
  <string>阿城,120</string>
  <string>安达,88</string>
  <string>巴彦,93</string>
  <string>拜泉,68</string>
  <string>宝清,101</string>
  <string>北安,56</string>
  <string>宾县,121</string>
  <string>勃利,128</string>
  <string>大庆,83</string>
```

c. 带正文的（正文格式为 application/x-www-form-urlencoded ）的 POST 请求：

```
伊通,651
永吉,654
榆树,627
镇赉,110
东辽,3566
抚松,3567
<?xml version="1.0" encoding="utf-8"?>
<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://WebXml.com.cn/">
  <string>安图,658</string>
  <string>白城,107</string>
  <string>白山,737</string>
  <string>长白,740</string>
  <string>长春,650</string>
  <string>长岭,622</string>
  <string>大安,111</string>
  <string>德惠,625</string>
```

d. 带正文（正文格式为 xml 的）POST 请求：

```
营口,775
章党,732
彰武,672
庄河,823
新宾,3565
<?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><getSupportCityStringResponse xmlns="http://WebXml.com
.cn/"><getSupportCityStringResult><string>鞍山,724</string><string>朝阳,3468</string><string>北票,709</string><string>本溪,728</string><string>本溪
县,731</string><string>昌图,674</string><string>长海,822</string><string>东港,3467</string><string>大连,864</string><string>大石桥,
778</string><string>大洼,774</string><string>丹东,784</string><string>灯塔,730</string><string>法库,676</string><string>凤城,783</string><string>抚顺
,733</string><string>阜新,673</string><string>盖州,777</string><string>海城,776</string><string>黑山,720</string><string>葫芦岛,771</string><string>
桓仁,736</string><string>建昌,770</string><string>建平县,712</string><string>金州,819</string><string>锦州,722</string><string>喀左,
714</string><string>开原,680</string><string>康平,675</string><string>宽甸,782</string><string>辽阳,729</string><string>辽阳县,727</string><string>辽
中,717</string><string>凌海,715</string><string>凌源,713</string><string>旅顺,863</string><string>盘山,3469</string><string>盘锦,
723</string><string>普兰店,820</string><string>清原,681</string><string>沈阳,726</string><string>绥中,772</string><string>台安,721</string><string>铁
岭,678</string><string>瓦房店,818</string><string>西丰,679</string><string>新民,718</string><string>兴城,773</string><string>岫岩,
781</string><string>义县,719</string><string>营口,775</string><string>章党,732</string><string>彰武,672</string><string>庄河,823</string><string>新宾
,3565</string></getSupportCityStringResult></getSupportCityStringResponse></soap:Body></soap:Envelope>
```


e. 不带正文的 POST 请求：


```
上海,311102
天津,311103
重庆,311104
香港,311201
澳门,311202
钓鱼岛,311203
<?xml version="1.0" encoding="utf-8"?>
<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://WebXml.com.cn/">
  <string>黑龙江,3113</string>
  <string>吉林,3114</string>
  <string>辽宁,3115</string>
  <string>内蒙古,3116</string>
  <string>河北,3117</string>
  <string>河南,3118</string>
  <string>山东,3119</string>
  <string>山西,31110</string>
```


完整运行结果存放：

四、实验总结

1. 通过该实验，首先对于 GUI 程序的编写有了更深的认识。了解了如何编写程序创建实现相应功能的窗口、如何监听鼠标，更新窗口信息为鼠标信息。
2. 其次了解了如何使用 PIL 对图片进行处理，对图片进行合并、在图片中加入指定元素等方法。
3. 最后是通过对所提供讲义的学习，学会 requests 的简单操作，学会解析 xml 的类库 ElementTree 基本方法的调用。

指导教师批阅意见：

成绩评定：

指导教师签字： 年　　月　　日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。