《多媒体技术》大作业

姓名：张半佛　　　　　完成日期：2019 年 12 月

题目要求：

把视频里面的数据变为表格形式，存储在 csv 中，选择第 1 类题目的，每一个时间段为一行，第一个元素是时间，之后是第一个的名字，对应的值（百分比，绝对值），然后第二个，第三个，有几个写几个。

time,No1_name,Value,No2_name, Value, No2_name, Value

解：

目录

一：核心算法原理

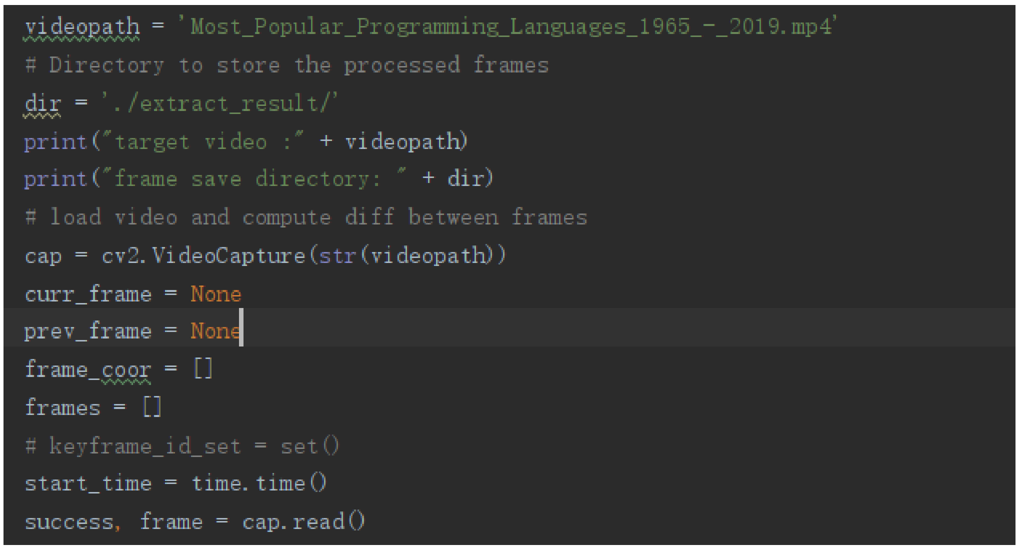
1. 提取关键帧：
   这里采用的方法是提取每一帧的时间，位置在图片的右下角，原图片为1280*720，这里提取【800：1280，560：720】作为该帧的时间图片，之后每一帧的时间图片都与前一帧的时间图片进行相似度比较，相似度差别大的说明该帧是一个季度的第一帧，然后把这张图保存下来。

2. 之后调用百度 OCR 接口，进行识别，这里实验了四种图片进行返回，原图片，宽和高都放大了两倍的图片，二值图像，灰度图，选择识别效果最好的返回。

返回的数据进行处理之后写入 csv 文件。

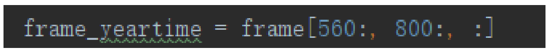
二：详细处理步骤解析

1. 读入视频，将每一帧取出来：



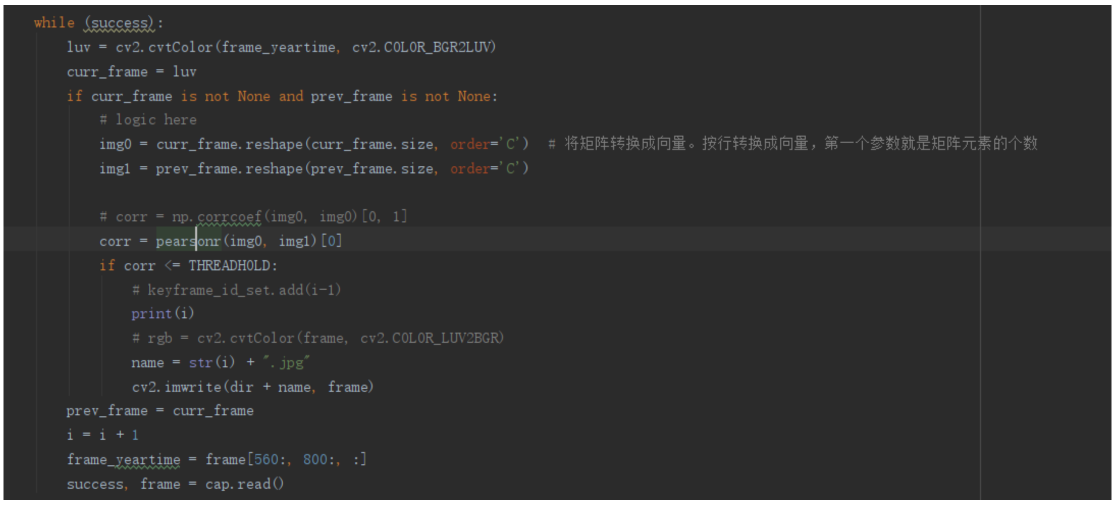
```python
videopath = 'Most_Popular_Programming_Languages_1965_-_2019.mp4'
# Directory to store the processed frames
dir = './extract_result/'
print("target video :" + videopath)
print("frame save directory: " + dir)
# load video and compute diff between frames
cap = cv2.VideoCapture(str(videopath))
curr_frame = None
prev_frame = None
frame_coor = []
frames = []
# keyframe_id_set = set()
start_time = time.time()
success, frame = cap.read()
```

之后取每一张图的右下角时间部分：



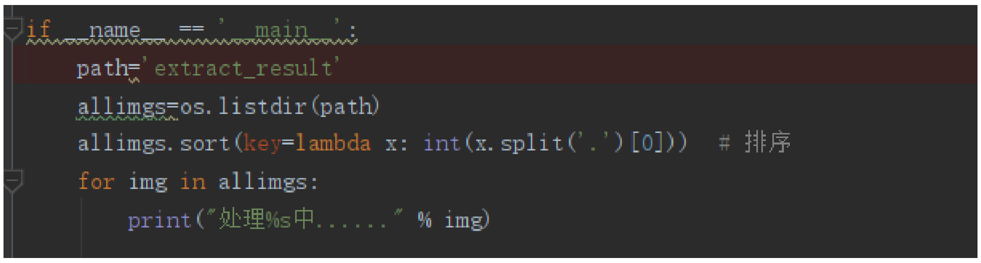
```python
frame_yeartime = frame[560:, 800:, :]
```

最后将相邻两个图片用皮尔逊系数进行比较，设置阈值 0.99，小于此值说明是一个季度的第一帧，然后保存下来：



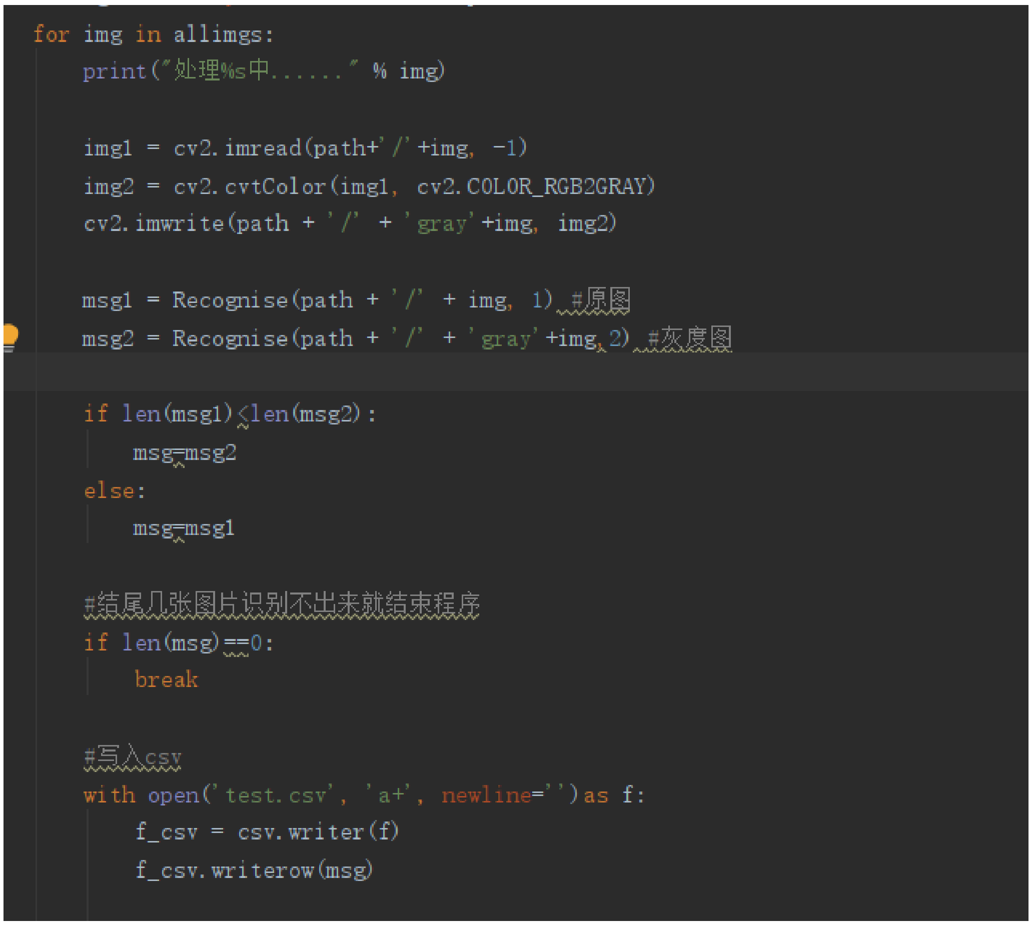
```python
while (success):
    luv = cv2.cvtColor(frame_yeartime, cv2.COLOR_BGR2LUV)
    curr_frame = luv
    if curr_frame is not None and prev_frame is not None:
        # logic here
        img0 = curr_frame.reshape(curr_frame.size, order='C')  # 将矩阵转换成向量。按行转换成向量，第一个参数就是矩阵元素的个数
        img1 = prev_frame.reshape(prev_frame.size, order='C')

        # corr = np.corrcoef(img0, img0)[0, 1]
        corr = pearsonr(img0, img1)[0]
        if corr <= THREADHOLD:
            # keyframe_id_set.add(i-1)
            print(i)
            # rgb = cv2.cvtColor(frame, cv2.COLOR_LUV2BGR)
            name = str(i) + ".jpg"
            cv2.imwrite(dir + name, frame)
    prev_frame = curr_frame
    i = i + 1
    frame_yeartime = frame[560:, 800:, :]
    success, frame = cap.read()
```

2. 首先遍历该文件夹，依次读取图片：



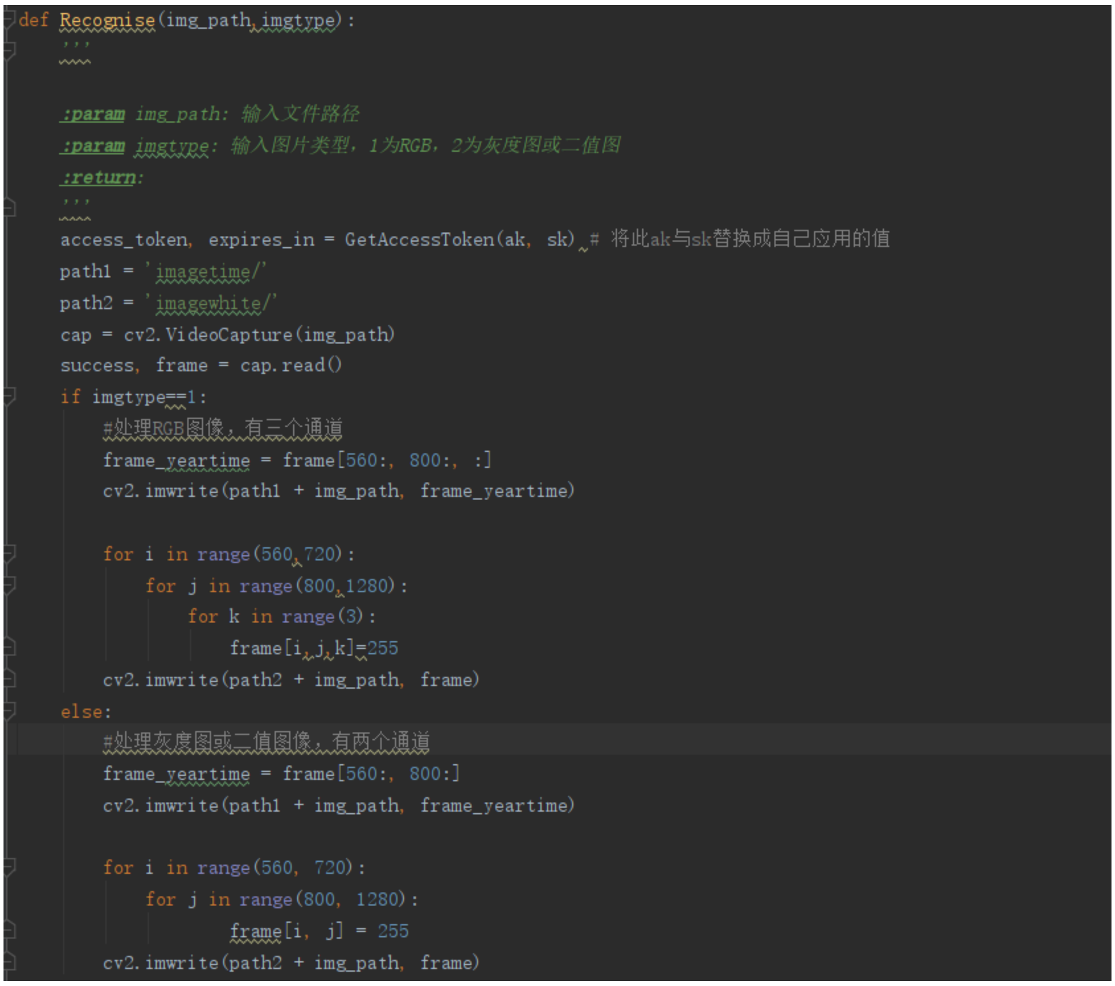
```
if __name__ == '__main__':
    path='extract_result'
    allimgs=os.listdir(path)
    allimgs.sort(key=lambda x: int(x.split('.')[0]))  # 排序
    for img in allimgs:
        print("处理%s中......" % img)
```

将原图像和灰度图像都进行识别，选取返回数量长的字符串进行写入 csv 文件。由于结尾几张图片识别不出来信息，则程序结束。

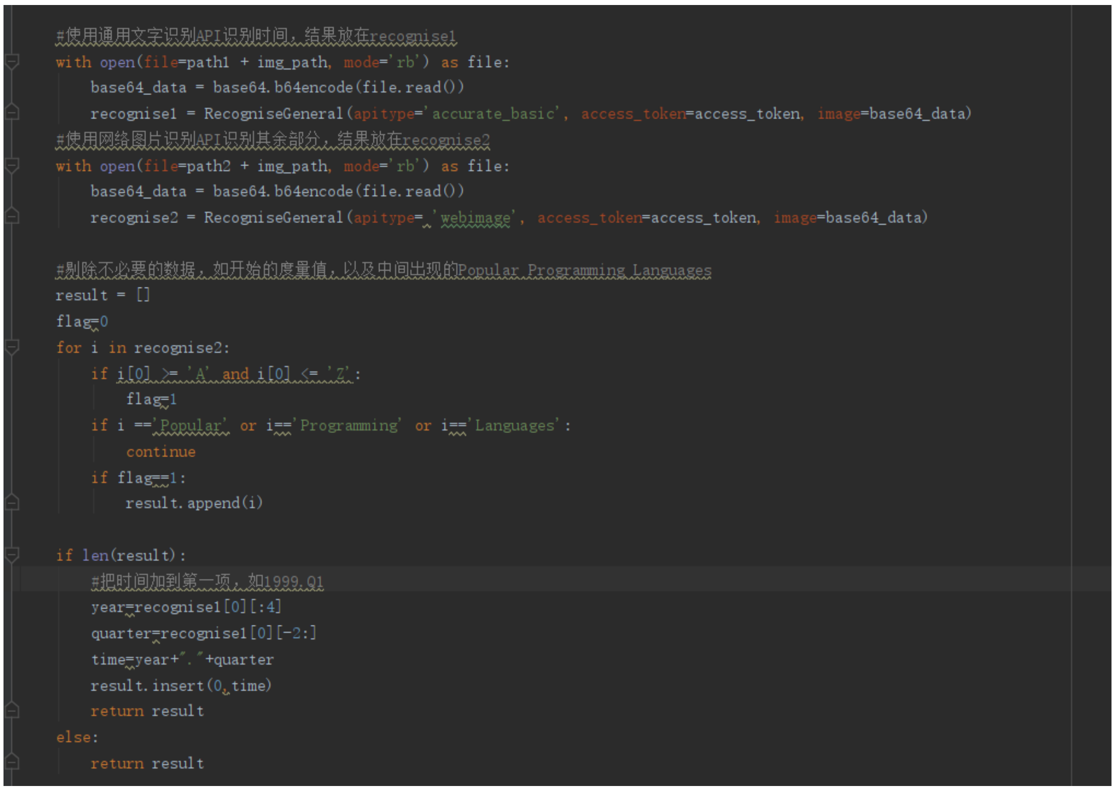


```
for img in allimgs:
    print("处理%s中......" % img)

    img1 = cv2.imread(path+'/'+img, -1)
    img2 = cv2.cvtColor(img1, cv2.COLOR_RGB2GRAY)
    cv2.imwrite(path + '/' + 'gray'+img, img2)

    msg1 = Recognise(path + '/' + img, 1) #原图
    msg2 = Recognise(path + '/' + 'gray'+img, 2) #灰度图

    if len(msg1)<len(msg2):
        msg=msg2
    else:
        msg=msg1

    #结尾几张图片识别不出来就结束程序
    if len(msg)==0:
        break

    #写入csv
    with open('test.csv', 'a+', newline='')as f:
        f_csv = csv.writer(f)
        f_csv.writerow(msg)
```

其中 Recognise 函数如下：



```python
def Recognise(img_path,imgtype):
    '''

    :param img_path: 输入文件路径
    :param imgtype: 输入图片类型，1为RGB，2为灰度图或二值图
    :return:
    '''
    access_token, expires_in = GetAccessToken(ak, sk) # 将此ak与sk替换成自己应用的值
    path1 = 'imagetime/'
    path2 = 'imagewhite/'
    cap = cv2.VideoCapture(img_path)
    success, frame = cap.read()
    if imgtype==1:
        #处理RGB图像，有三个通道
        frame_yeartime = frame[560:, 800:, :]
        cv2.imwrite(path1 + img_path, frame_yeartime)

        for i in range(560,720):
            for j in range(800,1280):
                for k in range(3):
                    frame[i,j,k]=255
        cv2.imwrite(path2 + img_path, frame)
    else:
        #处理灰度图或二值图像，有两个通道
        frame_yeartime = frame[560:, 800:]
        cv2.imwrite(path1 + img_path, frame_yeartime)

        for i in range(560, 720):
            for j in range(800, 1280):
                frame[i, j] = 255
        cv2.imwrite(path2 + img_path, frame)
```



```python
    #使用通用文字识别API识别时间，结果放在recognise1
    with open(file=path1 + img_path, mode='rb') as file:
        base64_data = base64.b64encode(file.read())
        recognise1 = RecogniseGeneral(apitype='accurate_basic', access_token=access_token, image=base64_data)
    #使用网络图片识别API识别其余部分，结果放在recognise2
    with open(file=path2 + img_path, mode='rb') as file:
        base64_data = base64.b64encode(file.read())
        recognise2 = RecogniseGeneral(apitype= 'webimage', access_token=access_token, image=base64_data)

    #剔除不必要的数据，如开始的度量值，以及中间出现的Popular Programming Languages
    result = []
    flag=0
    for i in recognise2:
        if i[0] >= 'A' and i[0] <= 'Z':
            flag=1
        if i =='Popular' or i=='Programming' or i=='Languages':
            continue
        if flag==1:
            result.append(i)

    if len(result):
        #把时间加到第一项，如1999.Q1
        year=recognise1[0][:4]
        quarter=recognise1[0][-2:]
        time=year+"."+quarter
        result.insert(0,time)
        return result
    else:
        return result
```

这里首先调用 GetAccess 来获取接口调用的 access_token 参数以及 token 的

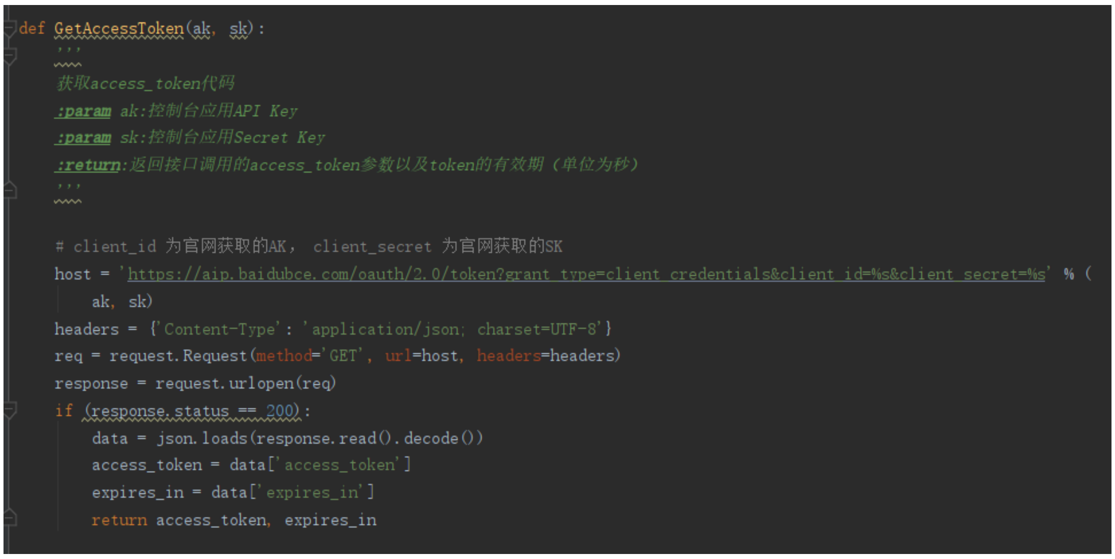
有效期（单位为秒）。

之后将图片变为 base64 编码格式，再调用 RecogniseGeneral 函数进行图片 OCR 识别。结果放在 recognise 里。

之后剔除不必要的数据，如开始的度量值，以及中间出现的 Popular Programming Languages。

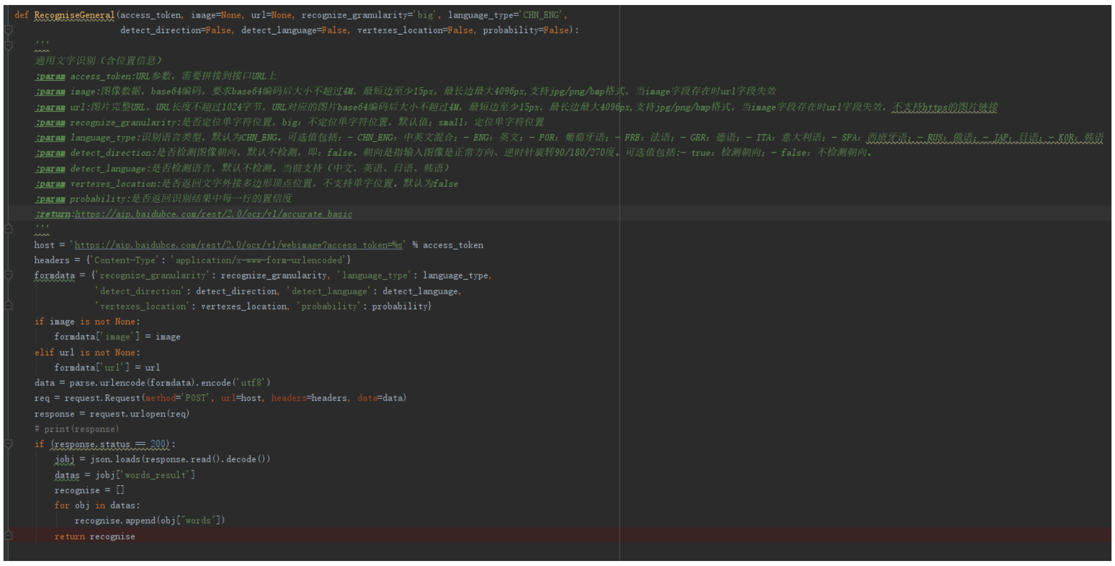
最后把时间放到第一项。

GetAccess 函数如下：



```python
def GetAccessToken(ak, sk):
    '''
    获取access_token代码
    :param ak:控制台应用API Key
    :param sk:控制台应用Secret Key
    :return:返回接口调用的access_token参数以及token的有效期（单位为秒）
    '''

    # client_id 为官网获取的AK， client_secret 为官网获取的SK
    host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=%s&client_secret=%s' % (
        ak, sk)
    headers = {'Content-Type': 'application/json; charset=UTF-8'}
    req = request.Request(method='GET', url=host, headers=headers)
    response = request.urlopen(req)
    if (response.status == 200):
        data = json.loads(response.read().decode())
        access_token = data['access_token']
        expires_in = data['expires_in']
        return access_token, expires_in
```

RecogniseGeneral 函数如下：



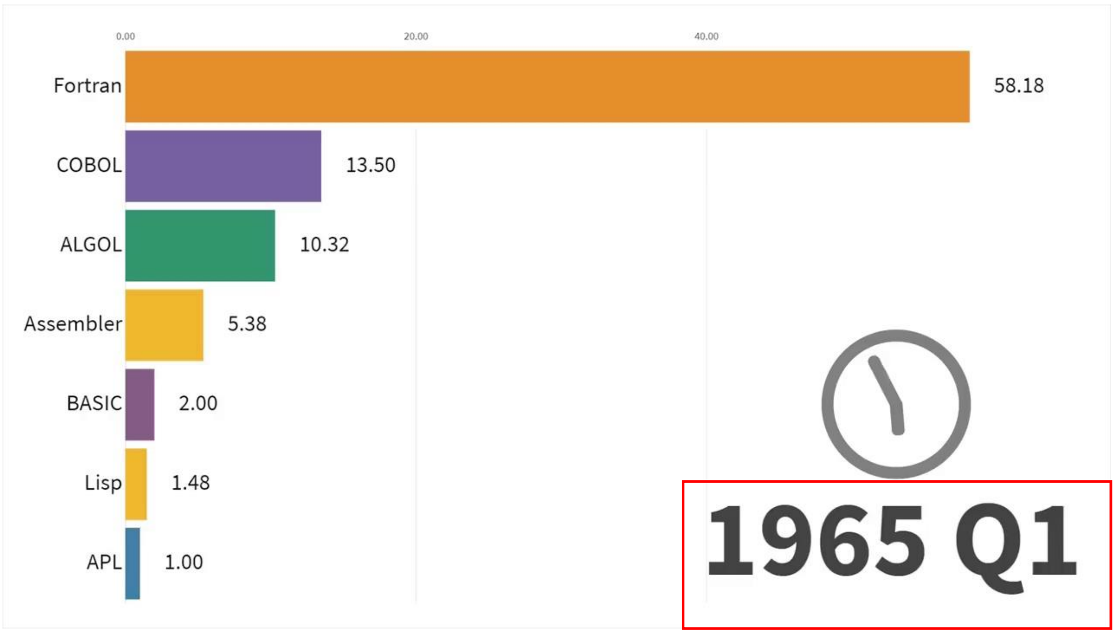
```python
def RecogniseGeneral(access_token, image=None, url=None, recognize_granularity='big', language_type='CHN_ENG',
                     detect_direction=False, detect_language=False, vertexes_location=False, probability=False):
    '''
    通用文字识别（含位置信息）
    :param access_token:URL参数，需要拼接到接口URL上
    :param image:图像数据，base64编码，要求base64编码后大小不超过4M，最短边至少15px，最长边最大4096px,支持jpg/png/bmp格式，当image字段存在时url字段失效
    :param url:图片完整URL，URL长度不超过1024字节，URL对应的图片base64编码后大小不超过4M，最短边至少15px，最长边最大4096px,支持jpg/png/bmp格式，当image字段存在时url字段失效，不支持https的图片链接
    :param recognize_granularity:是否定位单字符位置，big：不定位单字符位置，默认值；small：定位单字符位置
    :param language_type:识别语言类型，默认为CHN_ENG。可选值包括：- CHN_ENG：中英文混合；- ENG：英文；- POR：葡萄牙语；- FRE：法语；- GER：德语；- ITA：意大利语；- SPA：西班牙语；- RUS：俄语；- JAP：日语；- KOR：韩语
    :param detect_direction:是否检测图像朝向，默认不检测，即：false。朝向是指输入图像是正常方向、逆时针旋转90/180/270度。可选值包括:- true：检测朝向；- false：不检测朝向。
    :param detect_language:是否检测语言，默认不检测。当前支持（中文、英语、日语、韩语）
    :param vertexes_location:是否返回文字外接多边形顶点位置，不支持单字位置。默认为false
    :param probability:是否返回识别结果中每一行的置信度
    :return:https://aip.baidubce.com/rest/2.0/ocr/v1/accurate_basic
    '''
    host = 'https://aip.baidubce.com/rest/2.0/ocr/v1/webimage?access_token=%s' % access_token
    headers = {'Content-Type': 'application/x-www-form-urlencoded'}
    formdata = {'recognize_granularity': recognize_granularity, 'language_type': language_type,
                'detect_direction': detect_direction, 'detect_language': detect_language,
                'vertexes_location': vertexes_location, 'probability': probability}
    if image is not None:
        formdata['image'] = image
    elif url is not None:
        formdata['url'] = url
    data = parse.urlencode(formdata).encode('utf8')
    req = request.Request(method='POST', url=host, headers=headers, data=data)
    response = request.urlopen(req)
    # print(response)
    if (response.status == 200):
        jobj = json.loads(response.read().decode())
        datas = jobj['words_result']
        recognise = []
        for obj in datas:
            recognise.append(obj["words"])
        return recognise
```

返回的是识别结果。

三：实验问题

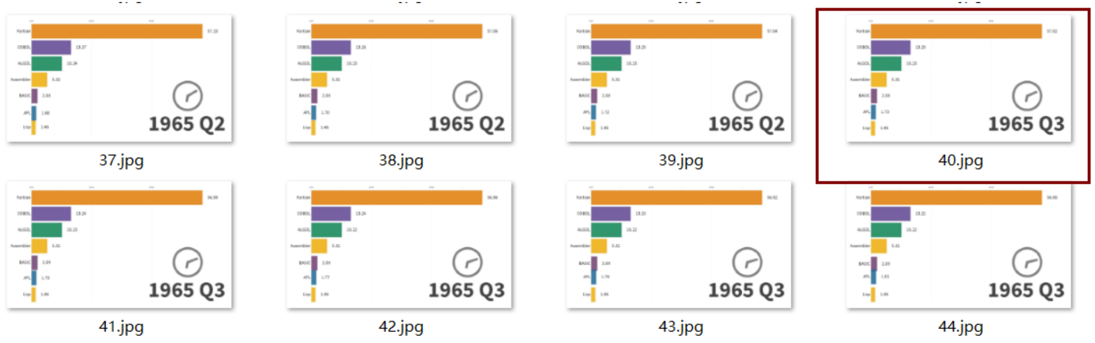
1. **如何每个季度都取到一个帧。**

这里采用的解决办法是由于每一帧图片大小为 1280*720，可以提取每一帧图片的右下角【800：1280，560：720】这一块，这一块表示的是时间。如下图所示：

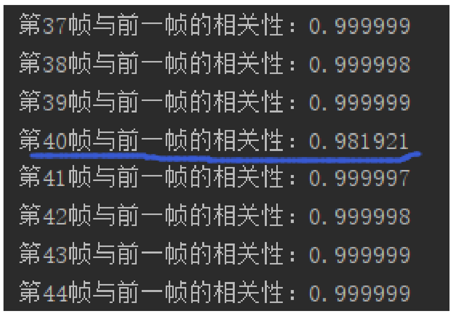


然后计算当前这一块区域与上一帧此块区域的相关性，这里采用的是皮尔逊系数。

以下面几张图为例（37 帧到 44 帧）：



计算相关性：

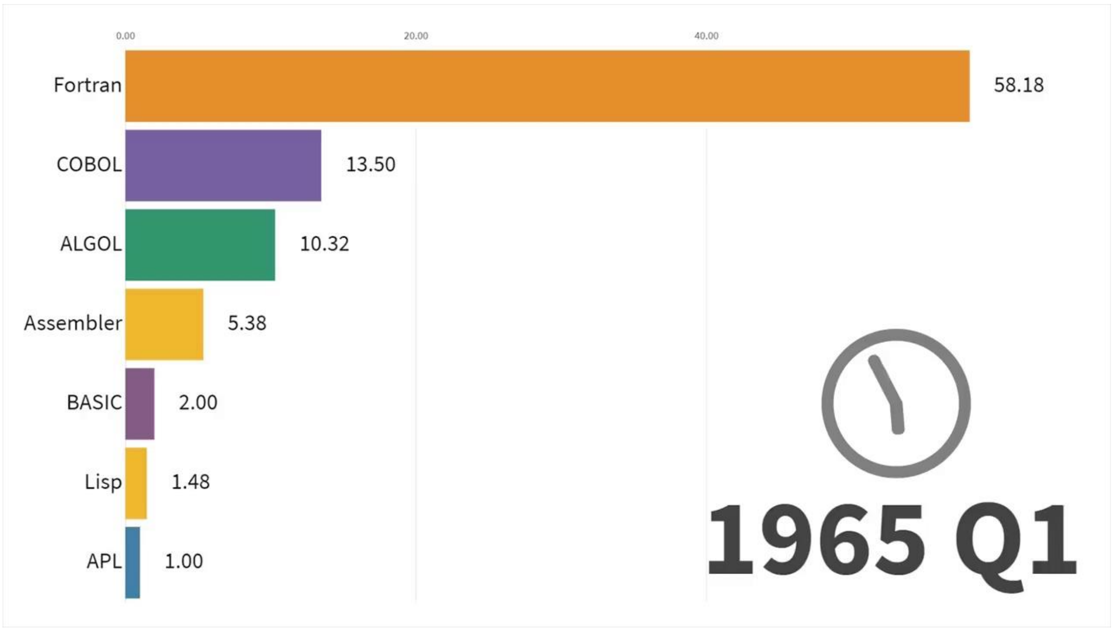


```
第37帧与前一帧的相关性：0.999999
第38帧与前一帧的相关性：0.999998
第39帧与前一帧的相关性：0.999999
第40帧与前一帧的相关性：0.981921
第41帧与前一帧的相关性：0.999997
第42帧与前一帧的相关性：0.999998
第43帧与前一帧的相关性：0.999999
第44帧与前一帧的相关性：0.999999
```

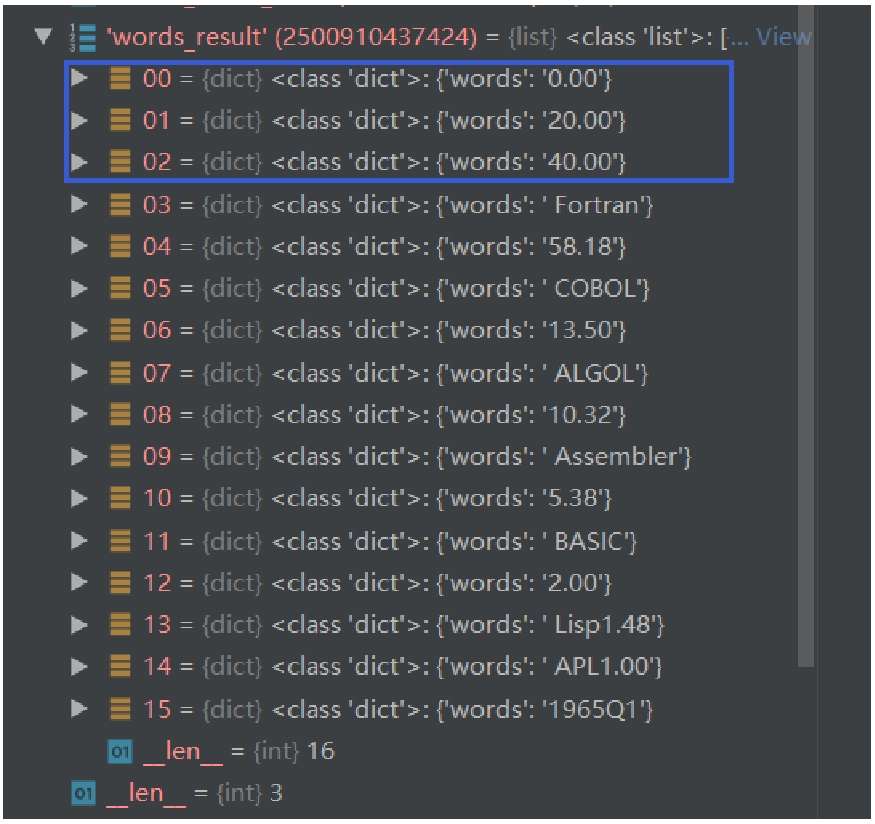
可以明显看到，每一季度的第一帧与前面帧的相关性较小，所以可以取阈值 0.99，当此帧的相关性小于阈值时，说明这一帧是一个季度的开始帧，将其保存下来。

这样就完成了关键帧的选取。

2. **处理多余数据。**
   如下面一张图：

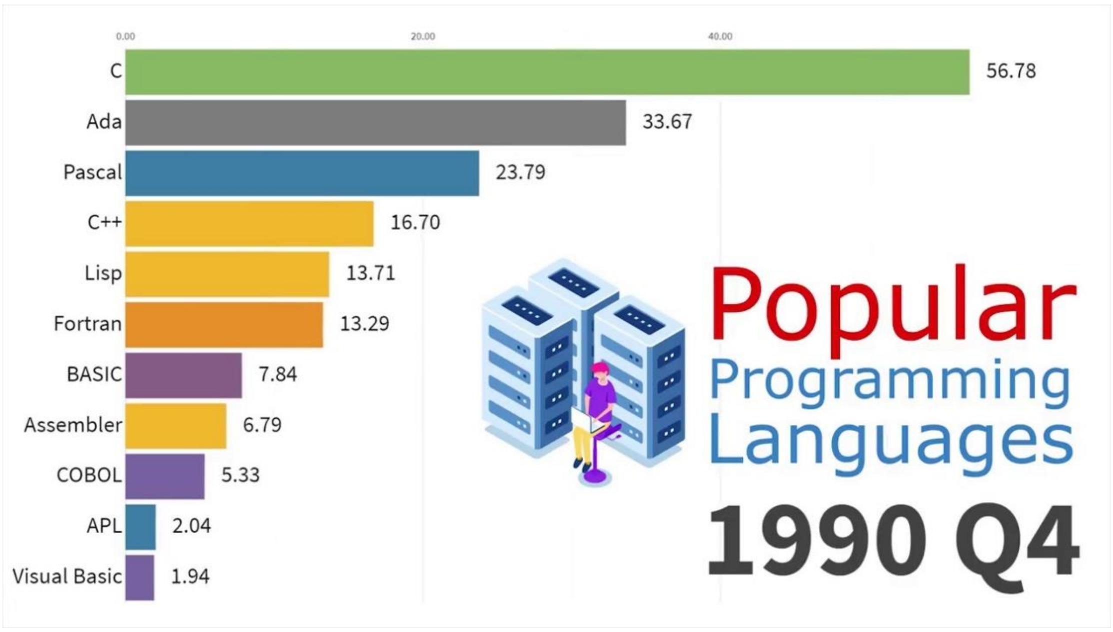


调用百度 OCR 之后，返回的结果是：



```
▼ 'words_result' (2500910437424) = {list} <class 'list'>: [... View
    ▶ 00 = {dict} <class 'dict'>: {'words': '0.00'}
    ▶ 01 = {dict} <class 'dict'>: {'words': '20.00'}
    ▶ 02 = {dict} <class 'dict'>: {'words': '40.00'}
    ▶ 03 = {dict} <class 'dict'>: {'words': ' Fortran'}
    ▶ 04 = {dict} <class 'dict'>: {'words': '58.18'}
    ▶ 05 = {dict} <class 'dict'>: {'words': ' COBOL'}
    ▶ 06 = {dict} <class 'dict'>: {'words': '13.50'}
    ▶ 07 = {dict} <class 'dict'>: {'words': ' ALGOL'}
    ▶ 08 = {dict} <class 'dict'>: {'words': '10.32'}
    ▶ 09 = {dict} <class 'dict'>: {'words': ' Assembler'}
    ▶ 10 = {dict} <class 'dict'>: {'words': '5.38'}
    ▶ 11 = {dict} <class 'dict'>: {'words': ' BASIC'}
    ▶ 12 = {dict} <class 'dict'>: {'words': '2.00'}
    ▶ 13 = {dict} <class 'dict'>: {'words': ' Lisp1.48'}
    ▶ 14 = {dict} <class 'dict'>: {'words': ' APL1.00'}
    ▶ 15 = {dict} <class 'dict'>: {'words': '1965Q1'}
      __len__ = {int} 16
    __len__ = {int} 3
```

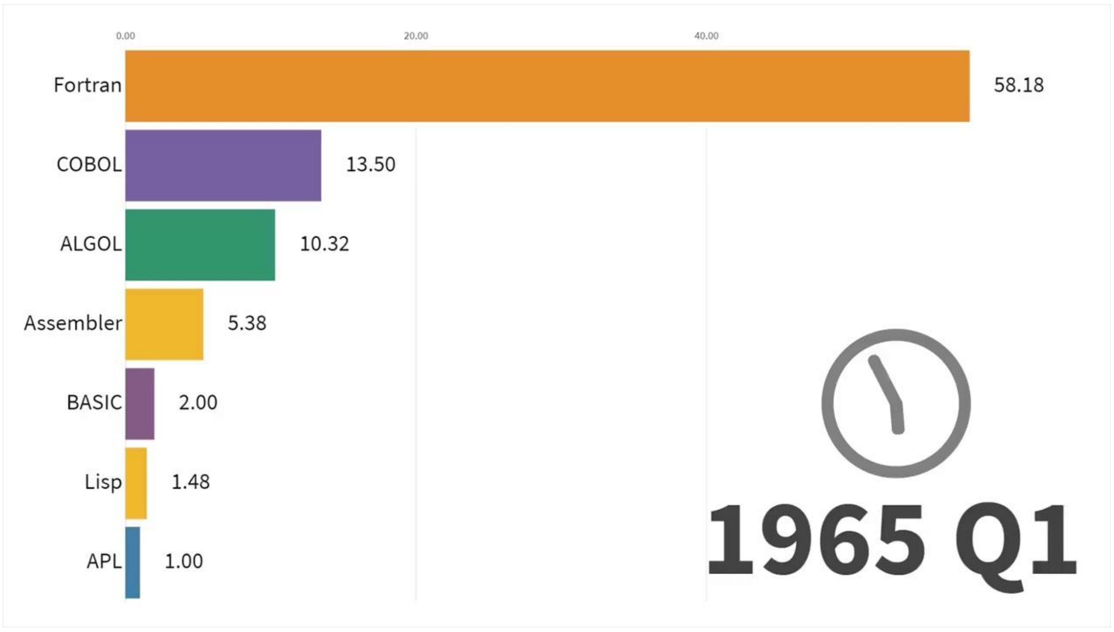
多余数据包括开始的度量值，以及后面有些帧出现的 Popular Programming Languages 这三个词语：



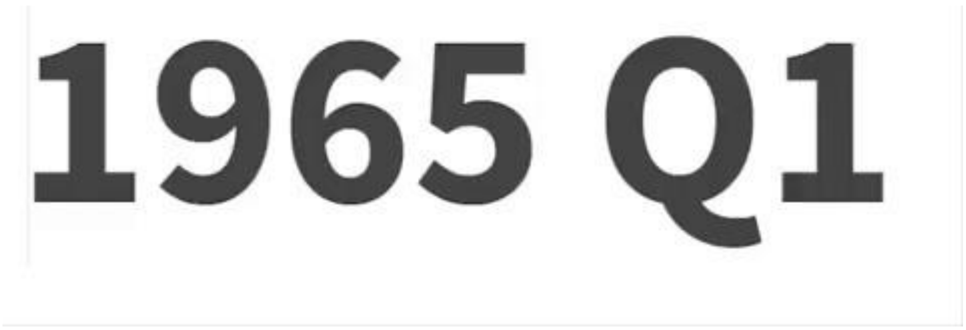
针对度量值：可以循环遍历返回的数据，当某一项为字母时，说明前面都是度量值，从此处开始保存数据即可。

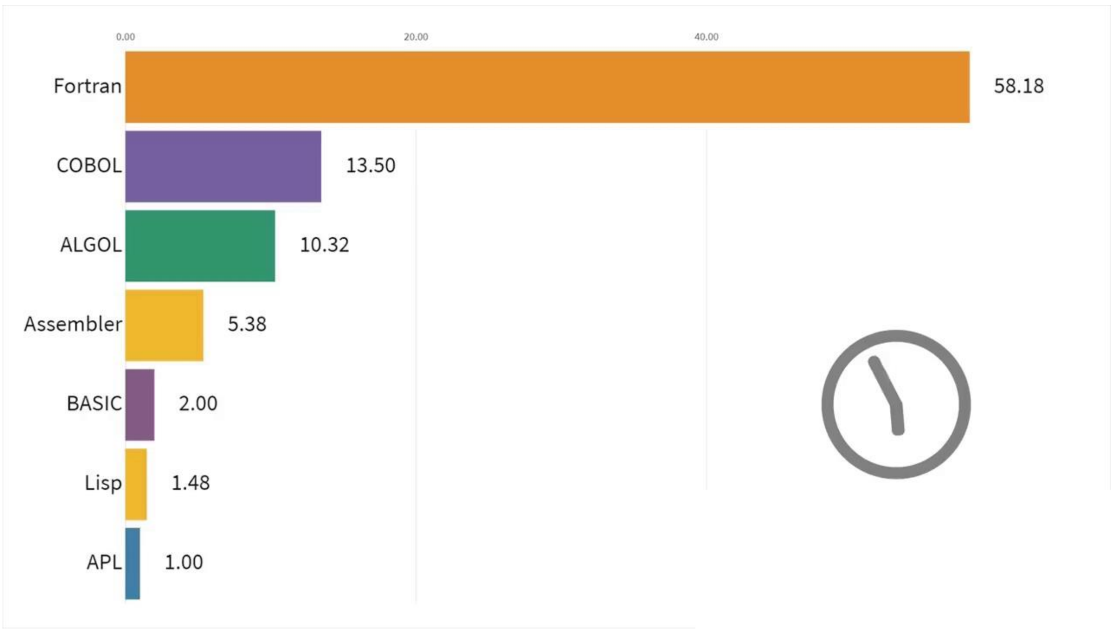
针对 Popular Programming Languages 这三个词语：可以直接在代码中删去。

3. **把时间提到第一位。**
由于不同图片返回时间的位置不一样，有的是最后一项返回时间，有的则在倒数第二项返回时间。
这里采用的解决方法是，先把图片右下角的时间进行识别。之后把该区域补全为白色，即把一张图片分开识别：
原图：



分开识别：

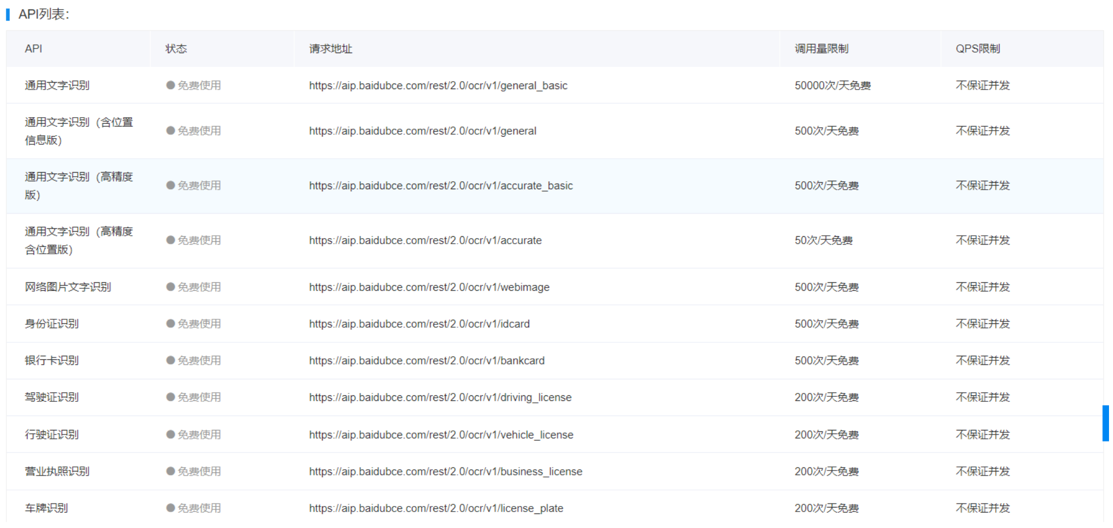




最后把第一个识别的结果加上第二个识别的结果。
这样可以保证时间能够正确放到第一位。

4. 百度 OCR 识别结果的准确性。

调用百度 OCR 的时候识别结果不能保证百分之百正确。

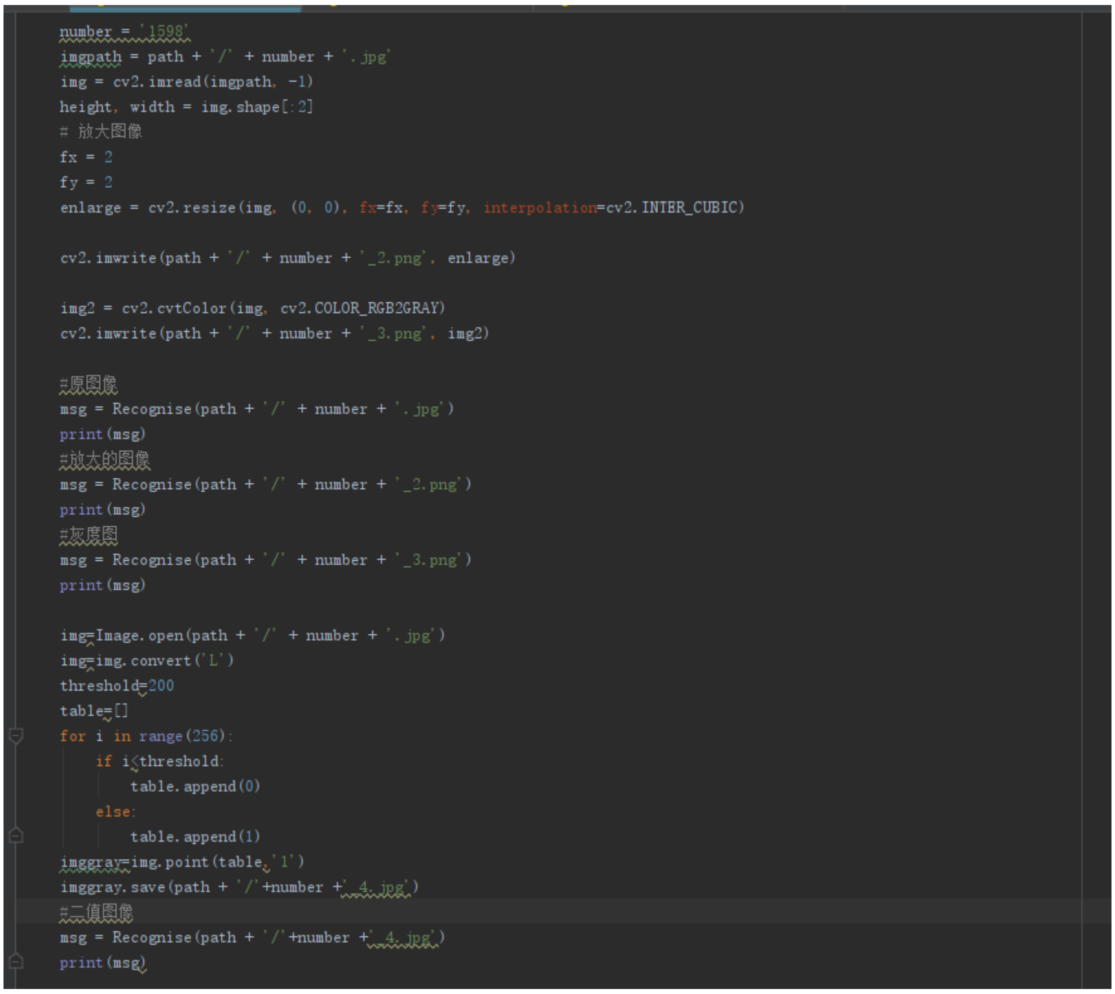

API列表:

API	状态	请求地址	调用量限制	QPS限制
通用文字识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/general_basic	50000次/天免费	不保证并发
通用文字识别（含位置信息版）	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/general	500次/天免费	不保证并发
通用文字识别（高精度版）	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/accurate_basic	500次/天免费	不保证并发
通用文字识别（高精度含位置版）	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/accurate	50次/天免费	不保证并发
网络图片文字识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/webimage	500次/天免费	不保证并发
身份证识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/idcard	500次/天免费	不保证并发
银行卡识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/bankcard	500次/天免费	不保证并发
驾驶证识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/driving_license	200次/天免费	不保证并发
行驶证识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/vehicle_license	200次/天免费	不保证并发
营业执照识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/business_license	200次/天免费	不保证并发
车牌识别	● 免费使用	https://aip.baidubce.com/rest/2.0/ocr/v1/license_plate	200次/天免费	不保证并发

可以使用如上接口，比如通用文字识别，高精度版等等，经过实验发现，有一个网络图片文字识别的 api 效果相比于其他是最好的。

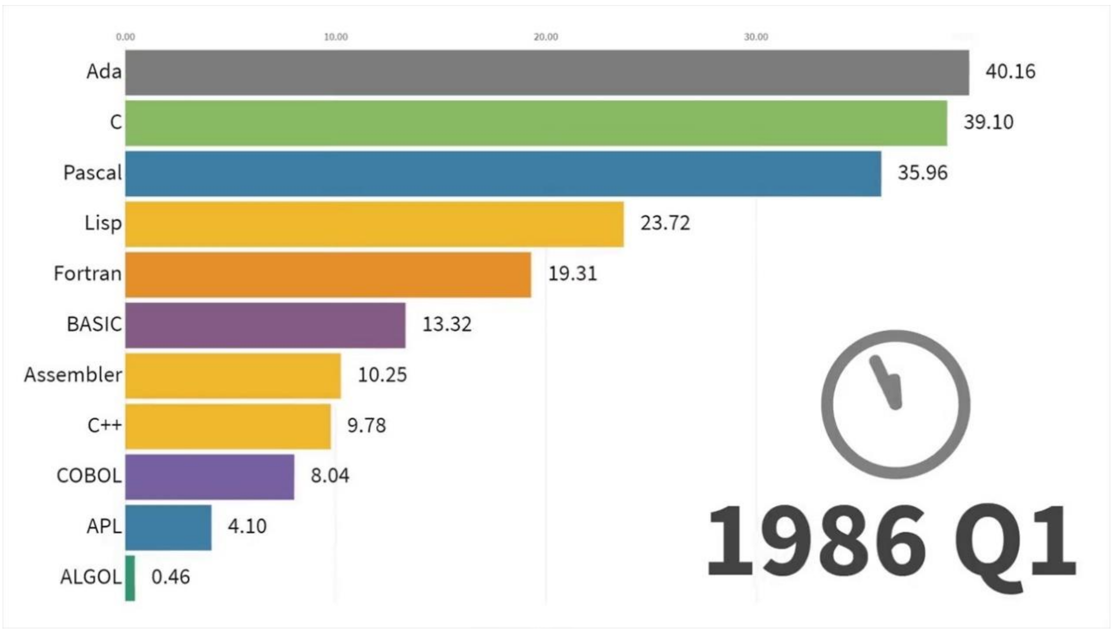
但是还是有问题的，比如有些图片的 C,R 两个字符识别不出来，但是将其转化为灰度图则能识别出来。
这里测试时使用了四种图片进行识别，原图片，宽和高都放大了两倍的图片，

二值图像，灰度图，发现有时候原图像效果最好，有时候灰度图效果最好，所以我们采用的方法是，将原图像和灰度图像都进行识别，选择识别结果的多的那组，识别的更全。

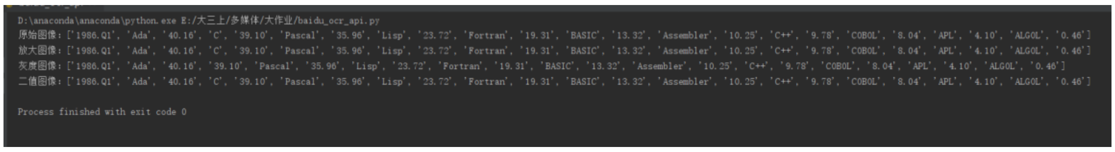


```python
number = '1598'
imgpath = path + '/' + number + '.jpg'
img = cv2.imread(imgpath, -1)
height, width = img.shape[:2]
# 放大图像
fx = 2
fy = 2
enlarge = cv2.resize(img, (0, 0), fx=fx, fy=fy, interpolation=cv2.INTER_CUBIC)

cv2.imwrite(path + '/' + number + '_2.png', enlarge)

img2 = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
cv2.imwrite(path + '/' + number + '_3.png', img2)

#原图像
msg = Recognise(path + '/' + number + '.jpg')
print(msg)
#放大的图像
msg = Recognise(path + '/' + number + '_2.png')
print(msg)
#灰度图
msg = Recognise(path + '/' + number + '_3.png')
print(msg)

img=Image.open(path + '/' + number + '.jpg')
img=img.convert('L')
threshold=200
table=[]
for i in range(256):
    if i<threshold:
        table.append(0)
    else:
        table.append(1)
imggray=img.point(table,'1')
imggray.save(path + '/'+number +'_4.jpg')
#二值图像
msg = Recognise(path + '/'+number +'_4.jpg')
print(msg)
```

比如如下图片：

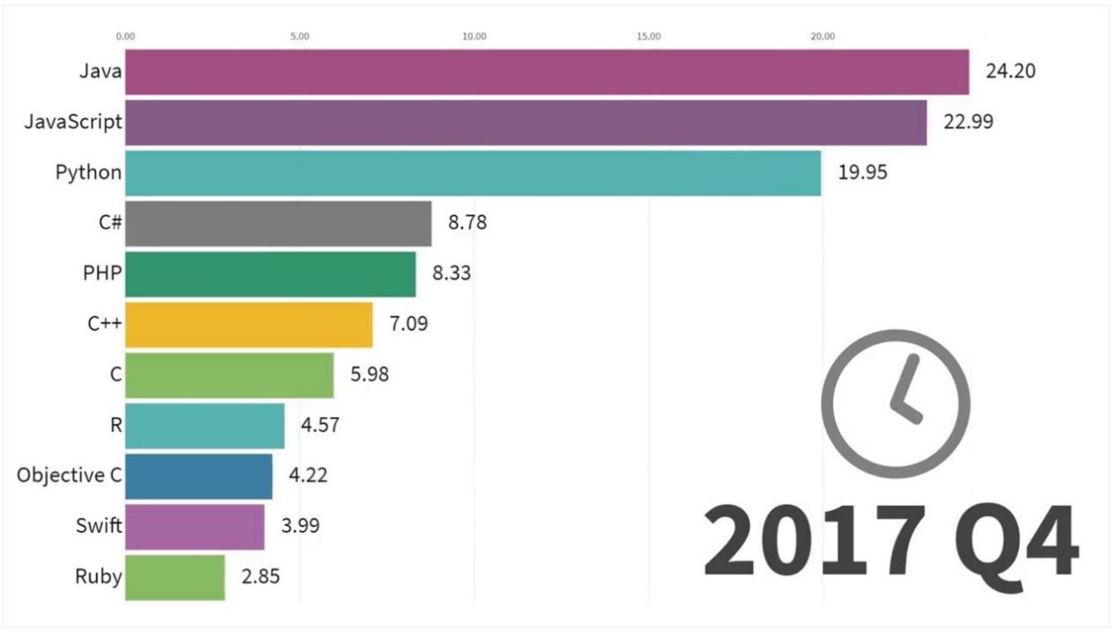


识别结果：

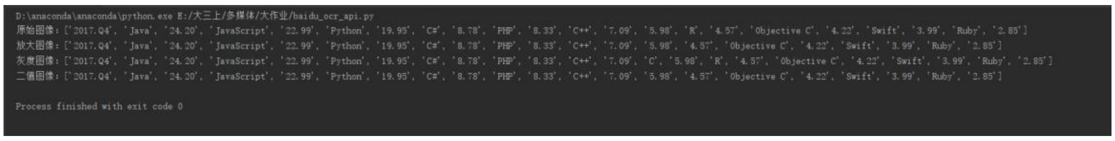

```
D:\anaconda\anaconda\python.exe E:/大三上/多媒体/大作业/baidu_ocr_api.py
原始图像：['1986.Q1', 'Ada', '40.16', 'C', '39.10', 'Pascal', '35.96', 'Lisp', '23.72', 'Fortran', '19.31', 'BASIC', '13.32', 'Assembler', '10.25', 'C++', '9.78', 'COBOL', '8.04', 'APL', '4.10', 'ALGOL', '0.46']
放大图像：['1986.Q1', 'Ada', '40.16', 'C', '39.10', 'Pascal', '35.96', 'Lisp', '23.72', 'Fortran', '19.31', 'BASIC', '13.32', 'Assembler', '10.25', 'C++', '9.78', 'COBOL', '8.04', 'APL', '4.10', 'ALGOL', '0.46']
灰度图像：['1986.Q1', 'Ada', '40.16', '39.10', 'Pascal', '35.96', 'Lisp', '23.72', 'Fortran', '19.31', 'BASIC', '13.32', 'Assembler', '10.25', 'C++', '9.78', 'COBOL', '8.04', 'APL', '4.10', 'ALGOL', '0.46']
二值图像：['1986.Q1', 'Ada', '40.16', 'C', '39.10', 'Pascal', '35.96', 'Lisp', '23.72', 'Fortran', '19.31', 'BASIC', '13.32', 'Assembler', '10.25', 'C++', '9.78', 'COBOL', '8.04', 'APL', '4.10', 'ALGOL', '0.46']

Process finished with exit code 0
```

可以看到原始图像比灰度图识别效果好。

对于如下图片：



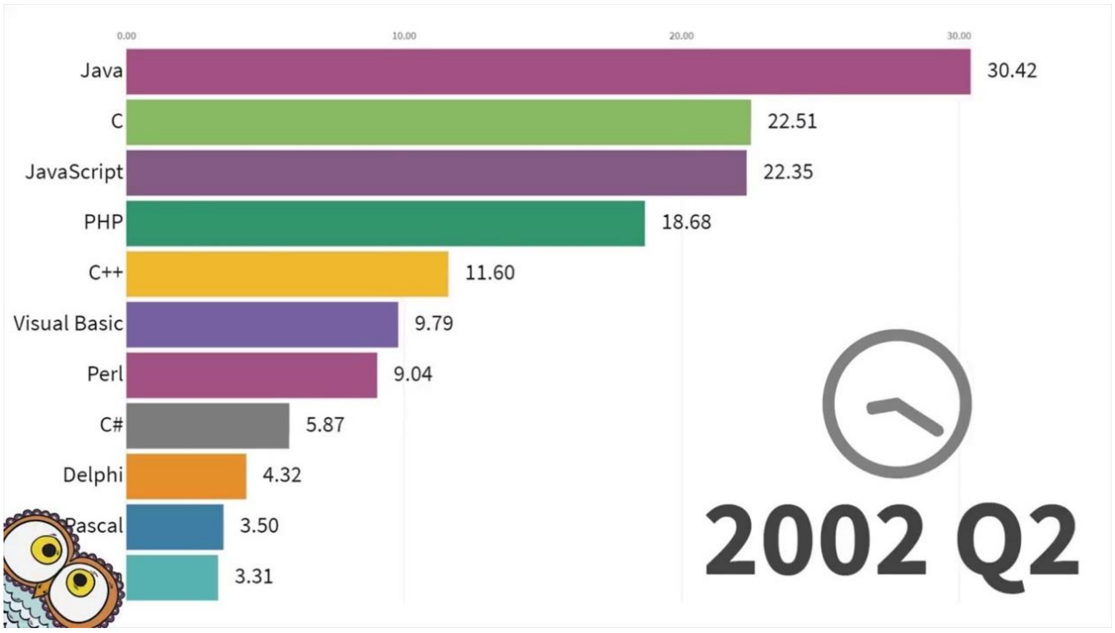
识别结果：



```
D:\anaconda\anaconda\python.exe E:/大三上/多媒体/大作业/baidu_ocr_api.py
原始图像：['2017.Q4', 'Java', '24.20', 'JavaScript', '22.99', 'Python', '19.95', 'C#', '8.78', 'PHP', '8.33', 'C++', '7.09', '5.98', 'R', '4.57', 'Objective C', '4.22', 'Swift', '3.99', 'Ruby', '2.85']
放大图像：['2017.Q4', 'Java', '24.20', 'JavaScript', '22.99', 'Python', '19.95', 'C#', '8.78', 'PHP', '8.33', 'C++', '7.09', '5.98', '4.57', 'Objective C', '4.22', 'Swift', '3.99', 'Ruby', '2.85']
灰度图像：['2017.Q4', 'Java', '24.20', 'JavaScript', '22.99', 'Python', '19.95', 'C#', '8.78', 'PHP', '8.33', 'C++', '7.09', 'C', '5.98', 'R', '4.57', 'Objective C', '4.22', 'Swift', '3.99', 'Ruby', '2.85']
二值图像：['2017.Q4', 'Java', '24.20', 'JavaScript', '22.99', 'Python', '19.95', 'C#', '8.78', 'PHP', '8.33', 'C++', '7.09', '5.98', '4.57', 'Objective C', '4.22', 'Swift', '3.99', 'Ruby', '2.85']

Process finished with exit code 0
```

可以看到灰度图比原始图识别效果好。

5. **有些图片被遮挡。**

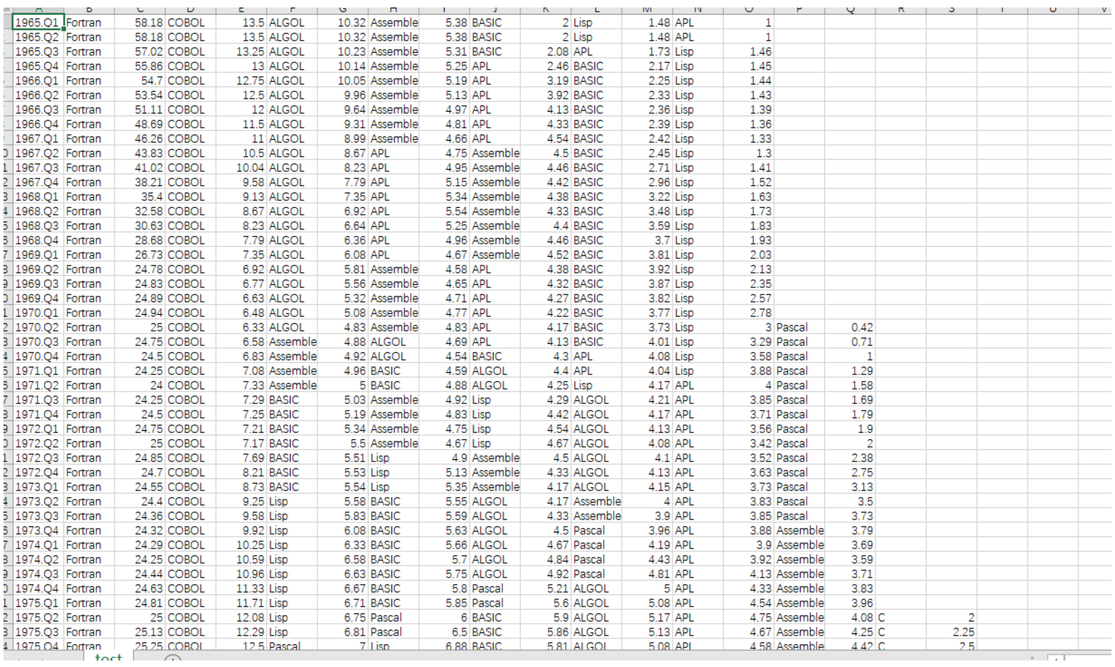


如这个图片，左下角被猫头鹰遮挡，而且整个季度都在遮挡，这样的图片是没有办法的。只能采用手动修改最后的数据补全。

不过 219 个季度，只有 3 个季度被遮挡了，修改还是比较方便的。

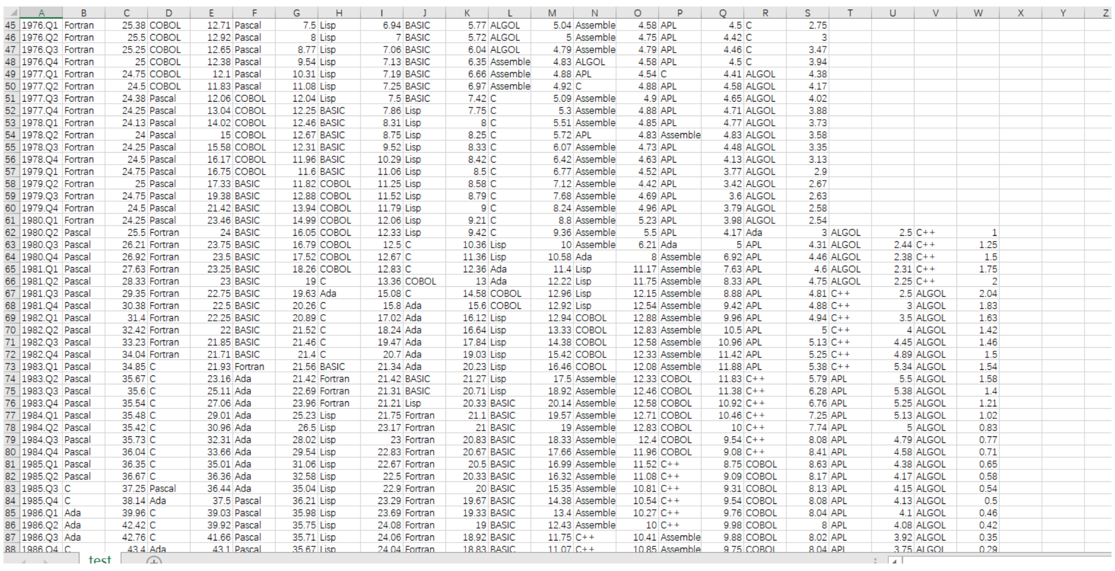
四：实验结果分析：

存储到 scv 文件，如下：



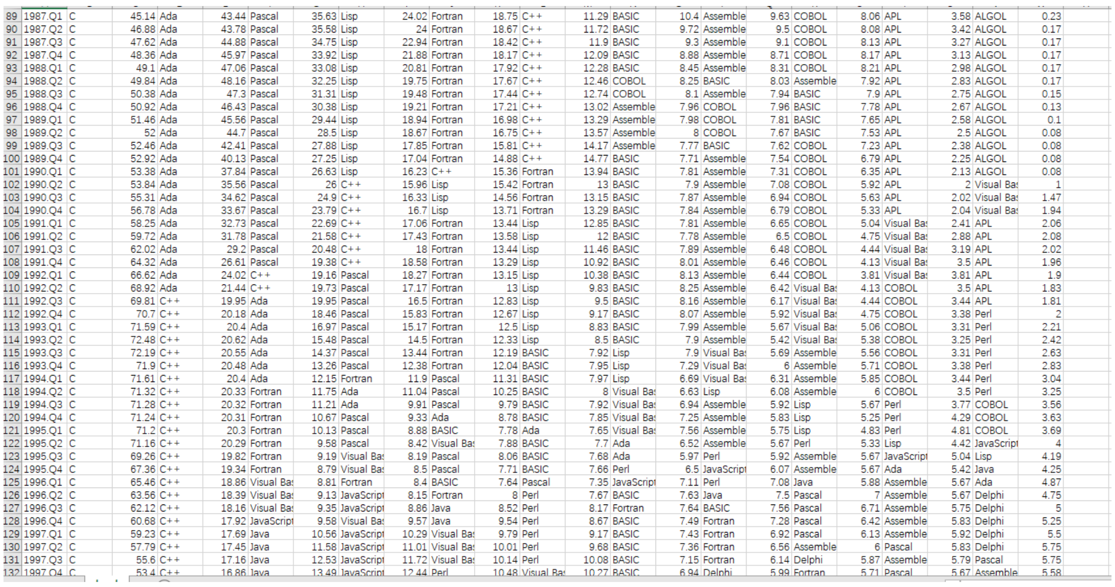
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1965.Q1	Fortran	58.18	COBOL	13.5	ALGOL	10.32	Assemble	5.38	BASIC	2	Lisp	1.48	APL	1				
1965.Q2	Fortran	58.18	COBOL	13.5	ALGOL	10.32	Assemble	5.38	BASIC	2	Lisp	1.48	APL	1				
1965.Q3	Fortran	57.02	COBOL	13.25	ALGOL	10.23	Assemble	5.31	BASIC	2.08	APL	1.73	Lisp	1.46				
1965.Q4	Fortran	55.86	COBOL	13	ALGOL	10.14	Assemble	5.25	APL	2.46	BASIC	2.17	Lisp	1.45				
1966.Q1	Fortran	54.7	COBOL	12.75	ALGOL	10.05	Assemble	5.19	APL	3.19	BASIC	2.25	Lisp	1.44				
1966.Q2	Fortran	53.54	COBOL	12.5	ALGOL	9.96	Assemble	5.13	APL	3.92	BASIC	2.33	Lisp	1.43				
1966.Q3	Fortran	51.11	COBOL	12	ALGOL	9.64	Assemble	4.97	APL	4.13	BASIC	2.36	Lisp	1.39				
1966.Q4	Fortran	48.69	COBOL	11.5	ALGOL	9.31	Assemble	4.81	APL	4.33	BASIC	2.39	Lisp	1.36				
1967.Q1	Fortran	46.26	COBOL	11	ALGOL	8.99	Assemble	4.66	APL	4.54	BASIC	2.42	Lisp	1.33				
1967.Q2	Fortran	43.83	COBOL	10.5	ALGOL	8.67	APL	4.75	Assemble	4.5	BASIC	2.45	Lisp	1.3				
1967.Q3	Fortran	41.02	COBOL	10.04	ALGOL	8.23	APL	4.95	Assemble	4.46	BASIC	2.71	Lisp	1.41				
1967.Q4	Fortran	38.21	COBOL	9.58	ALGOL	7.79	APL	5.15	Assemble	4.42	BASIC	2.96	Lisp	1.52				
1968.Q1	Fortran	35.4	COBOL	9.13	ALGOL	7.35	APL	5.34	Assemble	4.38	BASIC	3.22	Lisp	1.63				
1968.Q2	Fortran	32.58	COBOL	8.67	ALGOL	6.92	APL	5.54	Assemble	4.33	BASIC	3.48	Lisp	1.73				
1968.Q3	Fortran	30.63	COBOL	8.23	ALGOL	6.64	APL	5.25	Assemble	4.4	BASIC	3.59	Lisp	1.83				
1968.Q4	Fortran	28.68	COBOL	7.79	ALGOL	6.36	APL	4.96	Assemble	4.46	BASIC	3.7	Lisp	1.93				
1969.Q1	Fortran	26.73	COBOL	7.35	ALGOL	6.08	APL	4.67	Assemble	4.52	BASIC	3.81	Lisp	2.03				
1969.Q2	Fortran	24.78	COBOL	6.92	ALGOL	5.81	Assemble	4.58	APL	4.38	BASIC	3.92	Lisp	2.13				
1969.Q3	Fortran	24.83	COBOL	6.77	ALGOL	5.56	Assemble	4.65	APL	4.32	BASIC	3.87	Lisp	2.35				
1969.Q4	Fortran	24.89	COBOL	6.63	ALGOL	5.32	Assemble	4.71	APL	4.27	BASIC	3.82	Lisp	2.57				
1970.Q1	Fortran	24.94	COBOL	6.48	ALGOL	5.08	Assemble	4.77	APL	4.22	BASIC	3.77	Lisp	2.78				
1970.Q2	Fortran	25	COBOL	6.33	ALGOL	4.83	Assemble	4.83	APL	4.17	BASIC	3.73	Lisp	3	Pascal	0.42		
1970.Q3	Fortran	24.75	COBOL	6.58	Assemble	4.88	ALGOL	4.69	APL	4.13	BASIC	4.01	Lisp	3.29	Pascal	0.71		
1970.Q4	Fortran	24.5	COBOL	6.83	Assemble	4.92	ALGOL	4.54	BASIC	4.3	APL	4.08	Lisp	3.58	Pascal	1		
1971.Q1	Fortran	24.25	COBOL	7.08	Assemble	4.96	BASIC	4.59	ALGOL	4.4	APL	4.04	Lisp	3.88	Pascal	1.29		
1971.Q2	Fortran	24	COBOL	7.33	Assemble	5	BASIC	4.88	ALGOL	4.25	Lisp	4.17	APL	4	Pascal	1.58		
1971.Q3	Fortran	24.25	COBOL	7.29	BASIC	5.03	Assemble	4.92	Lisp	4.29	ALGOL	4.21	APL	3.85	Pascal	1.69		
1971.Q4	Fortran	24.5	COBOL	7.25	BASIC	5.19	Assemble	4.83	Lisp	4.42	ALGOL	4.17	APL	3.71	Pascal	1.79		
1972.Q1	Fortran	24.75	COBOL	7.21	BASIC	5.34	Assemble	4.75	Lisp	4.54	ALGOL	4.13	APL	3.56	Pascal	1.9		
1972.Q2	Fortran	25	COBOL	7.17	BASIC	5.5	Assemble	4.67	Lisp	4.67	ALGOL	4.08	APL	3.42	Pascal	2		
1972.Q3	Fortran	24.85	COBOL	7.69	BASIC	5.51	Lisp	4.9	Assemble	4.5	ALGOL	4.1	APL	3.52	Pascal	2.38		
1972.Q4	Fortran	24.7	COBOL	8.21	BASIC	5.53	Lisp	5.13	Assemble	4.33	ALGOL	4.13	APL	3.63	Pascal	2.75		
1973.Q1	Fortran	24.55	COBOL	8.73	BASIC	5.54	Lisp	5.35	Assemble	4.17	ALGOL	4.15	APL	3.73	Pascal	3.13		
1973.Q2	Fortran	24.4	COBOL	9.25	Lisp	5.58	BASIC	5.55	ALGOL	4.17	Assemble	4	APL	3.83	Pascal	3.5		
1973.Q3	Fortran	24.36	COBOL	9.58	Lisp	5.83	BASIC	5.59	ALGOL	4.33	Assemble	3.9	APL	3.85	Pascal	3.73		
1973.Q4	Fortran	24.32	COBOL	9.92	Lisp	6.08	BASIC	5.63	ALGOL	4.5	Pascal	3.96	APL	3.88	Assemble	3.79		
1974.Q1	Fortran	24.29	COBOL	10.25	Lisp	6.33	BASIC	5.66	ALGOL	4.67	Pascal	4.19	APL	3.9	Assemble	3.69		
1974.Q2	Fortran	24.25	COBOL	10.59	Lisp	6.58	BASIC	5.7	ALGOL	4.84	Pascal	4.43	APL	3.92	Assemble	3.59		
1974.Q3	Fortran	24.44	COBOL	10.96	Lisp	6.63	BASIC	5.75	ALGOL	4.92	Pascal	4.81	APL	4.13	Assemble	3.71		
1974.Q4	Fortran	24.63	COBOL	11.33	Lisp	6.67	BASIC	5.8	Pascal	5.21	ALGOL	5	APL	4.33	Assemble	3.83		
1975.Q1	Fortran	24.81	COBOL	11.71	Lisp	6.71	BASIC	5.85	Pascal	5.6	ALGOL	5.08	APL	4.54	Assemble	3.96		
1975.Q2	Fortran	25	COBOL	12.08	Lisp	6.75	Pascal	6	BASIC	5.9	ALGOL	5.17	APL	4.75	Assemble	4.08	C	2
1975.Q3	Fortran	25.13	COBOL	12.29	Lisp	6.81	Pascal	6.5	BASIC	5.86	ALGOL	5.13	APL	4.67	Assemble	4.25	C	2.25
1975.Q4	Fortran	25.25	COBOL	12.5	Pascal	7	Lisp	6.88	BASIC	5.81	ALGOL	5.08	APL	4.58	Assemble	4.42	C	2.5

test

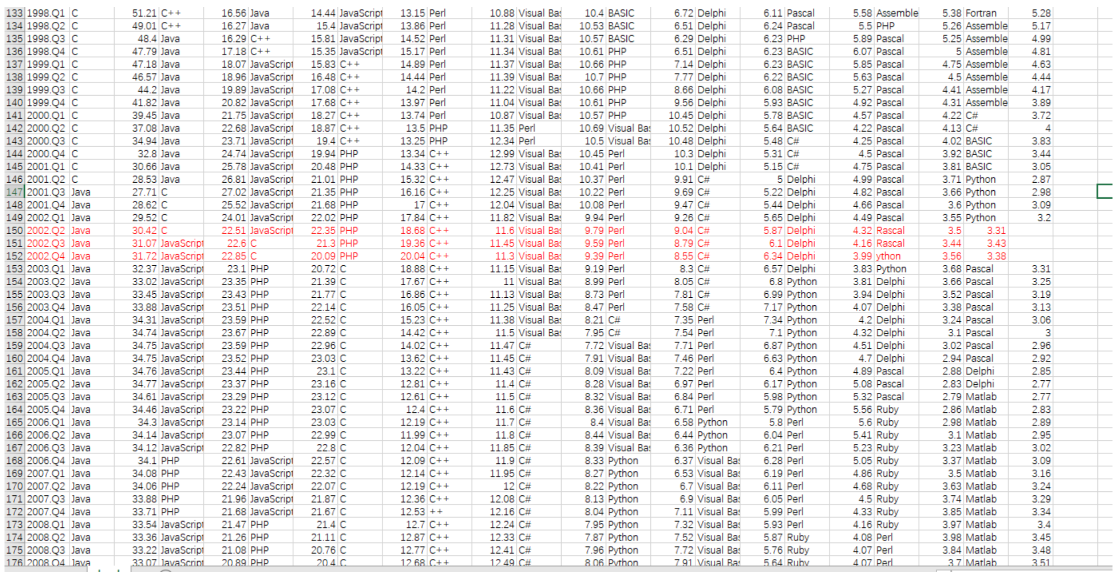


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
45	1976.Q1	Fortran	25.38	COBOL	12.71	Pascal	7.5	Lisp	6.94	BASIC	5.77	ALGOL	5.04	Assemble	4.58	APL	4.5	C	2.75				
46	1976.Q2	Fortran	25.5	COBOL	12.92	Pascal	8	Lisp	7	BASIC	5.72	ALGOL	5	Assemble	4.75	APL	4.42	C	3				
47	1976.Q3	Fortran	25.25	COBOL	12.65	Pascal	8.77	Lisp	7.06	BASIC	6.04	ALGOL	4.79	Assemble	4.79	APL	4.46	C	3.47				
48	1976.Q4	Fortran	25	COBOL	12.38	Pascal	9.54	Lisp	7.13	BASIC	6.35	Assemble	4.83	ALGOL	4.58	APL	4.5	C	3.94				
49	1977.Q1	Fortran	24.75	COBOL	12.1	Pascal	10.31	Lisp	7.19	BASIC	6.66	Assemble	4.88	APL	4.54	C	4.41	ALGOL	4.38				
50	1977.Q2	Fortran	24.5	COBOL	11.83	Pascal	11.08	Lisp	7.25	BASIC	6.97	Assemble	4.92	C	4.88	APL	4.58	ALGOL	4.17				
51	1977.Q3	Fortran	24.38	Pascal	12.06	COBOL	12.04	Lisp	7.5	BASIC	7.42	C	5.09	Assemble	4.9	APL	4.65	ALGOL	4.02				
52	1977.Q4	Fortran	24.25	Pascal	13.04	COBOL	12.25	BASIC	7.86	Lisp	7.75	C	5.3	Assemble	4.88	APL	4.71	ALGOL	3.88				
53	1978.Q1	Fortran	24.13	Pascal	14.02	COBOL	12.46	BASIC	8.31	Lisp	8	C	5.51	Assemble	4.85	APL	4.77	ALGOL	3.73				
54	1978.Q2	Fortran	24	Pascal	15	COBOL	12.67	BASIC	8.75	Lisp	8.25	C	5.72	APL	4.83	Assemble	4.83	ALGOL	3.58				
55	1978.Q3	Fortran	24.25	Pascal	15.58	COBOL	12.31	BASIC	9.52	Lisp	8.33	C	6.07	Assemble	4.73	APL	4.48	ALGOL	3.35				
56	1978.Q4	Fortran	24.5	Pascal	16.17	COBOL	11.96	BASIC	10.29	Lisp	8.42	C	6.42	Assemble	4.63	APL	4.13	ALGOL	3.13				
57	1979.Q1	Fortran	24.75	Pascal	16.75	COBOL	11.6	BASIC	11.06	Lisp	8.5	C	6.77	Assemble	4.52	APL	3.77	ALGOL	2.9				
58	1979.Q2	Fortran	25	Pascal	17.33	BASIC	11.82	COBOL	11.25	Lisp	8.58	C	7.12	Assemble	4.42	APL	3.42	ALGOL	2.67				
59	1979.Q3	Fortran	24.75	Pascal	19.38	BASIC	12.88	COBOL	11.52	Lisp	8.79	C	7.68	Assemble	4.69	APL	3.6	ALGOL	2.63				
60	1979.Q4	Fortran	24.5	Pascal	21.42	BASIC	13.94	COBOL	11.79	Lisp	9	C	8.24	Assemble	4.96	APL	3.79	ALGOL	2.58				
61	1980.Q1	Fortran	24.25	Pascal	23.46	BASIC	14.99	COBOL	12.06	Lisp	9.21	C	8.8	Assemble	5.23	APL	3.98	ALGOL	2.54				
62	1980.Q2	Pascal	25.5	Fortran	24	BASIC	16.05	COBOL	12.33	Lisp	9.42	C	9.36	Assemble	5.5	APL	4.17	Ada	3	ALGOL	2.5	C++	1
63	1980.Q3	Pascal	26.21	Fortran	23.75	BASIC	16.79	COBOL	12.5	C	10.36	Lisp	10	Assemble	6.21	Ada	5	APL	4.31	ALGOL	2.44	C++	1.25
64	1980.Q4	Pascal	26.92	Fortran	23.5	BASIC	17.52	COBOL	12.67	C	11.36	Lisp	10.58	Ada	8	Assemble	6.92	APL	4.46	ALGOL	2.38	C++	1.5
65	1981.Q1	Pascal	27.63	Fortran	23.25	BASIC	18.26	COBOL	12.83	C	12.36	Ada	11.4	Lisp	11.17	Assemble	7.63	APL	4.6	ALGOL	2.31	C++	1.75
66	1981.Q2	Pascal	28.33	Fortran	23	BASIC	19	C	13.36	COBOL	13	Ada	12.22	Lisp	11.75	Assemble	8.33	APL	4.75	ALGOL	2.25	C++	2
67	1981.Q3	Pascal	29.35	Fortran	22.75	BASIC	19.63	Ada	15.08	C	14.58	COBOL	12.96	Lisp	12.15	Assemble	8.88	APL	4.81	C++	2.5	ALGOL	2.04
68	1981.Q4	Pascal	30.38	Fortran	22.5	BASIC	20.26	C	15.8	Ada	15.6	COBOL	12.92	Lisp	12.54	Assemble	9.42	APL	4.88	C++	3	ALGOL	1.83
69	1982.Q1	Pascal	31.4	Fortran	22.25	BASIC	20.89	C	17.02	Ada	16.12	Lisp	12.94	COBOL	12.88	Assemble	9.96	APL	4.94	C++	3.5	ALGOL	1.63
70	1982.Q2	Pascal	32.42	Fortran	22	BASIC	21.52	C	18.24	Ada	16.64	Lisp	13.33	COBOL	12.83	Assemble	10.5	APL	5	C++	4	ALGOL	1.42
71	1982.Q3	Pascal	33.23	Fortran	21.85	BASIC	21.46	C	19.47	Ada	17.84	Lisp	14.38	COBOL	12.58	Assemble	10.96	APL	5.13	C++	4.45	ALGOL	1.46
72	1982.Q4	Pascal	34.04	Fortran	21.71	BASIC	21.4	C	20.7	Ada	19.03	Lisp	15.42	COBOL	12.33	Assemble	11.42	APL	5.25	C++	4.89	ALGOL	1.5
73	1983.Q1	Pascal	34.85	C	21.93	Fortran	21.56	BASIC	21.34	Ada	20.23	Lisp	16.46	COBOL	12.08	Assemble	11.88	APL	5.38	C++	5.34	ALGOL	1.54
74	1983.Q2	Pascal	35.67	C	23.16	Ada	21.42	Fortran	21.42	BASIC	21.27	Lisp	17.5	Assemble	12.33	COBOL	11.83	C++	5.79	APL	5.5	ALGOL	1.58
75	1983.Q3	Pascal	35.6	C	25.11	Ada	22.69	Fortran	21.31	BASIC	20.71	Lisp	18.92	Assemble	12.46	COBOL	11.38	C++	6.28	APL	5.38	ALGOL	1.4
76	1983.Q4	Pascal	35.54	C	27.06	Ada	23.96	Fortran	21.21	Lisp	20.33	BASIC	20.14	Assemble	12.58	COBOL	10.92	C++	6.76	APL	5.25	ALGOL	1.21
77	1984.Q1	Pascal	35.48	C	29.01	Ada	25.23	Lisp	21.75	Fortran	21.1	BASIC	19.57	Assemble	12.71	COBOL	10.46	C++	7.25	APL	5.13	ALGOL	1.02
78	1984.Q2	Pascal	35.42	C	30.96	Ada	26.5	Lisp	23.17	Fortran	21	BASIC	19	Assemble	12.83	COBOL	10	C++	7.74	APL	5	ALGOL	0.83
79	1984.Q3	Pascal	35.73	C	32.31	Ada	28.02	Lisp	23	Fortran	20.83	BASIC	18.33	Assemble	12.4	COBOL	9.54	C++	8.08	APL	4.79	ALGOL	0.77
80	1984.Q4	Pascal	36.04	C	33.66	Ada	29.54	Lisp	22.83	Fortran	20.67	BASIC	17.66	Assemble	11.96	COBOL	9.08	C++	8.41	APL	4.58	ALGOL	0.71
81	1985.Q1	Pascal	36.35	C	35.01	Ada	31.06	Lisp	22.67	Fortran	20.5	BASIC	16.99	Assemble	11.52	C++	8.75	COBOL	8.63	APL	4.38	ALGOL	0.65
82	1985.Q2	Pascal	36.67	C	36.36	Ada	32.58	Lisp	22.5	Fortran	20.33	BASIC	16.32	Assemble	11.08	C++	9.09	COBOL	8.17	APL	4.17	ALGOL	0.58
83	1985.Q3	C	37.25	Pascal	36.44	Ada	35.04	Lisp	22.9	Fortran	20	BASIC	15.35	Assemble	10.81	C++	9.31	COBOL	8.13	APL	4.15	ALGOL	0.54
84	1985.Q4	C	38.14	Ada	37.5	Pascal	36.21	Lisp	23.29	Fortran	19.67	BASIC	14.38	Assemble	10.54	C++	9.54	COBOL	8.08	APL	4.13	ALGOL	0.5
85	1986.Q1	Ada	39.96	C	39.03	Pascal	35.98	Lisp	23.69	Fortran	19.33	BASIC	13.4	Assemble	10.27	C++	9.76	COBOL	8.04	APL	4.1	ALGOL	0.46
86	1986.Q2	Ada	42.42	C	39.92	Pascal	35.75	Lisp	24.08	Fortran	19	BASIC	12.43	Assemble	10	C++	9.98	COBOL	8	APL	4.08	ALGOL	0.42
87	1986.Q3	Ada	42.76	C	41.66	Pascal	35.71	Lisp	24.06	Fortran	18.92	BASIC	11.75	C++	10.41	Assemble	9.88	COBOL	8.02	APL	3.92	ALGOL	0.35
88	1986.Q4	C	43.4	Ada	43.1	Pascal	35.67	Lisp	24.04	Fortran	18.83	BASIC	11.07	C++	10.85	Assemble	9.75	COBOL	8.04	APL	3.75	ALGOL	0.29

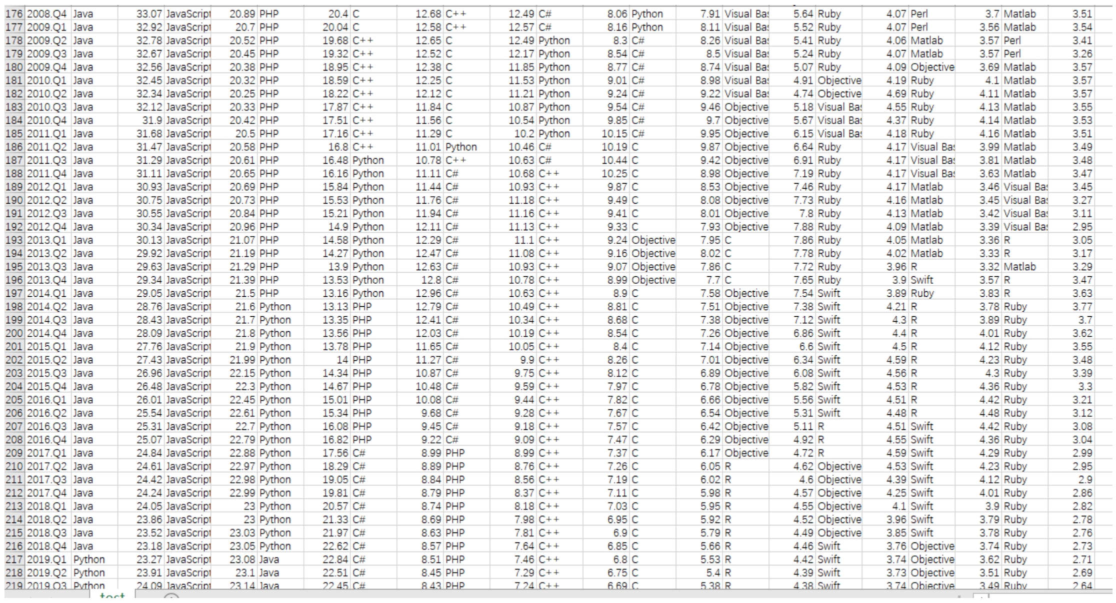
test



89	1987.Q1	C	45.14	Ada	43.44	Pascal	35.63	Lisp	24.02	Fortran	18.75	C++	11.29	BASIC	10.4	Assemble	9.63	COBOL	8.06	APL	3.58	ALGOL	0.23
90	1987.Q2	C	46.88	Ada	43.78	Pascal	35.58	Lisp	24	Fortran	18.67	C++	11.72	BASIC	9.72	Assemble	9.5	COBOL	8.08	APL	3.42	ALGOL	0.17
91	1987.Q3	C	47.62	Ada	44.88	Pascal	34.75	Lisp	22.94	Fortran	18.42	C++	11.9	BASIC	9.3	Assemble	9.1	COBOL	8.13	APL	3.27	ALGOL	0.17
92	1987.Q4	C	48.36	Ada	45.97	Pascal	33.92	Lisp	21.88	Fortran	18.17	C++	12.09	BASIC	8.88	Assemble	8.71	COBOL	8.17	APL	3.13	ALGOL	0.17
93	1988.Q1	C	49.1	Ada	47.06	Pascal	33.08	Lisp	20.81	Fortran	17.92	C++	12.28	BASIC	8.45	Assemble	8.31	COBOL	8.21	APL	2.98	ALGOL	0.17
94	1988.Q2	C	49.84	Ada	48.16	Pascal	32.25	Lisp	19.75	Fortran	17.67	C++	12.46	COBOL	8.25	BASIC	8.03	Assemble	7.92	APL	2.83	ALGOL	0.17
95	1988.Q3	C	50.38	Ada	47.3	Pascal	31.31	Lisp	19.48	Fortran	17.44	C++	12.74	COBOL	8.1	Assemble	7.94	BASIC	7.9	APL	2.75	ALGOL	0.15
96	1988.Q4	C	50.92	Ada	46.43	Pascal	30.38	Lisp	19.21	Fortran	17.21	C++	13.02	Assemble	7.96	COBOL	7.96	BASIC	7.78	APL	2.67	ALGOL	0.13
97	1989.Q1	C	51.46	Ada	45.56	Pascal	29.44	Lisp	18.94	Fortran	16.98	C++	13.29	Assemble	7.98	COBOL	7.81	BASIC	7.65	APL	2.58	ALGOL	0.1
98	1989.Q2	C	52	Ada	44.7	Pascal	28.5	Lisp	18.67	Fortran	16.75	C++	13.57	Assemble	8	COBOL	7.67	BASIC	7.53	APL	2.5	ALGOL	0.08
99	1989.Q3	C	52.46	Ada	42.41	Pascal	27.88	Lisp	17.85	Fortran	15.81	C++	14.17	Assemble	7.77	BASIC	7.62	COBOL	7.23	APL	2.38	ALGOL	0.08
100	1989.Q4	C	52.92	Ada	40.13	Pascal	27.25	Lisp	17.04	Fortran	14.88	C++	14.77	BASIC	7.71	Assemble	7.54	COBOL	6.79	APL	2.25	ALGOL	0.08
101	1990.Q1	C	53.38	Ada	37.84	Pascal	26.63	Lisp	16.23	C++	15.36	Fortran	13.94	BASIC	7.81	Assemble	7.31	COBOL	6.35	APL	2.13	ALGOL	0.08
102	1990.Q2	C	53.84	Ada	35.56	Pascal	26	C++	15.96	Lisp	15.42	Fortran	13	BASIC	7.9	Assemble	7.08	COBOL	5.92	APL	2	Visual Ba	1
103	1990.Q3	C	55.31	Ada	34.62	Pascal	24.9	C++	16.33	Lisp	14.56	Fortran	13.15	BASIC	7.87	Assemble	6.94	COBOL	5.63	APL	2.02	Visual Ba	1.47
104	1990.Q4	C	56.78	Ada	33.67	Pascal	23.79	C++	16.7	Lisp	13.71	Fortran	13.29	BASIC	7.84	Assemble	6.79	COBOL	5.33	APL	2.04	Visual Ba	1.94
105	1991.Q1	C	58.25	Ada	32.73	Pascal	22.69	C++	17.06	Fortran	13.44	Lisp	12.85	BASIC	7.81	Assemble	6.65	COBOL	5.04	Visual Ba	2.41	APL	2.06
106	1991.Q2	C	59.72	Ada	31.78	Pascal	21.58	C++	17.43	Fortran	13.58	Lisp	12	BASIC	7.78	Assemble	6.5	COBOL	4.75	Visual Ba	2.88	APL	2.08
107	1991.Q3	C	62.02	Ada	29.2	Pascal	20.48	C++	18	Fortran	13.44	Lisp	11.46	BASIC	7.89	Assemble	6.48	COBOL	4.44	Visual Ba	3.19	APL	2.02
108	1991.Q4	C	64.32	Ada	26.61	Pascal	19.38	C++	18.58	Fortran	13.29	Lisp	10.92	BASIC	8.01	Assemble	6.46	COBOL	4.13	Visual Ba	3.5	APL	1.96
109	1992.Q1	C	66.62	Ada	24.02	C++	19.16	Pascal	18.27	Fortran	13.15	Lisp	10.38	BASIC	8.13	Assemble	6.44	COBOL	3.81	Visual Ba	3.81	APL	1.9
110	1992.Q2	C	68.92	Ada	21.44	C++	19.73	Pascal	17.17	Fortran	13	Lisp	9.83	BASIC	8.25	Assemble	6.42	Visual Ba	4.13	COBOL	3.5	APL	1.83
111	1992.Q3	C	69.81	C++	19.95	Ada	19.95	Pascal	16.5	Fortran	12.83	Lisp	9.5	BASIC	8.16	Assemble	6.17	Visual Ba	4.44	COBOL	3.44	APL	1.81
112	1992.Q4	C	70.7	C++	20.18	Ada	18.46	Pascal	15.83	Fortran	12.67	Lisp	9.17	BASIC	8.07	Assemble	5.92	Visual Ba	4.75	COBOL	3.38	Perl	2
113	1993.Q1	C	71.59	C++	20.4	Ada	16.97	Pascal	15.17	Fortran	12.5	Lisp	8.83	BASIC	7.99	Assemble	5.67	Visual Ba	5.06	COBOL	3.31	Perl	2.21
114	1993.Q2	C	72.48	C++	20.62	Ada	15.48	Pascal	14.5	Fortran	12.33	Lisp	8.5	BASIC	7.9	Assemble	5.42	Visual Ba	5.38	COBOL	3.25	Perl	2.42
115	1993.Q3	C	72.19	C++	20.55	Ada	14.37	Pascal	13.44	Fortran	12.19	BASIC	7.92	Lisp	7.9	Visual Ba	5.69	Assemble	5.56	COBOL	3.31	Perl	2.63
116	1993.Q4	C	71.9	C++	20.48	Ada	13.26	Pascal	12.38	Fortran	12.04	BASIC	7.95	Lisp	7.29	Visual Ba	6	Assemble	5.71	COBOL	3.38	Perl	2.83
117	1994.Q1	C	71.61	C++	20.4	Ada	12.15	Fortran	11.9	Pascal	11.31	BASIC	7.97	Lisp	6.69	Visual Ba	6.31	Assemble	5.85	COBOL	3.44	Perl	3.04
118	1994.Q2	C	71.32	C++	20.33	Fortran	11.75	Ada	11.04	Pascal	10.25	BASIC	8	Visual Ba	6.63	Lisp	6.08	Assemble	6	COBOL	3.5	Perl	3.25
119	1994.Q3	C	71.28	C++	20.32	Fortran	11.21	Ada	9.91	Pascal	9.79	BASIC	7.92	Visual Ba	6.94	Assemble	5.92	Lisp	5.67	Perl	3.77	COBOL	3.56
120	1994.Q4	C	71.24	C++	20.31	Fortran	10.67	Pascal	9.33	Ada	8.78	BASIC	7.85	Visual Ba	7.25	Assemble	5.83	Lisp	5.25	Perl	4.29	COBOL	3.63
121	1995.Q1	C	71.2	C++	20.3	Fortran	10.13	Pascal	8.88	BASIC	7.78	Ada	7.65	Visual Ba	7.56	Assemble	5.75	Lisp	4.83	Perl	4.81	COBOL	3.69
122	1995.Q2	C	71.16	C++	20.29	Fortran	9.58	Pascal	8.42	Visual Ba	7.88	BASIC	7.7	Ada	6.52	Assemble	5.67	Perl	5.33	Lisp	4.42	JavaScript	4
123	1995.Q3	C	69.26	C++	19.82	Fortran	9.19	Visual Ba	8.19	Pascal	8.06	BASIC	7.68	Ada	5.97	Perl	5.92	Assemble	5.67	JavaScript	5.04	Lisp	4.19
124	1995.Q4	C	67.36	C++	19.34	Fortran	8.79	Visual Ba	8.5	Pascal	7.71	BASIC	7.66	Perl	6.5	JavaScript	6.07	Assemble	5.67	Ada	5.42	Java	4.25
125	1996.Q1	C	65.46	C++	18.86	Visual Ba	8.81	Fortran	8.4	BASIC	7.64	Pascal	7.35	JavaScript	7.11	Perl	7.08	Java	5.88	Assemble	5.67	Ada	4.87
126	1996.Q2	C	63.56	C++	18.39	Visual Ba	9.13	JavaScript	8.15	Fortran	8	Perl	7.67	BASIC	7.63	Java	7.5	Pascal	7	Assemble	5.67	Delphi	4.75
127	1996.Q3	C	62.12	C++	18.16	Visual Ba	9.35	JavaScript	8.86	Java	8.52	Perl	8.17	Fortran	7.64	BASIC	7.56	Pascal	6.71	Assemble	5.75	Delphi	5
128	1996.Q4	C	60.68	C++	17.92	JavaScript	9.58	Visual Ba	9.57	Java	9.54	Perl	8.67	BASIC	7.49	Fortran	7.28	Pascal	6.42	Assemble	5.83	Delphi	5.25
129	1997.Q1	C	59.23	C++	17.69	Java	10.56	JavaScript	10.29	Visual Ba	9.79	Perl	9.17	BASIC	7.43	Fortran	6.92	Pascal	6.13	Assemble	5.92	Delphi	5.5
130	1997.Q2	C	57.79	C++	17.45	Java	11.58	JavaScript	11.01	Visual Ba	10.01	Perl	9.68	BASIC	7.36	Fortran	6.56	Assemble	6	Pascal	5.83	Delphi	5.75
131	1997.Q3	C	55.6	C++	17.16	Java	12.53	JavaScript	11.72	Visual Ba	10.14	Perl	10.08	BASIC	7.15	Fortran	6.14	Delphi	5.87	Assemble	5.79	Pascal	5.75
132	1997.Q4	C	53.4	C++	16.86	Java	13.49	JavaScript	12.44	Perl	10.48	Visual Ba	10.27	BASIC	6.94	Delphi	5.99	Fortran	5.71	Pascal	5.67	Assemble	5.58



133	1998.Q1	C	51.21	C++	16.56	Java	14.44	JavaScript	13.15	Perl	10.88	Visual Bas	10.4	BASIC	6.72	Delphi	6.11	Pascal	5.58	Assemble	5.38	Fortran	5.28
134	1998.Q2	C	49.01	C++	16.27	Java	15.4	JavaScript	13.86	Perl	11.28	Visual Bas	10.53	BASIC	6.51	Delphi	6.24	Pascal	5.5	PHP	5.26	Assemble	5.17
135	1998.Q3	C	48.4	Java	16.29	C++	15.81	JavaScript	14.52	Perl	11.31	Visual Bas	10.57	BASIC	6.29	Delphi	6.23	PHP	5.89	Pascal	5.25	Assemble	4.99
136	1998.Q4	C	47.79	Java	17.18	C++	15.35	JavaScript	15.17	Perl	11.34	Visual Bas	10.61	PHP	6.51	Delphi	6.23	BASIC	6.07	Pascal	5	Assemble	4.81
137	1999.Q1	C	47.18	Java	18.07	JavaScript	15.83	C++	14.89	Perl	11.37	Visual Bas	10.66	PHP	7.14	Delphi	6.23	BASIC	5.85	Pascal	4.75	Assemble	4.63
138	1999.Q2	C	46.57	Java	18.96	JavaScript	16.48	C++	14.44	Perl	11.39	Visual Bas	10.7	PHP	7.77	Delphi	6.22	BASIC	5.63	Pascal	4.5	Assemble	4.44
139	1999.Q3	C	44.2	Java	19.89	JavaScript	17.08	C++	14.2	Perl	11.22	Visual Bas	10.66	PHP	8.66	Delphi	6.08	BASIC	5.27	Pascal	4.41	Assemble	4.17
140	1999.Q4	C	41.82	Java	20.82	JavaScript	17.68	C++	13.97	Perl	11.04	Visual Bas	10.61	PHP	9.56	Delphi	5.93	BASIC	4.92	Pascal	4.31	Assemble	3.89
141	2000.Q1	C	39.45	Java	21.75	JavaScript	18.27	C++	13.74	Perl	10.87	Visual Bas	10.57	PHP	10.45	Delphi	5.78	BASIC	4.57	Pascal	4.22	C#	3.72
142	2000.Q2	C	37.08	Java	22.68	JavaScript	18.87	C++	13.5	PHP	11.35	Perl	10.69	Visual Bas	10.52	Delphi	5.64	BASIC	4.22	Pascal	4.13	C#	4
143	2000.Q3	C	34.94	Java	23.71	JavaScript	19.4	C++	13.25	PHP	12.34	Perl	10.5	Visual Bas	10.48	Delphi	5.48	C#	4.25	Pascal	4.02	BASIC	3.83
144	2000.Q4	C	32.8	Java	24.74	JavaScript	19.94	PHP	13.34	C++	12.99	Visual Bas	10.45	Perl	10.3	Delphi	5.31	C#	4.5	Pascal	3.92	BASIC	3.44
145	2001.Q1	C	30.66	Java	25.78	JavaScript	20.48	PHP	14.33	C++	12.73	Visual Bas	10.41	Perl	10.1	Delphi	5.15	C#	4.75	Pascal	3.81	BASIC	3.05
146	2001.Q2	C	28.53	Java	26.81	JavaScript	21.01	PHP	15.32	C++	12.47	Visual Bas	10.37	Perl	9.91	C#	5	Delphi	4.99	Pascal	3.71	Python	2.87
147	2001.Q3	Java	27.71	C	27.02	JavaScript	21.35	PHP	16.16	C++	12.25	Visual Bas	10.22	Perl	9.69	C#	5.22	Delphi	4.82	Pascal	3.66	Python	2.98
148	2001.Q4	Java	28.62	C	25.52	JavaScript	21.68	PHP	17	C++	12.04	Visual Bas	10.08	Perl	9.47	C#	5.44	Delphi	4.66	Pascal	3.6	Python	3.09
149	2002.Q1	Java	29.52	C	24.01	JavaScript	22.02	PHP	17.84	C++	11.82	Visual Bas	9.94	Perl	9.26	C#	5.65	Delphi	4.49	Pascal	3.55	Python	3.2
150	2002.Q2	Java	30.42	C	22.51	JavaScript	22.35	PHP	18.68	C++	11.6	Visual Bas	9.79	Perl	9.04	C#	5.87	Delphi	4.32	Rascal	3.5	3.31	
151	2002.Q3	Java	31.07	JavaScript	22.6	C	21.3	PHP	19.36	C++	11.45	Visual Bas	9.59	Perl	8.79	C#	6.1	Delphi	4.16	Rascal	3.44	3.43	
152	2002.Q4	Java	31.72	JavaScript	22.85	C	20.09	PHP	20.04	C++	11.3	Visual Bas	9.39	Perl	8.55	C#	6.34	Delphi	3.99	ython	3.56	3.38	
153	2003.Q1	Java	32.37	JavaScript	23.1	PHP	20.72	C	18.88	C++	11.15	Visual Bas	9.19	Perl	8.3	C#	6.57	Delphi	3.83	Python	3.68	Pascal	3.31
154	2003.Q2	Java	33.02	JavaScript	23.35	PHP	21.39	C	17.67	C++	11	Visual Bas	8.99	Perl	8.05	C#	6.8	Python	3.81	Delphi	3.66	Pascal	3.25
155	2003.Q3	Java	33.45	JavaScript	23.43	PHP	21.77	C	16.86	C++	11.13	Visual Bas	8.73	Perl	7.81	C#	6.99	Python	3.94	Delphi	3.52	Pascal	3.19
156	2003.Q4	Java	33.88	JavaScript	23.51	PHP	22.14	C	16.05	C++	11.25	Visual Bas	8.47	Perl	7.58	C#	7.17	Python	4.07	Delphi	3.38	Pascal	3.13
157	2004.Q1	Java	34.31	JavaScript	23.59	PHP	22.52	C	15.23	C++	11.38	Visual Bas	8.21	C#	7.35	Perl	7.34	Python	4.2	Delphi	3.24	Pascal	3.06
158	2004.Q2	Java	34.74	JavaScript	23.67	PHP	22.89	C	14.42	C++	11.5	Visual Bas	7.95	C#	7.54	Perl	7.1	Python	4.32	Delphi	3.1	Pascal	3
159	2004.Q3	Java	34.75	JavaScript	23.59	PHP	22.96	C	14.02	C++	11.47	C#	7.72	Visual Bas	7.71	Perl	6.87	Python	4.51	Delphi	3.02	Pascal	2.96
160	2004.Q4	Java	34.75	JavaScript	23.52	PHP	23.03	C	13.62	C++	11.45	C#	7.91	Visual Bas	7.46	Perl	6.63	Python	4.7	Delphi	2.94	Pascal	2.92
161	2005.Q1	Java	34.76	JavaScript	23.44	PHP	23.1	C	13.22	C++	11.43	C#	8.09	Visual Bas	7.22	Perl	6.4	Python	4.89	Pascal	2.88	Delphi	2.85
162	2005.Q2	Java	34.77	JavaScript	23.37	PHP	23.16	C	12.81	C++	11.4	C#	8.28	Visual Bas	6.97	Perl	6.17	Python	5.08	Pascal	2.83	Delphi	2.77
163	2005.Q3	Java	34.61	JavaScript	23.29	PHP	23.12	C	12.61	C++	11.5	C#	8.32	Visual Bas	6.84	Perl	5.98	Python	5.32	Pascal	2.79	Matlab	2.77
164	2005.Q4	Java	34.46	JavaScript	23.22	PHP	23.07	C	12.4	C++	11.6	C#	8.36	Visual Bas	6.71	Perl	5.79	Python	5.56	Ruby	2.86	Matlab	2.83
165	2006.Q1	Java	34.3	JavaScript	23.14	PHP	23.03	C	12.19	C++	11.7	C#	8.4	Visual Bas	6.58	Python	5.8	Perl	5.6	Ruby	2.98	Matlab	2.89
166	2006.Q2	Java	34.14	JavaScript	23.07	PHP	22.99	C	11.99	C++	11.8	C#	8.44	Visual Bas	6.44	Python	6.04	Perl	5.41	Ruby	3.1	Matlab	2.95
167	2006.Q3	Java	34.12	JavaScript	22.82	PHP	22.8	C	12.04	C++	11.85	C#	8.39	Visual Bas	6.36	Python	6.21	Perl	5.23	Ruby	3.23	Matlab	3.02
168	2006.Q4	Java	34.1	PHP	22.61	JavaScript	22.57	C	12.09	C++	11.9	C#	8.33	Python	6.37	Visual Bas	6.28	Perl	5.05	Ruby	3.37	Matlab	3.09
169	2007.Q1	Java	34.08	PHP	22.43	JavaScript	22.32	C	12.14	C++	11.95	C#	8.27	Python	6.53	Visual Bas	6.19	Perl	4.86	Ruby	3.5	Matlab	3.16
170	2007.Q2	Java	34.06	PHP	22.24	JavaScript	22.07	C	12.19	C++	12	C#	8.22	Python	6.7	Visual Bas	6.11	Perl	4.68	Ruby	3.63	Matlab	3.24
171	2007.Q3	Java	33.88	PHP	21.96	JavaScript	21.87	C	12.36	C++	12.08	C#	8.13	Python	6.9	Visual Bas	6.05	Perl	4.5	Ruby	3.74	Matlab	3.29
172	2007.Q4	Java	33.71	PHP	21.68	JavaScript	21.67	C	12.53	++	12.16	C#	8.04	Python	7.11	Visual Bas	5.99	Perl	4.33	Ruby	3.85	Matlab	3.34
173	2008.Q1	Java	33.54	JavaScript	21.47	PHP	21.4	C	12.7	C++	12.24	C#	7.95	Python	7.32	Visual Bas	5.93	Perl	4.16	Ruby	3.97	Matlab	3.4
174	2008.Q2	Java	33.36	JavaScript	21.26	PHP	21.11	C	12.87	C++	12.33	C#	7.87	Python	7.52	Visual Bas	5.87	Ruby	4.08	Perl	3.98	Matlab	3.45
175	2008.Q3	Java	33.22	JavaScript	21.08	PHP	20.76	C	12.77	C++	12.41	C#	7.96	Python	7.72	Visual Bas	5.76	Ruby	4.07	Perl	3.84	Matlab	3.48
176	2008.Q4	Java	33.07	JavaScript	20.89	PHP	20.4	C	12.68	C++	12.49	C#	8.06	Python	7.91	Visual Bas	5.64	Ruby	4.07	Perl	3.7	Matlab	3.51



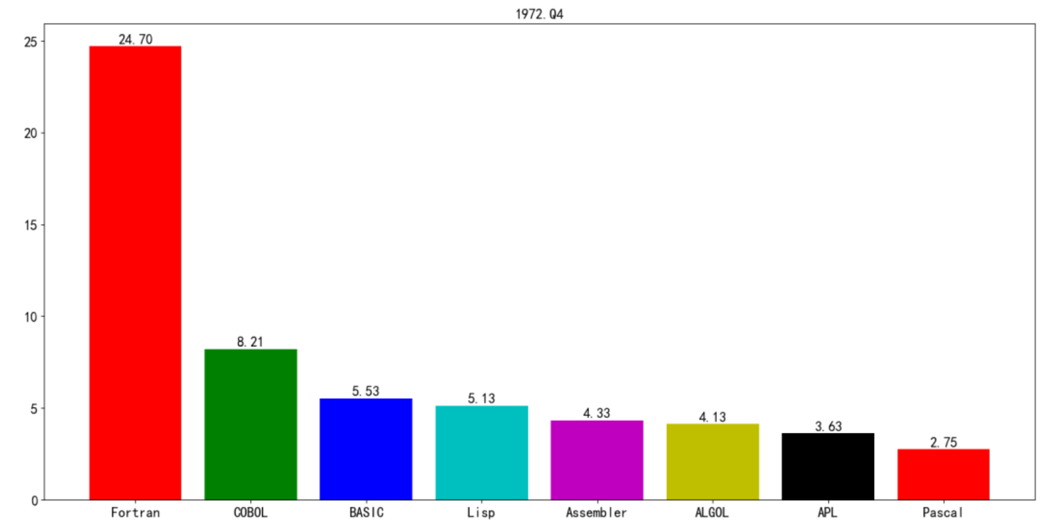
176	2008.Q4	Java	33.07	JavaScript	20.89	PHP	20.4	C	12.68	C++	12.49	C#	8.06	Python	7.91	Visual Bas	5.64	Ruby	4.07	Perl	3.7	Matlab	3.51
177	2009.Q1	Java	32.92	JavaScript	20.7	PHP	20.04	C	12.58	C++	12.57	C#	8.16	Python	8.11	Visual Bas	5.52	Ruby	4.07	Perl	3.56	Matlab	3.54
178	2009.Q2	Java	32.78	JavaScript	20.52	PHP	19.68	C++	12.65	C	12.49	Python	8.3	C#	8.26	Visual Bas	5.41	Ruby	4.06	Matlab	3.57	Perl	3.41
179	2009.Q3	Java	32.67	JavaScript	20.45	PHP	19.32	C++	12.52	C	12.17	Python	8.54	C#	8.5	Visual Bas	5.24	Ruby	4.07	Matlab	3.57	Perl	3.26
180	2009.Q4	Java	32.56	JavaScript	20.38	PHP	18.95	C++	12.38	C	11.85	Python	8.77	C#	8.74	Visual Bas	5.07	Ruby	4.09	Objective	3.69	Matlab	3.57
181	2010.Q1	Java	32.45	JavaScript	20.32	PHP	18.59	C++	12.25	C	11.53	Python	9.01	C#	8.98	Visual Bas	4.91	Objective	4.19	Ruby	4.1	Matlab	3.57
182	2010.Q2	Java	32.34	JavaScript	20.25	PHP	18.22	C++	12.12	C	11.21	Python	9.24	C#	9.22	Visual Bas	4.74	Objective	4.69	Ruby	4.11	Matlab	3.57
183	2010.Q3	Java	32.12	JavaScript	20.33	PHP	17.87	C++	11.84	C	10.87	Python	9.54	C#	9.46	Objective	5.18	Visual Bas	4.55	Ruby	4.13	Matlab	3.55
184	2010.Q4	Java	31.9	JavaScript	20.42	PHP	17.51	C++	11.56	C	10.54	Python	9.85	C#	9.7	Objective	5.67	Visual Bas	4.37	Ruby	4.14	Matlab	3.53
185	2011.Q1	Java	31.68	JavaScript	20.5	PHP	17.16	C++	11.29	C	10.2	Python	10.15	C#	9.95	Objective	6.15	Visual Bas	4.18	Ruby	4.16	Matlab	3.51
186	2011.Q2	Java	31.47	JavaScript	20.58	PHP	16.8	C++	11.01	Python	10.46	C#	10.19	C	9.87	Objective	6.64	Ruby	4.17	Visual Bas	3.99	Matlab	3.49
187	2011.Q3	Java	31.29	JavaScript	20.61	PHP	16.48	Python	10.78	C++	10.63	C#	10.44	C	9.42	Objective	6.91	Ruby	4.17	Visual Bas	3.81	Matlab	3.48
188	2011.Q4	Java	31.11	JavaScript	20.65	PHP	16.16	Python	11.11	C#	10.68	C++	10.25	C	8.98	Objective	7.19	Ruby	4.17	Visual Bas	3.63	Matlab	3.47
189	2012.Q1	Java	30.93	JavaScript	20.69	PHP	15.84	Python	11.44	C#	10.93	C++	9.87	C	8.53	Objective	7.46	Ruby	4.17	Matlab	3.46	Visual Bas	3.45
190	2012.Q2	Java	30.75	JavaScript	20.73	PHP	15.53	Python	11.76	C#	11.18	C++	9.49	C	8.08	Objective	7.73	Ruby	4.16	Matlab	3.45	Visual Bas	3.27
191	2012.Q3	Java	30.55	JavaScript	20.84	PHP	15.21	Python	11.94	C#	11.16	C++	9.41	C	8.01	Objective	7.8	Ruby	4.13	Matlab	3.42	Visual Bas	3.11
192	2012.Q4	Java	30.34	JavaScript	20.96	PHP	14.9	Python	12.11	C#	11.13	C++	9.33	C	7.93	Objective	7.88	Ruby	4.09	Matlab	3.39	Visual Bas	2.95
193	2013.Q1	Java	30.13	JavaScript	21.07	PHP	14.58	Python	12.29	C#	11.1	C++	9.24	Objective	7.95	C	7.86	Ruby	4.05	Matlab	3.36	R	3.05
194	2013.Q2	Java	29.92	JavaScript	21.19	PHP	14.27	Python	12.47	C#	11.08	C++	9.16	Objective	8.02	C	7.78	Ruby	4.02	Matlab	3.33	R	3.17
195	2013.Q3	Java	29.63	JavaScript	21.29	PHP	13.9	Python	12.63	C#	10.93	C++	9.07	Objective	7.86	C	7.72	Ruby	3.96	R	3.32	Matlab	3.29
196	2013.Q4	Java	29.34	JavaScript	21.39	PHP	13.53	Python	12.8	C#	10.78	C++	8.99	Objective	7.7	C	7.65	Ruby	3.9	Swift	3.57	R	3.47
197	2014.Q1	Java	29.05	JavaScript	21.5	PHP	13.16	Python	12.96	C#	10.63	C++	8.9	C	7.58	Objective	7.54	Swift	3.89	Ruby	3.83	R	3.63
198	2014.Q2	Java	28.76	JavaScript	21.6	Python	13.13	PHP	12.79	C#	10.49	C++	8.81	C	7.51	Objective	7.38	Swift	4.21	R	3.78	Ruby	3.77
199	2014.Q3	Java	28.43	JavaScript	21.7	Python	13.35	PHP	12.41	C#	10.34	C++	8.68	C	7.38	Objective	7.12	Swift	4.3	R	3.89	Ruby	3.7
200	2014.Q4	Java	28.09	JavaScript	21.8	Python	13.56	PHP	12.03	C#	10.19	C++	8.54	C	7.26	Objective	6.86	Swift	4.4	R	4.01	Ruby	3.62
201	2015.Q1	Java	27.76	JavaScript	21.9	Python	13.78	PHP	11.65	C#	10.05	C++	8.4	C	7.14	Objective	6.6	Swift	4.5	R	4.12	Ruby	3.55
202	2015.Q2	Java	27.43	JavaScript	21.99	Python	14	PHP	11.27	C#	9.9	C++	8.26	C	7.01	Objective	6.34	Swift	4.59	R	4.23	Ruby	3.48
203	2015.Q3	Java	26.96	JavaScript	22.15	Python	14.34	PHP	10.87	C#	9.75	C++	8.12	C	6.89	Objective	6.08	Swift	4.56	R	4.3	Ruby	3.39
204	2015.Q4	Java	26.48	JavaScript	22.3	Python	14.67	PHP	10.48	C#	9.59	C++	7.97	C	6.78	Objective	5.82	Swift	4.53	R	4.36	Ruby	3.3
205	2016.Q1	Java	26.01	JavaScript	22.45	Python	15.01	PHP	10.08	C#	9.44	C++	7.82	C	6.66	Objective	5.56	Swift	4.51	R	4.42	Ruby	3.21
206	2016.Q2	Java	25.54	JavaScript	22.61	Python	15.34	PHP	9.68	C#	9.28	C++	7.67	C	6.54	Objective	5.31	Swift	4.48	R	4.48	Ruby	3.12
207	2016.Q3	Java	25.31	JavaScript	22.7	Python	16.08	PHP	9.45	C#	9.18	C++	7.57	C	6.42	Objective	5.11	R	4.51	Swift	4.42	Ruby	3.08
208	2016.Q4	Java	25.07	JavaScript	22.79	Python	16.82	PHP	9.22	C#	9.09	C++	7.47	C	6.29	Objective	4.92	R	4.55	Swift	4.36	Ruby	3.04
209	2017.Q1	Java	24.84	JavaScript	22.88	Python	17.56	C#	8.99	PHP	8.99	C++	7.37	C	6.17	Objective	4.72	R	4.59	Swift	4.29	Ruby	2.99
210	2017.Q2	Java	24.61	JavaScript	22.97	Python	18.29	C#	8.89	PHP	8.76	C++	7.26	C	6.05	R	4.62	Objective	4.53	Swift	4.23	Ruby	2.95
211	2017.Q3	Java	24.42	JavaScript	22.98	Python	19.05	C#	8.84	PHP	8.56	C++	7.19	C	6.02	R	4.6	Objective	4.39	Swift	4.12	Ruby	2.9
212	2017.Q4	Java	24.24	JavaScript	22.99	Python	19.81	C#	8.79	PHP	8.37	C++	7.11	C	5.98	R	4.57	Objective	4.25	Swift	4.01	Ruby	2.86
213	2018.Q1	Java	24.05	JavaScript	23	Python	20.57	C#	8.74	PHP	8.18	C++	7.03	C	5.95	R	4.55	Objective	4.1	Swift	3.9	Ruby	2.82
214	2018.Q2	Java	23.86	JavaScript	23	Python	21.33	C#	8.69	PHP	7.98	C++	6.95	C	5.92	R	4.52	Objective	3.96	Swift	3.79	Ruby	2.78
215	2018.Q3	Java	23.52	JavaScript	23.03	Python	21.97	C#	8.63	PHP	7.81	C++	6.9	C	5.79	R	4.49	Objective	3.85	Swift	3.78	Ruby	2.76
216	2018.Q4	Java	23.18	JavaScript	23.05	Python	22.62	C#	8.57	PHP	7.64	C++	6.85	C	5.66	R	4.46	Swift	3.76	Objective	3.74	Ruby	2.73
217	2019.Q1	Python	23.27	JavaScript	23.08	Java	22.84	C#	8.51	PHP	7.46	C++	6.8	C	5.53	R	4.42	Swift	3.74	Objective	3.62	Ruby	2.71
218	2019.Q2	Python	23.91	JavaScript	23.1	Java	22.51	C#	8.45	PHP	7.29	C++	6.75	C	5.4	R	4.39	Swift	3.73	Objective	3.51	Ruby	2.69
219	2019.Q3	Python	24.09	JavaScript	23.14	Java	22.45	C#	8.43	PHP	7.24	C++	6.69	C	5.38	R	4.38	Swift	3.74	Objective	3.49	Ruby	2.64

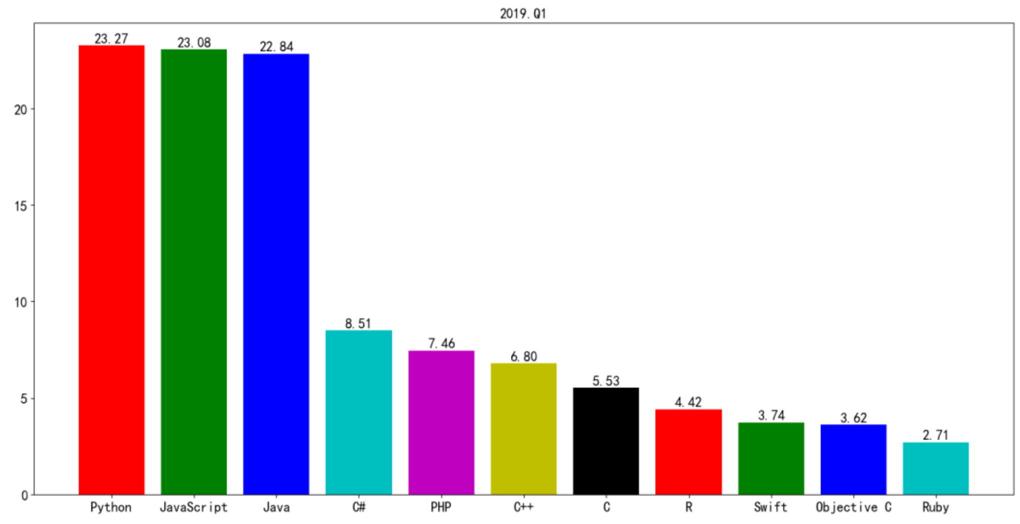
总共 1965.Q1 到 2019.Q3 总共 219 个季度。

红色的部分就是被猫头鹰遮挡的季度，只能手动修改。
其他的数据则不需要修改。

动态展示如下：

这里就随便截取几张图片：





变化分析：

这里主要分析几个曾经或者现在特别流行的语言。视频中的数据是从 1965 年开始统计的，刚开始我们可以看到排在第一位的是 Fortran，它是为科学、工程问题或企事业管理中的那些能够用数学公式表达的问题而设计的，其数值计算的功能较强。Fortran 在当时之所以这样流行是因为 Fortran 语言是世界上第一个被正式推广使用的高级语言。它是 1954 年被提出来的，1956 年开始正式使用，它始终是数值计算领域所使用的主要语言。Fortran 语言问世以来，根据需要几

经发展，先后推出了不同的版本，其中最流行的是 1958 年出现的 FortranⅡ和 1962 年出现的 FortranⅣ。FortranⅣ(即 Fortran 66)流行了十几年，几乎统治了所有的数值计算领域，许多应用程序和程序库都是用 FortranⅣ语言编写的。Fortran 的优势是简单好写，计算效率也很高，非常适合短小的计算程序。但对于很多大规模科学计算来说，核心部分往往是高强度的矩阵运算，不管用什么语言都需要高度优化、甚至拿 C 或者汇编根据体系结构特点来写。

在 1977 年左右，Pascal 语言增长迅速反超 COBOL 然后继续增长在 1980 年左右超过 Fortran。Pascal 语言是由瑞士 Niklaus Wirth 教授于六十年代末设计并创立的。Pascal 语言语法严谨，层次分明，程序易写，可读性强，是第一个结构化编程语言。严格的结构化形式；丰富完备的数据类型；运行效率高；查错能力强。 Pascal 语言还是一种自编译的语言，这就使它的可靠性大大提高了。Pascal 语言重大影响来自于程序设计教学，它的出现取代了 Fortran 作为程序设计入门语言的地位。

后来到了 1982 年 C 语言的出现到 1986 年 Ada 和 C 语言不分上下，都占百分之四十多。Ada 是一种程序设计语言。源于美国军方的一个计划，旨在整合美军事系统中运行着上百种不同的程序设计语言。Ada 不仅体现了许多现代软件的开发原理，而且将这些原理付诸实现。同时，Ada 语言的使用可大大改善软件系统的清晰性、可靠性、有效性、可维护性。Ada 是现有的语言中无与伦比的一种大型通用程序设计语言，它是现代计算机语言的成功代表，集中反映了程序语言研究的成果。Ada 的出现，标志着软件工程成功地进入了国家和国际的规模。Ada 在那个年代曾是美国国防部指定唯一可用于军用系统开发的语言。这也是当时它比较流行的原因之一。

1988 年以后到 1997 年 C 语言一直稳居第一，占到百分之 70 多。c 语言的主体设计完成后，Thompson 和 Ritchie 用它完全重写了 UNIX，在开发中，他们还考虑把 UNIX 移植到其他类型的计算机上使用。C 语言强大的移植性（Portability），C 语言程序则可以使用在任意架构的处理器上，只要那种架构的

处理器具有对应的 C 语言编译器和库，然后将 C 源代码编译、连接成目标二进制文件之后即可运行。且随着 UNIX 的发展，C 语言也得到了不断的完善。为了利于 C 语言的全面推广，许多专家学者和硬件厂商联合组成了 C 语言标准委员会，并在之后的 1989 年，诞生了第一个完备的 C 标准。直到今天，各种版本的 UNIX 内核和周边工具仍然使用 C 语言作为最主要的开发语言。C 语言优点：（1）简洁的语言（2）具有结构化的控制语句（3）丰富的数据类型（4）丰富的运算符（5）可对物理地址进行直接操作（6）代码具有较好的可移植性（7）可生成高质量、目标代码执行效率高的程序，另外和其他高级语言相比，C 语言可以生成高质量和高效率的目标代码，故通常应用于对代码质量和执行效率要求较高的嵌入式系统程序的编写。所以 C 语言当非常流行的另一个原因是在 20 世纪 80 年代，随着微电子工艺水平的提高，嵌入式系统的发展也非常迅速。20 世纪 90 年代，在分布控制、柔性制造、数字化通信和信息家电等巨大需求的牵引下，嵌入式系统进一步快速发展。

直到 1997 年 Java 出现，到 2001 年左右 java 已经排到第一 JavaScript 也紧跟在后面，

Java 是一门面向对象编程语言，不仅吸收了 C++语言的各种优点，还摒弃了 C++里难以理解的多继承、指针等概念，因此 Java 语言具有功能强大和简单易用两个特征。Java 语言作为静态面向对象编程语言的代表，极好地实现了面向对象理论，允许程序员以优雅的思维方式进行复杂的编程。同时又因为此时的 Android 系统在手机上广泛使用脑，许多的 Android 应用都是 Java 程序员开发者开发。虽然 Android 运用了不同的 JVM 以及不同的封装方式，但是代码还是用 Java 语言所编写。相当一部分的手机中都支持 JAVA 游戏，这就使 JAVA 更进一步的流行。JavaScript 是一种属于网络的脚本语言,已经被广泛用于 Web 应用开发,常用来为网页添加各式各样的动态功能,为用户提供更流畅美观的浏览效果。通常

JavaScript 脚本是通过嵌入在 HTML 中来实现自身的功能的。JavaScript 的流行很大一部分是跟 21 世纪互联网的快速发展有关。

大概 2012 年左右，Python 开始迅速增长。由于 Python 语言的简洁性、易读性以及可扩展性，在国外用 Python 做科学计算的研究机构日益增多，一些知名大学已经采用 Python 来教授程序设计课程。众多开源的科学计算软件包都提供了 Python 的调用接口，例如著名的计算机视觉库 OpenCV、三维可视化库 VTK、医学图像处理库 ITK。而 Python 专用的科学计算扩展库就更多了，例如如下 3 个十分经典的科学计算扩展库：NumPy、SciPy 和 matplotlib，它们分别为 Python 提供了快速数组处理、数值运算以及绘图功能。因此 Python 语言及其众多的扩展库所构成的开发环境十分适合工程技术、科研人员处理实验数据、制作图表，甚至开发科学计算应用程序。而最几年 Python 突然流行的原因主要有以下几个关键节点。12~14 年是云计算最火的几年，大批创业公司和巨头进军云计算领域，大家都在做 IAAS，最著名的云计算开源平台 OpenStack 就是基于 Python 开发的，为此催生出不少 Python 岗位。14~15 年 O2O、P2P 产品如雨后春笋般冒出，Python 适合快速搭建原型。16~17 年人工智能成为热门话题而人工智能、机器学习的首选语言就是 Python。同时大数据的发展也使得 Python 更加被普遍使用。数据挖掘、分析、机器学习、人工智能都需要大数据的支撑，而真正有大数据的厂商没几个，所以小厂不得不通过爬虫去获取数据。而爬虫也需要用到 Python。

从以上分析可以预测，每一种曾经排在过第一的语言大多会持续一段较长的时间，这都与当时世界计算机水平的发展息息相关，也与当时热门的研究话题有关。而且排在第一的语言大都会领先排名第二很多。所以从目前 Python、Java、JavaScript 排在前三各占百分之二十多来看，Python 应该会继续增长至第一，且会与第二名拉开较大差距。并以此持续一段时间直到计算机有新的研究热门话题出现或者一种新的语言诞生。