实习任务总结

1.我所完成的

（1）对网站的访问并且提取html信息(通过get方法)

```
func RequestGet(url string) (html string) {
    /*完成信息抓取*/
    method := "GET"
    client := &http.Client{}
    req, _ := http.NewRequest(method, url, nil)
    res, _ := client.Do(req)
    defer res.Body.Close()

    body, _ := ioutil.ReadAll(res.Body)
    html = string(body)
    return html
}
```

（2）实现对html信息的正则转换

```
func Change(html string) (htmlchange string) {
    /* 正则转换*/
    src := html
    //将HTML标签全转换成小写
    re, _ := regexp.Compile("\\<[\\S\\s]+?\\>")
    src = re.ReplaceAllStringFunc(src, strings.ToLower)

    //去除STYLE
    re, _ = regexp.Compile("\\<style[\\S\\s]+?\\</style\\>")
    src = re.ReplaceAllString(src, "")

    //去除SCRIPT
    re, _ = regexp.Compile("\\<script[\\S\\s]+?\\</script\\>")
    src = re.ReplaceAllString(src, "")

    //去除所有尖括号内的HTML代码，并换成换行符
    re, _ = regexp.Compile("\\<[\\S\\s]+?\\>")
    src = re.ReplaceAllString(src, "\n")

    //去除连续的换行符
    re, _ = regexp.Compile("\\s{2,}")
    src = re.ReplaceAllString(src, "\n")

    htmlchange = strings.TrimSpace(src)
    return htmlchange
}
```

（3）实现提取html中的url,文章简介,热度等信息功能

```
func GetTheUrl(html string) (url []string, content []string, hot []string) {
    /*提取html中的url,文章简介,热度等信息*/
    dom, err := goquery.NewDocumentFromReader(strings.NewReader(html))
    if err != nil {
        log.Fatalln(err)
    }

    dom.Find("td[class='al']>a").Each(func(i int, selection *goquery.Selection)
{
        href, _ := selection.Attr("href")
        url = append(url, "https://tophub.today"+href)
    })
    dom.Find("td[class='al']").Each(func(i int, selection *goquery.Selection) {
        con := selection.Text()
        content = append(content, con)
    })
    dom.Find("td[class='al']+td").Each(func(i int, selection *goquery.Selection)
{
        h := selection.Text()
        hot = append(hot, h)
    })
    return url, content, hot
}
```

(4) 实现邮件的发送功能

```
func Sendmail(body string) {

    /*发送邮件*/
    identity := ""
    sender := "2038975825@qq.com"
    pwd := "dumqnhmcxthbbgfg"
    host := "smtp.qq.com"
    port := "25"
    sendTo := []string{"2038975825@qq.com"}
    senderName := "ZhiHuHot"
    title := "ZhiHuHot"
    auth := smtp.PlainAuth(identity, sender, pwd, host)
    content_type := "Content-Type: text/html; charset=UTF-8"
    msg := []byte("To: " + strings.Join(sendTo, ",") + "\nFrom: " + senderName +
        "<" + sender + ">\nSubject: " + title + "\n" + content_type + "\n" +
body + "\n")

    url := host + ":" + port
    err := smtp.SendMail(url, auth, sender, sendTo, msg)
    if err != nil {
        fmt.Printf("\n\nsend mail error: %v", err)
        return
    }
    fmt.Println("\n\nsend mail succes")
    return
}
```

(5) 实现写入文件功能

```
func Write(content []string, hot []string, urlofhtml []string) {
```

```
    f, err := os.Create("test.txt")
    if err != nil {

        fmt.Println(err)
        return
    }
    for i := 0; i < 10; i++ {
        _, _ = f.WriteString(content[i])
        _, _ = f.WriteString("\n")
        _, _ = f.WriteString(hot[i])
        _, _ = f.WriteString("\n")
        _, _ = f.WriteString(urlofhtml[i])
        _, _ = f.WriteString("\n")
    }
    err = f.Close()
    if err != nil {
        fmt.Println(err)
        return
    }
}
```

(6) 实现对第三步提取的url进行再次访问并且获取文章内容

```
func Done() {
    /*启动函数*/
    url := "https://tophub.today/n/mproPpoq6O"

    /*对知乎发送请求并且获取html信息*/
    html := RequestGet(url)

    /*发送邮件，直接发送html信息，qq邮箱会对其自动进行渲染*/
    Sendmail(html)

    /*对html信息进行分析，获取上面的url，热度，内容等信息*/
    urlofhtml, content, hot := GetTheUrl(html)
    /*写入文件*/
    Write(content, hot, urlofhtml)

    /*对从html中得到的url信息再次发送请求并且获取html信息*/
    AllArticle := ""
    for j := 0; j < 3; j++ {
        newhtml := RequestGet(urlofhtml[j])
        fmt.Println(Change(newhtml))
        AllArticle += Change(newhtml)
    }
    Sendmail(AllArticle)

}
```

(7) 实现定时启动

```
func main() {
    /*主函数*/
    for i:=0;i<30;i++{
        Done()
        fmt.Println("sleep")
        /*每12小说定时启动*/
        time.Sleep(time.Hour * 12)
    }
}
```

（8）一个副产品，通过广度优先搜索提取网站所有url链接及链接对应网站包含的所有url信息

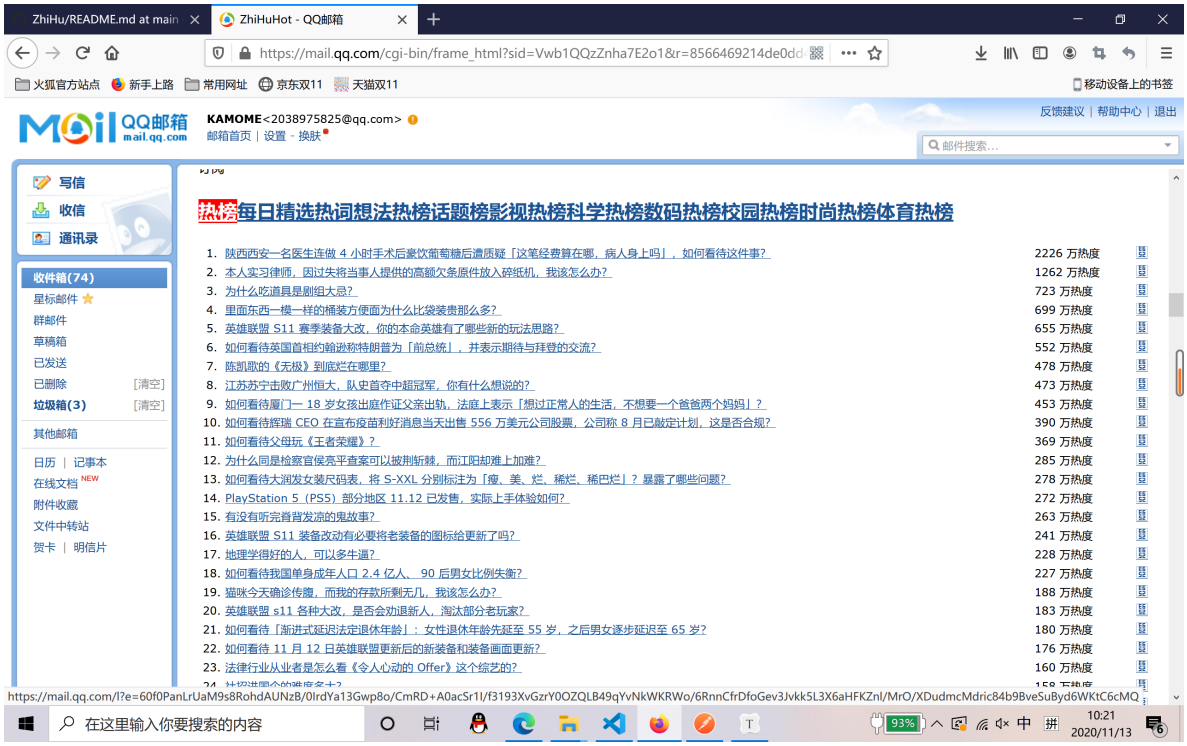
```
package main

import (
    "fmt"
    "io/ioutil"
    "net/http"
    "regexp"
    "strings"
    "time"
)

var href_reg *regexp.Regexp

var hrefs_been_found map[string]int

var hrefs_undone []string

func get_all_href(url string) []string {
    var ret []string
    resp, err := http.Get(url)
    if err != nil {
        fmt.Println(err)
        return ret
    }
    defer resp.Body.Close()
    body, _ := ioutil.ReadAll(resp.Body)

    hrefs := href_reg.FindAllString(string(body), -1)

    for _, v := range hrefs {
        str := strings.Split(v, "\"")[1]

        if len(str) < 1 {
            continue
        }

        switch str[0] {
        case 'h':
            ret = append(ret, str)
        case '/':
            if len(str) != 1 && str[1] == '/' {
                ret = append(ret, "http:"+str)
```

```
            }

            if len(str) != 1 && str[1] != '/' {
                ret = append(ret, url+str[1:])
            }
        default:
            ret = append(ret, url+str)

        }

    }

    return ret
}

func init_global_var() {
    href_pattern := "href=\"(.+?)\""
    href_reg = regexp.MustCompile(href_pattern)

    hrefs_been_found = make(map[string]int)
}

func is_href_been_found(href string) bool {
    _, ok := hrefs_been_found[href]
    return ok
}

func add_hrefs_to_undone_list(hrefs []string) {
    for _, value := range hrefs {
        ok := is_href_been_found(value)
        if !ok {
            fmt.Printf("new url:(%s)\n", value)
            hrefs_undone = append(hrefs_undone, value)
            hrefs_been_found[value] = 1
        } else {
            hrefs_been_found[value]++
        }

    }
}

func main() {
    init_global_var()

    var pos = 0
    var urls = []string{"https://tophub.today/n/mproPpoq6O"}
    add_hrefs_to_undone_list(urls)

    for {
        if pos >= len(hrefs_undone) {
            break
        }
        url := hrefs_undone[0]
        hrefs_undone = hrefs_undone[1:]

        hrefs := get_all_href(url)
        add_hrefs_to_undone_list(hrefs)
        time.Sleep(time.Second / 10)
```

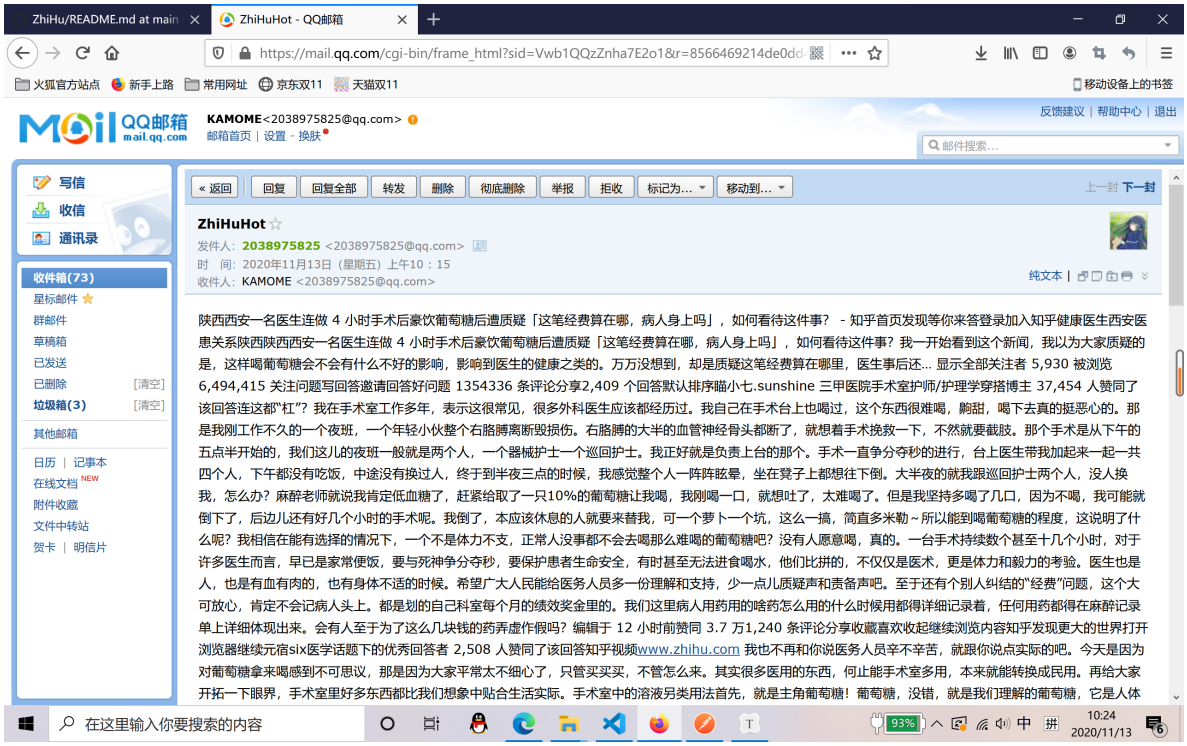
```
    }
    fmt.Println("end")
}
```

2.最终实现效果

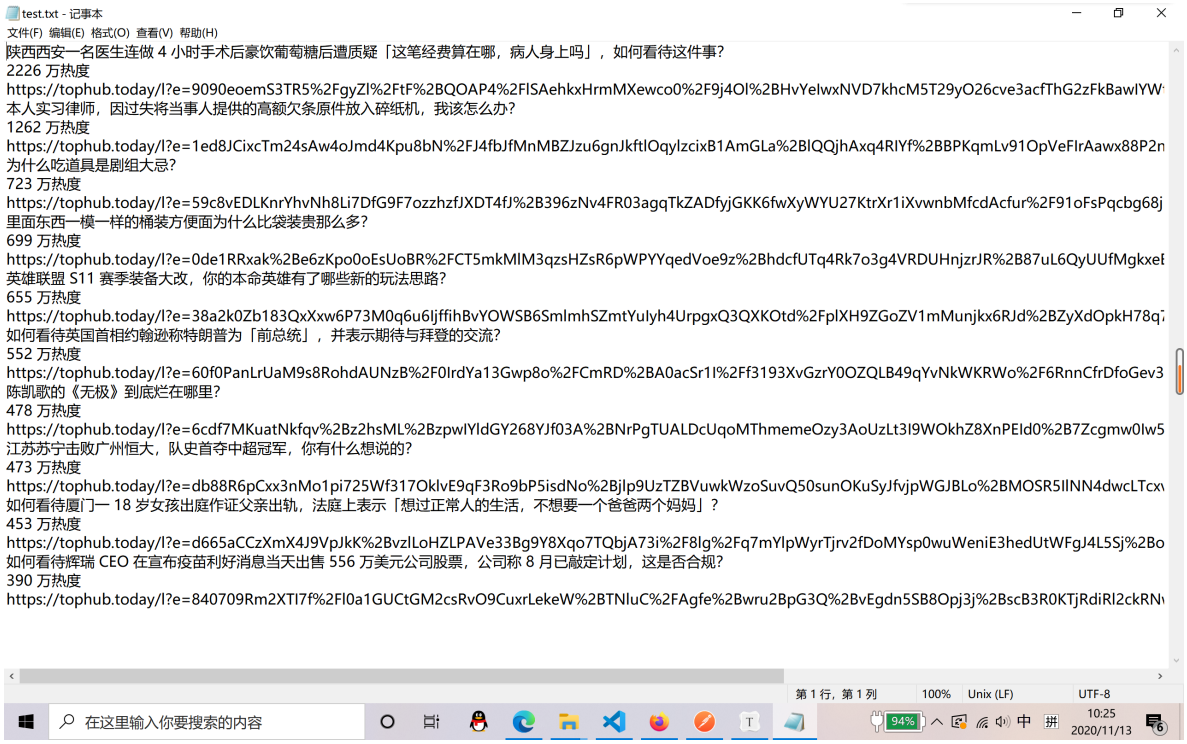
（1）通过直接发送html信息，qq邮箱自动渲染得到的结果



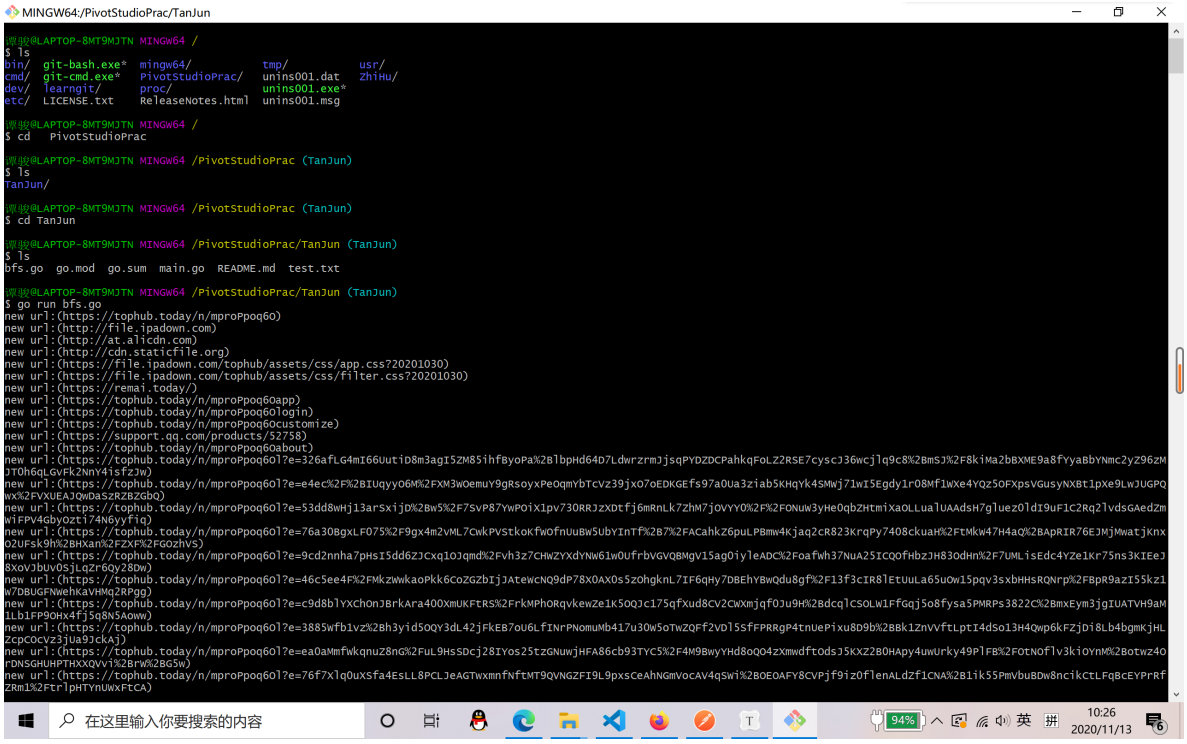
（2）通过发送进行正则表达式处理后热榜前五的高赞回答结果



（3）写入文本的结果（包含热度，概要，链接等信息）



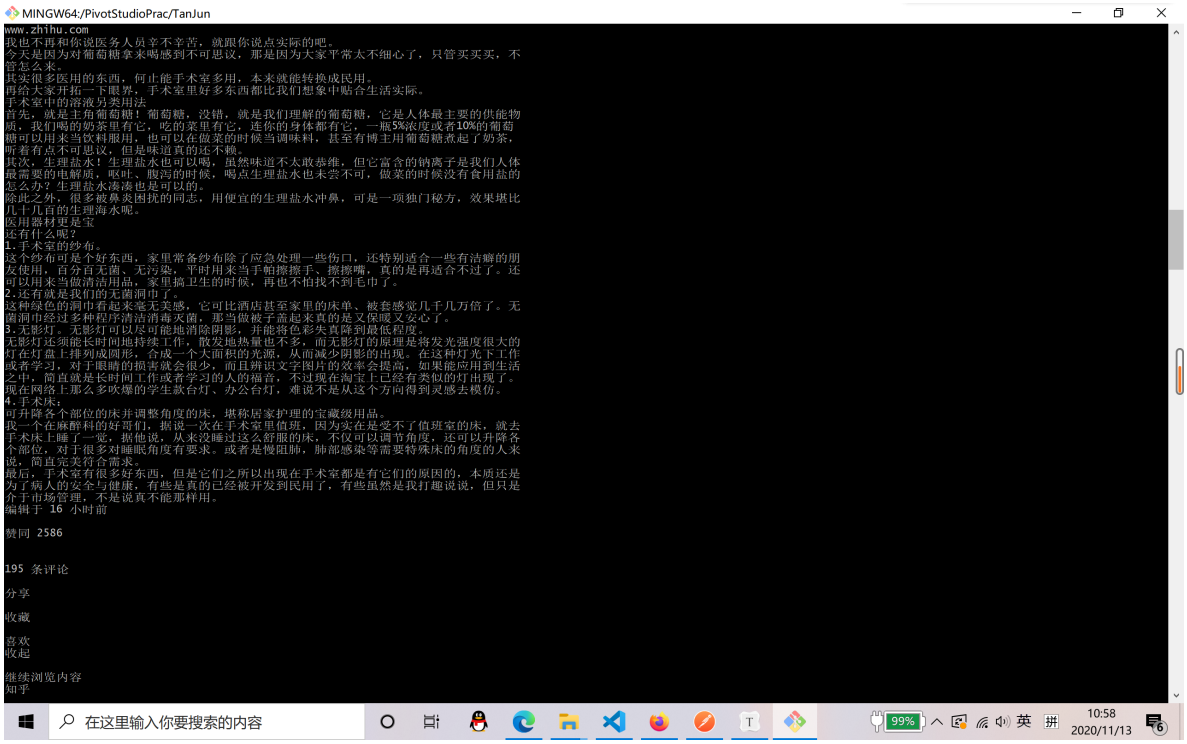
（4）广度优先获取全部url信息的结果（以下还有很多）



3.我的不足

（1）go mod出问题，导致无法导入本地包，只能把所有的函数写在一个main里面。

（2）对于正则化处理后的内容，在命令行界面可以正常换行和排版，但是打包成邮件发送就变得很混乱。（命令行界面如下）



（3）多次访问ip地址会被网站登记，加入cookie等也无法解决，需要手动输入验证码

（4）没有采用数据库管理数据，信息只是简单地存在txt文件里面。

（5）没有采用读入的方式来传入邮箱信息，容易导致信息泄露。

（6）直接发送的url信息可能会过期导致访问不上。