2020.7.15

使用java实现的选课系统，链表保存数据，数据存储在文件中。

下面是将链表数据序列化到文件中的demo

1. studentsList.add(**new** Students(id,name,sex,birthday));
2. System.out.println("successfully added");
3. ObjectOutputStream oos=**new** ObjectOutputStream(
4. **new** FileOutputStream(**new** File("src//phase\_1//data//students.dat")));
6. **for** (Students students : studentsList) {
7. oos.writeObject(students);
8. }
9. oos.flush();

读取数据，还是以学生为例：

1. List<Students> studentsList=**new** ArrayList<>();
3. ObjectInputStream ois=**new** ObjectInputStream(
4. **new** FileInputStream(**new** File("src//phase\_1//data//students.dat")));
5. **while** (**true**){
6. **try** {
7. studentsList.add((Students)ois.readObject());
8. }
9. **catch** (IOException | ClassNotFoundException e){
10. System.out.println("成功读取学生信息");
11. **break**;
12. }
13. }

2020.7.16

百度ueditor富文本编辑器

Head中内容

1. <!-- 样式文件 -->
2. **<link** rel="stylesheet" href="./umeditor/themes/default/css/umeditor.css"**>**
3. <!-- 引用jquery -->
4. **<script** src="./umeditor/third-party/jquery.min.js"**></script>**
5. <!-- 配置文件 -->
6. **<script** type="text/javascript" src="./umeditor/umeditor.config.js"**></script>**
7. <!-- 编辑器源码文件 -->
8. **<script** type="text/javascript" src="./umeditor/umeditor.js"**></script>**
9. <!-- 语言包文件 -->
10. **<script** type="text/javascript" src="./umeditor/lang/zh-cn/zh-cn.js"**></script>**
11. <!-- 实例化编辑器代码 -->
12. **<script** type="text/javascript"**>**
13. $(function(){
14. window.um = UM.getEditor('container', {
15. /\* 传入配置参数,可配参数列表看umeditor.config.js \*/
16. toolbar: ['undo redo | bold italic underline']
17. });
18. });
19. **</script>**

文本中插入即可

2020.7.17

Html字体粗细

**<p** style="font-weight: 100"**>**最多添加10个多图产品，每个多图产品最多展示6张图片**</p>**

[Flask向模板中JS传值简便方式](http://139.196.168.156/2020/07/03/flask%e5%90%91%e6%a8%a1%e6%9d%bf%e4%b8%adjs%e4%bc%a0%e5%80%bc%e7%ae%80%e4%be%bf%e6%96%b9%e5%bc%8f/)

后台传值

return render\_template(‘statistics/numberofuserlogin/login\_number.html’, result\_json = json.dumps(result))

模板中JS取值：

js\_object = eval(‘{{result\_json|safe }}’)

2020.7.18

Flask+heroku+gunicorn的网站部署

http://139.196.168.156/2020/07/05/flaskherokugunicorn%e7%9a%84%e7%bd%91%e7%ab%99%e9%83%a8%e7%bd%b2/

1、配置项目

既然是要用到gunicorn部署，就需要配置gunicorn的运行

在项目根目录中新建一个文件Procfile，注意，是没有后缀的，文件内容为

web: gunicorn app:app --log-file -

然后将项目上传到github

2、在heroku上部署

打开网页，点击创建应用，选择部署，连接github上的项目，然后选择自动部署，然后点击手动部署

3、测试

点击打开应用打开自己的主页，如果出现问题可以查看log，根据相应问题的描述选择相应的解决方案

2020.7.19

更换安卓模拟器路径

<http://139.196.168.156/2020/06/21/%e6%9b%b4%e6%8d%a2%e5%ae%89%e5%8d%93%e6%a8%a1%e6%8b%9f%e5%99%a8%e8%b7%af%e5%be%84/>

修改系统变量

变量名：ANDROID\_SDK\_HOME

变量值：存放avd的路径，如：D:\AVD

在Android Studio中运行模拟器时，提示Error: Not enough disk space to run AVD ‘….’. Exiting。是说安装模拟的磁盘空间不足，导致无法运行模拟器。

Android Studio 安装之后，默认的模拟器会安装在C盘中，可以打开模拟器管理界面，右键查看（Show on disk）模拟器安装的目录。如下图：

我的模拟器安装路径为： C:\Users\Administrator\.android\avd， 其中的avd文件夹就是用来存放模拟器镜像文件的。

第一步：需要将Pixel\_API\_27.avd的文件夹移出C盘，放入空间足够的磁盘中。

第二步：就是要修改Pixel\_API\_27.ini（在此就称为模拟器配置文件吧，我也是新手不太懂）文件中模拟器的路径。

然后重新运行模拟器，没有问题应该可以运行起来。

2020.7.20

leetcode\_最佳观光组合

<http://139.196.168.156/2020/06/24/leetcode_%e6%9c%80%e4%bd%b3%e8%a7%82%e5%85%89%e7%bb%84%e5%90%88/>

题目描述

给定正整数数组 A，A[i] 表示第 i 个观光景点的评分，并且两个景点 i 和 j 之间的距离为 j – i。

一对景点（i < j）组成的观光组合的得分为（A[i] + A[j] + i – j）：景点的评分之和减去它们两者之间的距离。

返回一对观光景点能取得的最高分。

题解

官方题解如下：

我们考虑从前往后枚举 jj 来统计答案，对于每个观光景点 jj 而言，我们需要遍历 [0,j-1][0,j−1] 的观光景点 ii 来计算组成观光组合 (i,j)(i,j) 得分的最大值 \textit{cnt}\_jcntj​ 来作为第 jj 个观光景点的值，那么最后的答案无疑就是所有观光景点值的最大值，即 \max\_{j=0..n-1}\{cnt\_j\}maxj=0..n−1​{cntj​}。但是枚举 jj 需要 O(n)O(n) 的时间复杂度，遍历 [0,j-1][0,j−1] 的观光景点 ii 也需要 O(n)O(n) 的时间复杂度，因此该方法总复杂度为 O(n^2)O(n2)，不能通过所有测试用例，我们需要进一步优化时间复杂度。

我们回过头来看得分公式，我们可以将其拆分成 A[i]+iA[i]+i 和 A[j]-jA[j]−j 两部分，这样对于统计景点 jj 答案的时候，由于 A[j]-jA[j]−j 是固定不变的，因此最大化 A[i]+i+A[j]-jA[i]+i+A[j]−j 的值其实就等价于求 [0,j-1][0,j−1] 中 A[i]+iA[i]+i 的最大值 mxmx，景点 jj 的答案即为 mx+A[j]-jmx+A[j]−j 。而 mxmx 的值我们只要从前往后枚举 jj 的时候同时维护即可，这样每次枚举景点 jj 的时候，寻找使得得分最大的 ii 就能从 O(n)O(n) 降至 O(1)O(1) 的时间复杂度，总时间复杂度就能从 O(n^2)O(n2) 降至 O(n)O(n)。class Solution { public int maxScoreSightseeingPair(int[] A) { int ans = 0, mx = A[0] + 0; for (int j = 1; j < A.length; ++j) { ans = Math.max(ans, mx + A[j] – j); // 边遍历边维护 mx = Math.max(mx, A[j] + j); } return ans; } }

class Solution {

public int maxScoreSightseeingPair(int[] A) {

int ans = 0, mx = A[0] + 0;

for (int j = 1; j < A.length; ++j) {

ans = Math.max(ans, mx + A[j] - j);

// 边遍历边维护

mx = Math.max(mx, A[j] + j);

}

return ans;

}

}

官方题解其实讲解得晦涩难懂，其实这道题可以用更简洁的语言来说明

抓住一个重点，i<j。为什么这是重点，因为如果你写了一个双层的循环，那么一定会超时，如何用一个循环解决问题，我们可以抓住i和j的先后关系，将他们的迭代写在一个循环里，j跟随循环刷新，但最终结果res可以滞后更新。

总结来说就是 A[j]-j这两个包含j的变量不单独拎出来，作为变量随着循环改变。A[j]-j中的j作为i的迭代使用，将这两个变量提出单独更新。代码如下：

A=[8,1,5,2,6]

i\_max=0

j\_max=0

res=0

for j in range(0,len(A)):

res=max(res,i\_max+A[j]-j)

i\_max=max(A[j]+j,i\_max)

print(res)

2020.7.21

搭建自己的LNMP服务器

<http://139.196.168.156/2020/06/21/%e6%90%ad%e5%bb%ba%e8%87%aa%e5%b7%b1%e7%9a%84lnmp%e6%9c%8d%e5%8a%a1%e5%99%a8/>

这里使用的是majaro系统

1、安装nmp

sudo pacman -S nginx php-fpm mariadb php-gd

2、查看本机外网ip

ip address

3、访问一下试试，默认端口是80

nginx配置文件的默认目录是 etc/nginx

4、启动服务

systemctl start nginx

systemctl start php-fpm

systemctl enable nginx

systemctl enable mysqld

systemctl enable php-fpm

外网访问一下试试，发现没法连接

打开ssh服务

systemctl start sshd.service

5、设置内网穿透

使用sukura frp

linux安装命令如下

wget https://qianqu.me/frp/frpc\_linux\_amd64 —no-check-certificate

chmod +x frpc\_linux\_amd64

./frpc\_linux\_amd64

去官网找到自己的密匙并输入

或者 ./frpc\_linux\_amd64 -t 访问密钥 -s 服务器ID 设置开机自启