1. 需求

输入IP，通过查询IP地址库，得到国家，省份和城市。

1. 背景信息

2.1 解释一个名词：

MVGeoID是一个10位长的整型数字。用于表示国家，省份和城市。格式如下：

1156130400： 前四个数字代表国家，1156是中国； 中间两个数字代表省份，13是河北省；后面四个数字代表城市0400是邯郸。因此1156130400代表中国河北省邯郸市。

1392000000：表示日本。

2.2 我们目前有两个excel文档：

2.2.1 IP(20160905).csv： 保存了每个ip地址段，对应的MVGeoID。 例如：

|  |  |  |
| --- | --- | --- |
| 1.2.17.0 | 1.2.127.255 | 1156440100 |

该文件有16万多条数据，对于中国能够细化到城市，对于海外，只能到国家范围。

* + 1. ip\_code.xlsv：保存了每个MVGeoID对应的国家，省份，城市信息。例如：

|  |  |  |
| --- | --- | --- |
| 全球 | 布隆迪 | 1108000000 |
| 河北省 | 承德市 | 1156130800 |

通过这两个文件，我们就可以根据IP地址获取到地理位置信息

1. 设计方案

3.1 使用一个二进制文件（ipadr.dat）存储ip和MVGeoID的对应关系。

二进制文件包含两个部分。

第一部分1024个字节长度，包含了256个整型（一个整型4个字节），从第一个开始存储头字节对应的ip地址段的数量。例如：

|  |
| --- |
| 0.0.0.0，1.0.0.255，1000000000  1.0.1.0, 1.0.3.255,1156350100 |
| 1.0.4.0,1.0.7.255,1036000000 |
| 1.0.8.0,2.0.15.255,1156440100 |
| 对于0开头的地址段，数量为1（0.0.0.0~0.255.255.255）  对于1开头的地址段，数量为4（1.0.0.0~1.0.0.255；1.0.1.0~1.0.3.255；1.0.4.0~1.0.7.255；1.0.8.0~1.255.255.255） |

第二部分保存了每个地址段对应的MVGeoID的数据。两个整型为一组，第一个整型是IP地址段的截止IP对应的整型，比如0.0.0.0~0.255.255.255，就是取inet\_aton(0.255.255.255)的值；第二个整型就是MVGeoID，即1000000000.

第二部分是按照IP地址段从小到大的顺序依次存储了每个IP地址段所对应的MVGeoID。

* 1. 基于二进制文件的存储格式，可以编写算法来获取每个IP对应的MVGeoID，具体如下：

1. 在load（）方法里，先加载文件的前1024个字节，计算出从0到255开头的ip地址段分别对应的字段长度。
2. 在find（）方法里，通过给出的ip参数，通过第一个ip数字，先定位到第二部分的offset上，然后逐一比对第一个字节，找到第一个比输入ip大的地址，则返回第二个整型，即MVGeoID
3. 背景知识

4.1 python的struct库使用

struct提供了对于二级制数据的处理方法，struct.unpack(‘256I’,binary[0:1024])，其中binary是一个二级制数组，该函数解析binary的前1024个字节，并将其转化为256个Integer，返回的是长度为256的整型数组。

tmp =inet\_aton(ip)

Struct.pack(‘4c’,\*tmp),将一个字符串转成4个char

* 1. inet\_aton():该方法降一个ip地址转换为32位的整形。如192.168.33.123对应的二进制位 11000000 10101000 00100001 01111011，将其合在一起即为一个32位的整型数了。
  2. Little-Endian & Big-Endian

不同的CPU有不同的字节序类型，这些字节序是指整数在内存中保存的顺序。

1． Little-endian：将低序字节存储在起始地址（低位编址）

2． Big-endian：将高序字节存储在起始地址（高位编址）

*例子2*：如果我们将0x1234abcd写入到以0x0000开始的内存中，则结果为

　　　　big-endian   little-endian

0x0000    0x12         0xcd

0x0001    0x23         0xab

0x0002    0xab         0x34

0x0003    0xcd         0x12

因此，如果用ultraedit打开看，从左到右的byte正好跟数字的格式一致，则为BE