python必知模板

- **os模块**
  系统编程的模块，可以处理文件目录等

  - os.sep 更改操作系统中的路径分隔符。
  - os.getcwd()获取当前路径，这个在Python代码中比较常用。
  - os.listdir() 列出当前目录下的所有文件和文件夹。
  - os.remove() 方法可以删除指定的文件。
  - os.system() 方法用来运行shell命令。
  - os.chdir() 改变当前目录，到指定目录中。
- **time模块**
  涉及时间的操作
  - 表示时间的方式
    - 时间戳
    - 格式化的时间str（字符串）
    - 元组 (struct_time)以及calendar

[索引]	[属性]	[值]
0	tm_year(年)	比如：2013
1	tm_mon(月)	1~12
2	tm_mday(日)	1~31
3	tm_hour(小时)	0~23
4	tm_min(分钟)	0~59
5	tm_sec(秒)	0~59
6	tm_wday(周)	0~6 (0是周日，依此类推)
7	tm_yday	(一年中第几天，1~366)
8	tm_isdst(夏令制)	(默认为-1)
  - 常用函数
    - 1.time.localtime([secs])：这个函数的作用是将时间戳，转换成当前时区的时间结构，返回的是一个元组。secs参数如果没有提供的话，系统默认会以当前时间做为参数。
    - 2.time.time() 这个模块的核心之一就是time()，它会把从纪元开始以来的秒数作为一个float值返回。
    - 3.time.ctime() 将一个时间戳，转换为一个时间的字符串。
    - 4.time.sleep() 经常在写程序的想让程序暂停一下再运行，这个时间sleep()方法就派上用场了，它可以让程序休眠，参数是以秒计算。
    - 5.time.clock() 返回浮点数，可以计算程序运行的总时间，也可以用来计算两次clock()之间的间隔。
    - 6.time.strftime() 将strume_time这个元组，根据你规定的格式，输邮字符串
- **re模块**
  正则表达式
  - **match object**对象
    search和match匹配成功后形成的对象
    - 方法

- group() 返回被 RE 匹配的字符串 #group（）返回re整体匹配的字符串 #group (n,m) 返回组号为n，m所匹配的字符串，如果组号不存在，则返回indexError异常
- start() 返回匹配开始的位置
- end() 返回匹配结束的位置
- span() 返回一个元组包含匹配 (开始,结束) 的位置
- groups() 方法返回一个包含正则表达式中所有小组字符串的元组，从 1 到所含的小组号，通常groups()不需要参数，返回一个元组，元组中的元就是正则表达式中定义的组。

- 常使用字符
  - 元字符 #反斜杠后的元字符变为普通字符
    - . 匹配任意（除"\n"） a.c abc
    - \ 转义 元-》普通字符 *a\.c;a\|c a.c;a|c*
    - * 匹配字符0或多次 abc* ab;abccc
    - + 匹配字符1次或无限次 abc+ abc;abccc
    - ? 匹配个字符0次或1次 abc? ab;abc
    - ^ 匹配字符串开头 ^abc abc
    - $ 匹配字符串末尾 abc$ abc
    - | 匹配|左右表达式任意一个，从左到右，没有()，范围整个正则表达式 abc|def abc def
    - {} {m}匹配前一个字符m次，{m,n}匹配前一个字符m至n次，若省略n，则匹配m至无限次 ab{1,2}c abc abbc
    - [] 字符集。可以是任意字符。可以逐个列出，也可给出范围，如[abc]或[a-c]。[^abc]表示取反，即非abc。特殊字符在字符集中失去特殊含义。\反斜杠恢复特殊含义。 a[bcd]e abe ace ade
    - () 分组，有编号 (abc){2}； a(123|456)c abcabc； a456c
  - 预定义字符集 #斜杠后字母变为特殊字符
    - \d 数字:[0-9] a\bc a1c
    - \D 非数字:[^\d] a\Dc abc
    - \s 匹配任何空白字符:[<空格>\t\r\n\f\v] a\sc a c
    - \S 非空白字符:[^\s] a\Sc abc
    - \w 匹配包括下划线在内的任何字字符:[A-Za-z0-9_] a\wc abc
    - \W 匹配非字母字符，即匹配特殊字符 a\Wc a c
    - \A 仅匹配字符串开头,同^ \Aabc abc
    - \Z 仅匹配字符串结尾，同$ abc\Z abc
    - \b 匹配\w和\W之间，即匹配单词边界匹配一个单词边界，也就是指单词和空格间的位置。例如， 'er\b' 可以匹配"never" 中的 'er'，但不能匹配 "verb" 中的 'er'。 \babc\ba\b!bc 空格abc空格a!bc
    - \B [^\b] a\Bbc abc
  - 特殊分组用法
    - (?P<name>) 分组，除了原有的编号外再指定额外的别名 (?P<id>abc){2} abcabc
    - (?P=name) 引用别名为<name>的分组匹配到字符串 (?P<id>\d)abc(?P=id) 1abc15abc5

        - \<number>　引用编号为<number>的分组匹配到字符串　　(\d)abc\1 1abc15abc5
    - 常用功能函数
        - **compile（）** #编译正则表达式 #返回一个对象
            - *re.compile(pattern #匹配的字符串,flags=0 #标志位)*
        - **match()** #决定RE是否在字符串刚开始的位置匹配 #不是完全匹配
            - *re.match(pattern ，　string , flags=0)*
        - **search（）** #查找模式匹配找到第一个匹配然后返回 #没有匹配返回None。
            - *re.search(pattern, string, flags=0)*
        - **re.findall（）** **#遍历全部 #获得所有匹配的字符串 #返回列表**
            - *re.findall(pattern, string, flags=0)*
        - **finditer()** **#找到所有匹配的字符 #作为一个迭代器返回**
            - *re.finditer(pattern, string, flags=0)*
        - **split（）** **#匹配后分割字符串**
            - re.split(pattern, string, maxsplit #分割的最大次数)
        - **sub()** **#替换每个匹配的字符串**
            - re.sub(pattern，repl #替换的内容，string，count #替换次数默认全部替换)
        - **subn()** **#添加返回替换次数**
            - subn(pattern,repl,string,count=0,flag=0)
        - **flag的**可能取法
            - re.S(DOTALL)　使.匹配包括换行在内的所有字符
            - re.I（IGNORECASE）　　使匹配对大小写不敏感
            - re.L（LOCALE）　　做本地化识别（locale-aware)匹配，法语等
            - re.M(MULTILINE)　多行匹配，影响^和$
            - re.X(VERBOSE) 该标志通过给予更灵活的格式以便将正则表达式写得更易于理解
            - re.U 根据Unicode字符集解析字符，这个标志影响\w,\W,\b,\B
- **pickle模块**

  持久化储存数据
    - Pickle对象串行化

      将任意一个Python对象转换成一系统字节的这个操作过程叫做串行化对象。
    - Pickle与CPickle对比

      前者是用Python来实现的模块，CPickle用C来实现，它的速度要比pickle快，建议有CPickle应该使用它。
    - 函数
        - **pickle.dump()**
            - pickle.dump(对象, 文件，[使用协议])
            - 索引0为ASCII，1是旧式2进制，2是新式2进制协议，不同之处在于后者更高效一些。默认的话dump方法使用0做协议。
        - **pickle.load()**
            - pickle.load(文件)

        - 从“文件”中，读取字符串，将它们反序列化转换为Python的数据对象，可以正常像操作数据类型的这些方法来操作它们。

- **random模块**

  生成随机浮点数、整数、字符串，随机选择列表序列中的一个元素，打乱一组数据

  - 函数
    - **random.random()** 返回0<=n<1之间的随机实数n；
    - **random.choice(seq)** 从序列seq中返回随机的元素；
    - **random.getrandbits(n)** 以长整型形式返回n个随机位；
    - **random.shuffle(seq[, random])** 原地指定seq序列；
    - **random.sample(seq, n)** 从序列seq中选择n个随机且独立的元素；
    - **random.uniform()**它可以设定浮点数的范围，一个是上限，一个是下限。
    - **random.randint()**随机生一个整数int类型，可以指定这个整数的范围，同样有上限和下限值

- **scoket模块**

  功能是在2个程序之间建立信息的通道

  - socket包括2个套接字，一个是服务器端(server)，一个是客户端(client)。在一个程序中创建服务器端的套接字，让它等客户端的连接，这样它就在这个IP和端口处，监听。
  - 方法
    - 一个是send，另一个是recv，它们用来传输数据信息。

- **math模块**
  - 常量
    - math.pi
    - math.e
  - 函数
    - **math.ceil(i)** #这个方法对i向上取整
    - **math.floor(i)** #是向下取整
    - **math.pow(a, b)** #返回a的b次方
    - **math.sqrt(i)** #返回i的平方根
-