作业讲解1：

```
def substring(strs):
    n=len(set(strs))
    while True:
        substring_list=[strs[i:i+n] for i in range(len(strs)) if i<len(strs)-n+1 and len(strs[i:i+n])==n]
        if substring_list:
            break
        n-=1
    return set(substring_list),n
```

共享利用set，可到出所有组合，一个个判断。

共再过滤重复的，即可得到不含重复字符的最大子串。

2. 求任意n连续元素最大平均值和其中元素。

```
def max_avg(lista,k):
    sub_list=[lista[i:i+k] for i in range(len(lista)) if i<len(lista)-k+1]
    sum_list=(list(map(lambda x:sum(x),sub_list)))/k  # map可以传入一个列表或多个列表
    max1=max(sum_list)
    maxV=sub_list[sum_list.index(max1)]
    print(max1/k,max_sub_list)
```

numpy 统计函数：平均值、方差、最大最小值。

对应有amax

```
print(np.amin(a,0))  按列
print(np.amin(a,1))  按行
```

```
len(set(strs[i:i+n]))
print(np.amax(a1))  找为个值
```

np.mean 均值 算术平均值

np.std 标准差

np.var 方差

numpy sort() 得到后排

```
a=np.array([[3,7],[9,1]])
print(a)
print(np.sort(a,axis=0))
```

axis=0 按行

```
print(np.sort(a))  默认按行
```

3. 反转一个整数

①

```
strs='abcd'
print(strs[::-1])  # dcba
```

推荐使用的方法。

②

```
strs='abcdefh'
print(strs[0:-1])  # fedcb
```

取不到后(-1)

取得到前

```
def reversed_int(a):
    if a>0:
        return str(a)[::-1]
    else:
        return '-'.join(str(a)[:0:-1])
```

取不到的

③

```
from functools import reduce
print(reduce(lambda x,y:y+x,strs))
```

也可以反转

```
list(reversed(list(strs)))
```

介：join 空字符串拼接。

作业讲解2：

```
def func(a):
    if isinstance(a, list):  #判断类型
        for i in a:
            if isinstance(i, list):
                func(i)
            else:
                newlist.append(i)
```

arr.T 为矩阵转置、np.transpose(arr) 也可以转置

arr.transpose(1, 0) 行→列 列→行（轴向转置）

arr.swapaxes(1,0) #只能传两个参数

比如 arr.transpose(2,1,0) arr.swapaxes(0,2)

np.dot(arr1, arr1.T) #矩阵相乘. #其中T为属性 transpose 为方法.

np.vstack 和 np.hstack：

arr1 → array([[0,1,2], [3,4,5]])

print(np.hstack((arr1, arr1))) 可以是元组、列表, numpy数据类型 → 可以不止两个，而是多个. 而vstack是竖着排

[[0 1 2 0 1 2]
 [3 4 5 3 4 5]]

concatenate 连接: np.concatenate((arr, arr1), axis=1)

2×3 → 2×6   整合使用 拼接

ndarray.itemsize 每个元素的大小，以字节为单位.

多维矩阵相乘，除了最后两维 其他维度要相同，实际上也是最后两个维度相乘.

相加: np.add(arr, arr1).

x=np.array([3,1,2])

y=np.argsort(x)

y → # array([1,2,0], dtype=int64)
(返回索引)

np.argmax(), np.argmin():

np.argmax(a, axis=0) 列

np.argmax(a, axis=1) 行

np.argmin(a, axis=1) 行

X=np.arange(9.).reshape(3,3)

np.where(X<5) # 返回索引,

y=
np.where(X>3)

(array([1,1,2,2,2], dtype=int64), array([1,2,0,1,2], dtype=int64))

X[y] → 获取索引对应值

numpy.nonzero(a) 获得非零元素索引, → a[y]

获取值

np.unique 去重: A=[1,2,2], a=np.unique(A)

→ list(a) → [1,2] # 也可用集合去重

数据分析,机器学习用pandas, 开源。anaconda 自带, 版本要对应, 强处理表格; 混合数据, 而numpy处理统一数值数组数据。

pandas 主要数据结构 Series 和 DataFrame。

series 保存任何类型,

import pandas as pd

obj = pd.Series([4,7,-5,3]) →

int32 存4个字节, int64. 占8个字节(空间)

obj为
```
0   4
1   7
2  -5
3   3
dtype: int64
```

a.astype(np.float32) 转换数据类型.

obj2 = pd.Series([4,7,-5,3]).index(['d','b','a','c']) 指定索引

默认 0, 1, 2, 3

通过索引获取值 obj2['c'] → 3., 索引可以为字母。

mydict = dict(zip(mylist, myarr)) → 在高级编程中

zip使用. ('a', 0) 返回元组

print(ser3.head(10)) 打印前10个数据

对应 tail 为末尾

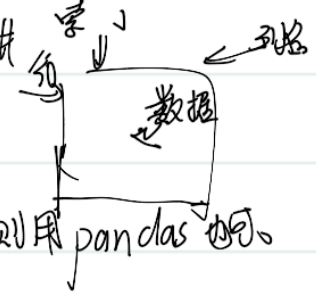
obj.dtype
obj.index 索引
obj4.values

obj.isnull() 一样的 判断非空

也可以通过字典创建 Series, 以及 pd.isnull(obj4), pd.notnull(obj4)

索引是字符串, 可使用负索引。整数型索引, 就不可以用负索引。

frame = pd.DataFrame(data)



frame 共显示出表格, 行和列

数据比较复杂、且有关联, 用数据库, 否则用 pandas 也可。linux, shell, oracle 要会用。

反扒机制基本上就无法爬了。

frame.colums ['year', ...], frame2.year 用这个对数取列, frame2.loc['three'] 对应索引 并取出three行

frame2.reindex(['a','b','c','d']) → 多出的索引行用 NaN 填充, 可使原本索引顺序倒顺序。data = frame2.drop('b') → 排除 b 行 {axis=0 排除行 / axis=1 排除列.} > pandas 和 numpy 不一样的。