##### 实验5 算法设计与实现（1）

###### 1. 实验目的

（1）了解基本算法的设计

（2）掌握用Python语言进行算法实现

###### 2. 实验任务

**实验任务5-1 求解阿姆斯特朗数**

阿姆斯特朗数也叫水仙花数,它的定义如下:若一个n位自然数的各位数字的n次方之和等于它本身,则称这个自然数为阿姆斯特朗数。例如153(153=1\*1\*1+3\*3\*3+5\*5\*5)是一个三位数的阿姆斯特朗数,8208则是一个四位数的阿姆斯特朗数。

现请你编一个程序找出所有的三位数到七位数中的阿姆斯特朗数。

**实验指导：**用穷举法解决该问题

1. 确定变量i的穷举的空间[100,10000000]
2. 获得该变量每一位上的数字（方法有多种）
3. 计算其位数次方
4. 如果是水仙花数，则输出i，否则继续穷举

**实验任务5-2 编写程序实现给定区间二分查找**

**要求：**

（1）接收用户从键盘输入一个N个数的有序整数序列

（2）用户输入一个数，在该有序整数序列中用二分搜索查找该数，若找到，则输出其在整数序列中的位置编号；若未找到，则输出“NO FOUND!”

**实验指导：**

设R[low..high]是当前的查找区间

（1）首先确定该区间的中点位置：R[mid]

（2）然后将待查的K值与R[mid]比较：若相等，则查找成功并返回此位置，否则须确定新的查找区间，继续二分查找，具体方法如下：

①若R[mid] >K，则由表的有序性可知R[mid..n]均大于K，因此若表中存在关键字等于K的结点，则该结点必定是在位置mid左边的子表R[1..mid-1]中，故新的查找区间是左子表R[1..mid-1]。

②类似地，若R[mid] <K，则要查找的K必在mid的右子表R[mid+1..n]中，即新的查找区间是右子表R[mid+1..n]。下一次查找是针对新的查找区间进行的。

下面左图为长度为n的列表进行二分查找的算法流程图（答案不唯一）

#5-2.py

#编写函数接收用户从键盘连续输入一组数据

def getInput():

l=[]

print("Please input all the numbers one by one")

print("Press whitespace to quit:")

while(True):

b=input()

#如果用户输入了一个空格，则停止接收输入

if b==" ":

break

else:

l.append(int(b))

return l

#编写函数在指定的列表l中搜索关键字key

def binarySearch(l,key):

请补充代码

if \_\_\_name\_\_=="\_\_main\_\_":

l=getInput()

key=int(input("Please input the key you want to search:"))

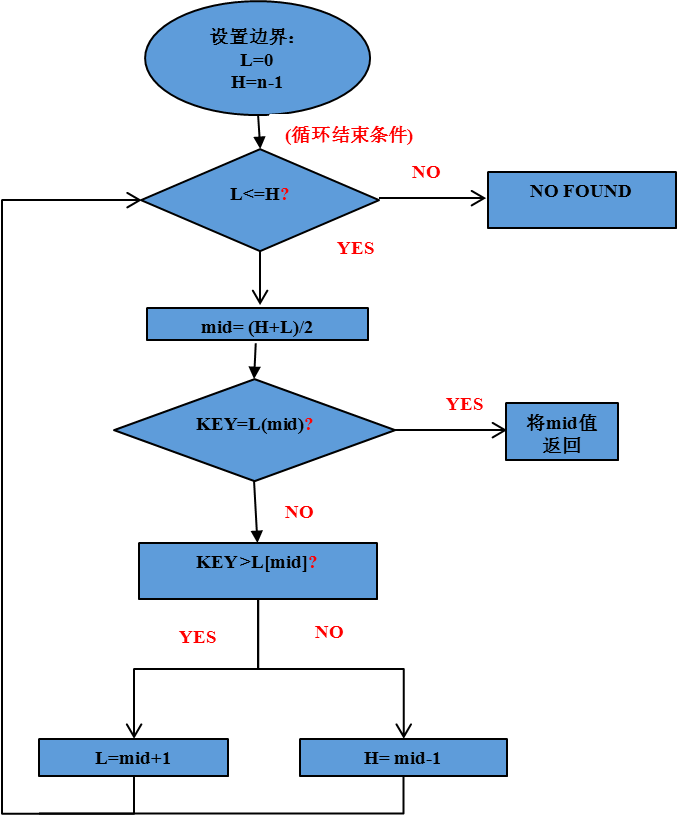
result=binarySearch(l,key)

if result==-1:#若查找失败

print("NO FOUND!")

else:#若查找成功，返回位置索引

print("The index of the key in the list is:",result)



**实验任务5-3 利用枚举法和二分法结合求解一元三次方程（选做，有加分）**

有形如： 这样的一个一元三次方程。给出该方程中各项的系数(a，b，c，d均为实数)，并约定：该方程存在三个不同实根（注意：根的范围在-100至100之间），并且根与根之差的绝对值>=1。

**提示：**

记方程，若存在2个数和，且，，则在之间一定有一个根。

**要求：**由小到大依次输出这三个实根，并精确到小数点后2位。

样例

输入：1 -5 -4 20

输出：-2.00 2.00 5.00

实验指导：

（1）根据提示“二个根之间差的绝对值>＝1”，说明在任何一个子区间[A,A+1）中，要么无根，要么存在唯一的一个根；提示又说明可以在任何一个存在唯一根的子区间[A,A+1）中用二分法查找这个根。

（2）据此，可以得出如下的参考算法：

1）先在区间集[-100,-99)、[-99,-98)…[99,100)、[100,100)中利用提示的条件进行穷举。首先判断该区间的端点是否为根（即满足），若满足，则输出该根。

2）判断该区间是否满足 ，若满足条件，则判断该区间有且必有一个根，在该区间内用二分法查找这个根（缩小查找区间）。

3）整个循环直到“所有区间都穷举完毕”。

（3）在有且仅有一个根区间中查找根的二分法设计思想为每次缩小一半的区间，并通过检查是否满足 判断该子区间是否为解所在的区间：

令low=, high=

While(low<high)

mid=(low+high)/2

if f(low)\*f(mid)<0

mid=high

if f(mid)\*f(high )<0

low=mid

return mid (循环完毕后mid即为方程的解)

**语法提示：**

（1）浮点数的相等不能直接用“==”，而必须用内置函数abs()，例如判断浮点数a是否等于0，应表示为：abs(a) <1e-4（取决于计算的精度要求）

（2）浮点数按照指定的精度输出应表示为print(“%.2f”%a)，该语句表示将浮点数a精确到小数点后2位进行输出。