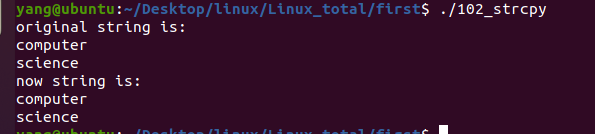
第一部分：

第一题

定义一个指针和统计长度的变量，指针指向数组的头部，从头开始遍历，直到不为’\0’，指针每往后加一位，长度变量+1。（代码：杨子易）



第二题

定义一个空数组和一个指向它的指针p1，定义一个指向被复制字符串的指针p2，遍历（p2的值不为’/0’）被复制的字符串，令p2的值等于p1。（代码：杨子易）  


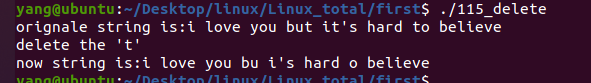
第十一题

定义两个变量，i=0，j=字符串长度，判断s[i]是否等于s[j]，如果不等于，直接输出no并跳出循环，如果等于，i++，j++。直至i与j相遇，则是回文字符串。（代码：杨子易）

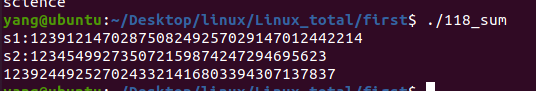


第十五题

定义一个空字符串，遍历原始字符串，如果没有待删除的字母，则直接赋值给空字符串，如果含有待删除的字母，不赋值给空字符串并且控制空字符串的变量不加加。（代码：杨子易）

第十八题

因为需要考虑到位数不齐和进位的问题，所以将字符串s1上存放位数最多的大数，然后在s2的首位insert 0，直至二者位数相等。将s1和s2 reverse，定义一个flag进位符，二者逐位相加并加上flag，如果值大于10，则flag为1并将加得的值减10赋给该位的值，小于则为0，则flag为0并将加得的值直接赋给该位的值，最后再逆序输出值。（代码：杨子易）



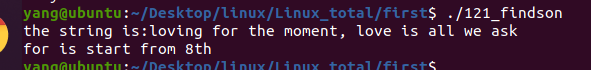
第二十题

因为需要查询每个字符出现的次数，最多只有26个字母，故建立哈希表，定义一个长度为26的数组，所有值全赋值为0，开始遍历该字符串，例如z出现了一次，则将a[25]的值加一，最后遍历该数组，输出值仅为1对应的数组下表。（代码：杨子易）



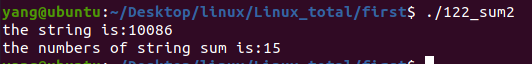
第二十一题

C++使用find函数，返回值>0则是该子串所在的位置。（代码：杨子易）



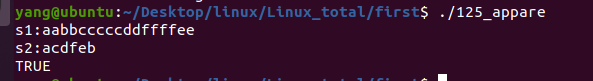
第二十二题

定义sum=0，遍历字符串，因为ASCII码，将遍历到的值-‘0’，输出结果sum即可。（代码：杨子易）



第二十五题

因为要找s1中出现的字母，所以建一个哈希表，定义一个数组a[26]，并将所有值赋0，遍历s1，将s1出现到的字母添加到数组中，最后再遍历s2和表中所有的字母进行对比，如果全符合，则true。（代码：杨子易）



第二十六题

因为要以空格为单位逆序输出，则先定义一个string数组，遍历初始数组的同时，以空格为分隔符，将每个单词加入到每个数组之中，最后逆序输出该数组即可。（代码：杨子易）



第二十九题

找最长最长回文子串，使用“马拉车算法”。

考虑到字符串可能是奇数或者偶数，所以在每个字符和首位都插入一个特殊，这样让字符串固定为奇数，就有对称中心了。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | A | # | B | # | B | # | A | # |
| 0 | 3 | 0 | 3 | 9 | 3 | 0 | 3 | 0 |

以对称中心开始，左边和右边的对称值完全相等，围绕中心对称。

但是如果对称中心不在中间，则·可能会超出边界。

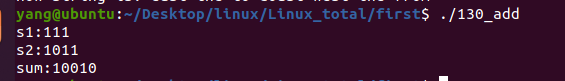
所以，如果第i个数小于右边减i，那么他们就相互对称，如果大于，则需要穿过中心对称点逐个字符判定了。

故，如果在i处的回文数超过了右边界，让i成为中间点，更新边界r。（代码：杨子易）



第三十题

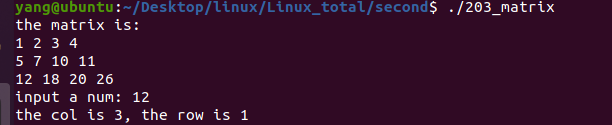
定义三个vector<int>，两个用来将string字符串转换为int类型，并比较这两个大小，如果a1比a2小，swap二者，将两个数从begin到end逆序并补零对其位数，定义flag=0，将a1[i]加a2[i]加flag的值进行判断，如果大于二，flag=1，如果小于2，将值赋给a3[i]。（代码：杨子易）



第二部分：

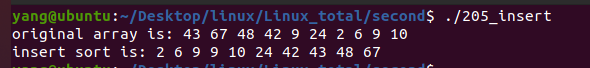
第三题

因为是有序数组，所以比较每一行的最后一列即可知道该数字在哪一行，再根据找到的那一行进行遍历，即可找到该矩阵的位置。（代码：杨子易）



第五题

直接插入排序，算法复杂度n^2，稳定排序。直接插入升序排序分为两部分，左边部分是已经从小到大排好的，右边是待排，从右边选一个数，依次与左边的最大数到最小数比较大小，插入比他小的数后面。（代码：杨子易）



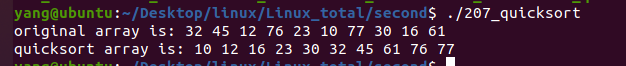
第六题

希尔排序，算法复杂度小于n^2，核心是分组问题，每一次分组长度为总长度的1/2，组内进行排序，排完之后再一次分组，再一次排序，直至d=1的时候是最后一次排序。（代码：杨子易）



第七题

快速排序，算法复杂度nlongn，找一个作为pivot，左边是所有小于pivot的数，右边全是大于pivot的数。左边的L下表和右边的R下标依次与pivot比较，如果L下标对应的值小于pivot则不变并且L下标+1，同理右边也是一样，当一次排序完毕，分别对其左子序列和右子序列进行之前的操作。（代码：杨子易）



第三部分

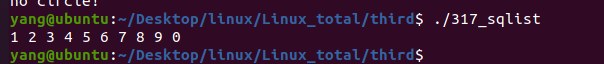
第十五题

判断单链表是否有环。假设存在环路，在一个环路内，速度不相等的两个结点必定会相遇，所以定义一个slow结点和一个fast结点，fast结点指向下一个的下一个，slow指向下一个结点，slow每次+1，fast每次+2，当二者相遇则有环，当fast为null或者fast的next为null之时，没有环。（代码：杨子易）



第十七题

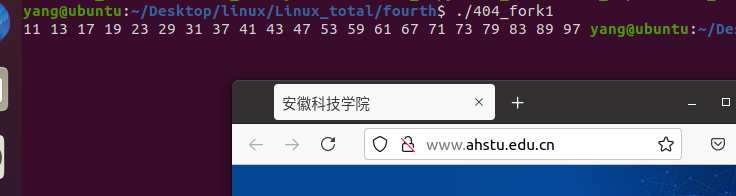
建一个新线性表L3，将线性表LA和线性表LB的data[i]对比，取得二者相比较小的的data[i]放入L3之中。（代码：杨子易）



第四部分

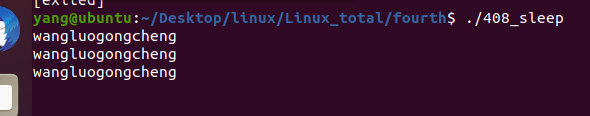
第四题

先pid=fork()，pid等于0，system调用firefox，大于0，求素数（i模上2到根号m取模都不为0）。（代码：杨子易）



第八题

不用sleep函数实现睡眠，使用到了信号，在mysleep函数中定义sigaction的act和oldact，先sigemptyset初始化屏蔽字，再设置对时钟信号的捕捉，最后再返回旧的信号屏蔽字。（代码：杨子易）



第九题

定义两个函数，一个是处理信号2，一个是处理信号3，主函数通过signal来将接受到的信号转发给处理函数，例如接收到信号2，count次数++，倘若count次数等于7，kill这个进程，接收到信号3同理。（代码：杨子易）

