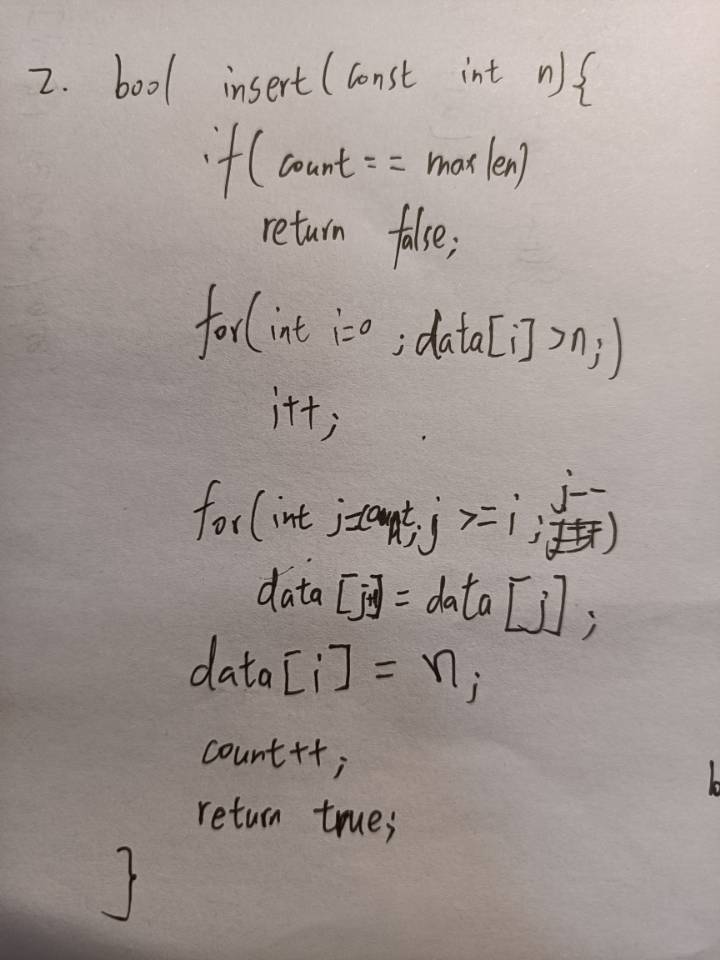
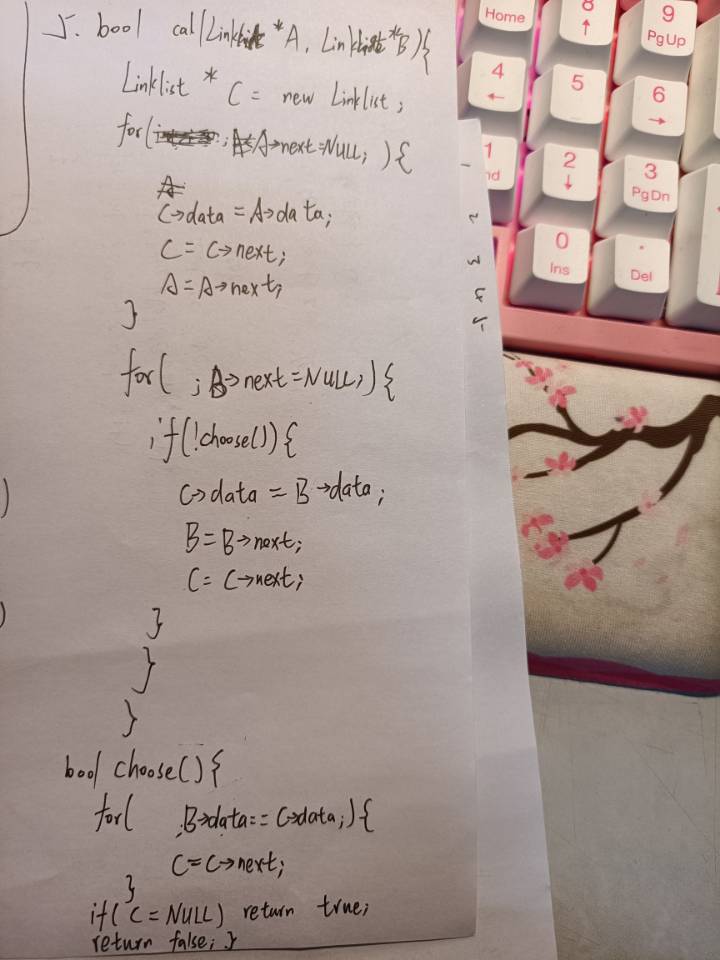
2. 假设顺序表L中的元素递增有序，设计算法在顺序表中插入元素x，要求插入后仍保持其递增有序特性，并要求时间尽可能少。



5. 假设链表A、B分别表示两个集合，设计算法以求解C= A∪B，并分析算法的时间复杂度。



9.设计算法以比较链串*S1*和链串*S2*的大小，若*S1* < *S2*，返回-1；若*S1* ＝ *S2*，返回0；否则返回1。

#include "iostream"  
#include "string"  
using namespace std;  
  
int ssss(string s1, string s2){  
 int sa = s1.length();  
 int sb = s2.length();  
 if(sa > sb){  
 for(int i = 0; i < sb; i++) {  
 if (s1[i] > s2[i]) {  
 return 1;  
 } else if (s1[i] < s2[i]) {  
 return -1;  
 }  
 if(s1[i] == s2[i] && i == sb - 1){  
 return 1;  
 }  
 if(s1[i] == s2[i] && i != sb - 1){  
 continue;  
 }  
 }  
 } else if(sb > sa){  
 for(int i = 0; i < sa; i++) {  
 if (s1[i] > s2[i]) {  
 return 1;  
 } else if (s1[i] < s2[i]) {  
 return -1;  
 }  
 if (s1[i] == s2[i] && i == sa - 1) {  
 return 1;  
 }  
 if (s1[i] == s2[i] && i != sa - 1) {  
 continue;  
 }  
 }  
 } else if(sa == sb){  
 for(int i = 0; i < sa; i++) {  
 if (s1[i] > s2[i]) {  
 return 1;  
 } else if (s1[i] < s2[i]) {  
 return -1;  
 }  
 if (s1[i] == s2[i] && i == sa - 1) {  
 return 1;  
 }  
 if (s1[i] == s2[i] && i != sa - 1) {  
 return 0;  
 }  
 }  
 }  
}  
int main(){  
 cout << "请输入两字符串" << endl;  
 string s1,s2;  
 cin >> s1 >> s2;  
 cout << ssss(s1,s2) << endl;  
 return 0;  
}