【2014华为机试题】

|  |  |
| --- | --- |
|  | 输入一串数字，找到其中包含的最大递增数。递增数是指相邻的数位从小到大排列的数字。如： 2895345323，递增数有：289，345，23, 那么最大的递增数为345。 |

|  |
| --- |
|          String str = br.readLine();             char[] charArr = str.toCharArray();             //用于存放最大的递增数           int maxNum = 0;             //str用做读入数据后边没用其他用处，就可以拿来放临时递增数           str = "";             for (int i=0; i<charArr.length; i++){                 //将当前i指向的字符存到字符串中               if (str.length() ==0){                   str += charArr[i];               }                 //题目巧合，恰好是比较一位的字符串，若是两位之上遍不可以直接比较字符               //如果第二个大于第一个，将第二个也存放到str中               //注意if的第一个条件 \*\*\*i < charArr.length-1\*\*\*               if (i < charArr.length-1 && charArr[i+1] > charArr[i]){                   str += charArr[i+1];               }                 //若不满足，表明当前递增数结束               //对该数进行比较，存储               else {                   int temp = Integer.parseInt(str);                   if(temp > maxNum){                       maxNum = temp;                   }                     str = "";               }           } |

题目一：子串分离   
题目描述：     
通过键盘输入任意一个字符串序列，字符串可能包含多个子串，子串以空格分隔。请编写一个程序，自动分离出各个子串，并使用’,’将其分隔，并且在最后也补充一个’,’并将子串存储。   
如果输入“abc def gh i        d”，结果将是abc,def,gh,i,d,

1. void DivideString(const char\* pInputStr , long lInputLen , char\* pOutputStr)
2. {
3. int i , j;
4. bool flag;
5. for(i = 0 ; pInputStr[i] == ' ' ; ++i)   //跳过字符串前面的空格
6. ;
7. flag = true;
8. for(j = 0 ; i < lInputLen ; ++i)
9. {
10. if(pInputStr[i] != ' ')
11. {
12. if(!flag)
13. flag = true;
14. pOutputStr[j++] = pInputStr[i];    //将各个子串分离保存下来
15. }
16. else
17. {
18. if(flag)
19. pOutputStr[j++] = ',';
20. flag = false;
21. }
22. }
23. pOutputStr[j++] = ',';
24. pOutputStr[j++] = '\0';
25. }

题目二：逆序链表输出。   
题目描述：     
将输入的一个单向链表，逆序后输出链表中的值。链表定义如下：   
typedef struct tagListNode   
{   
      int value;   
      struct tagListNode \*next;   
}ListNode;

// 返回新链表头节点

LinkNode \*reverse\_link(LinkNode \*head)

{    if(head == NULL)

     return NULL;

    LinkNode \*prev , \*curr , \*reverse\_head , \*temp;

    prev = NULL , curr = head;

    while(curr->next)

    {

        temp = curr->next;

        curr->next = prev;

        prev = curr;

        curr = temp;

    }

curr->next = prev;

    reverse\_head = curr;

    return reverse\_head;

}

**2012届华为校园招聘机试题**

1、选秀节目打分，分为专家评委和大众评委，score[] 数组里面存储每个评委打的分数，judge\_type[] 里存储与 score[] 数组对应的评委类别，judge\_type[i] == 1，表示专家评委，judge\_type[i] == 2，表示大众评委，n表示评委总数。打分规则如下：专家评委和大众评委的分数先分别取一个平均分（平均分取整），然后，总分 = 专家评委平均分  \* 0.6 + 大众评委 \* 0.4，总分取整。如果没有大众评委，则 总分 = 专家评委平均分，总分取整。函数最终返回选手得分。

1. int cal\_score(int score[], int judge\_type[], int n)
2. {
3. int i , TotalExpertScore , TotalPublictScore , ExpertNum , PublicNum;
4. double res = 0;
5. TotalExpertScore = TotalPublictScore = 0;
6. ExpertNum = PublicNum = 0;
7. for(i = 0 ; i < n ; ++i)
8. {
9. if(1 == judge\_type[i])      //专家评委
10. {
11. TotalExpertScore += score[i];
12. ++ExpertNum;
13. }
14. else                        //大众评委
15. {
16. TotalPublictScore += score[i];
17. ++PublicNum;
18. }
19. }
20. if(0 == PublicNum)              //没有大众评委
21. {
22. return TotalExpertScore/ExpertNum;
23. }
24. else
25. {
26. res = 0.6\*TotalExpertScore/ExpertNum + 0.4\*TotalPublictScore/PublicNum;
27. return (int)res;
28. }
29. }

2、给定一个数组input[] ，如果数组长度n为奇数，则将数组中最大的元素放到 output[] 数组最中间的位置，如果数组长度n为偶数，则将数组中最大的元素放到 output[] 数组中间两个位置偏右的那个位置上，然后再按从大到小的顺序，依次在第一个位置的两边，按照一左一右的顺序，依次存放剩下的数。

1. void sort(int input[] , int n , int output[])
2. {
3. int i , j , k , temp , mid;
4. for(i = 0 ; i < n - 1 ; ++i)       //冒泡排序（降序）
5. {
6. for(j = 0 ; j < n - i - 1 ; ++j)
7. {
8. if(input[j] < input[j+1])
9. {
10. temp = input[j];
11. input[j] = input[j+1];
12. input[j+1] = temp;
13. }
14. }
15. }//for
16. mid = n>>1;
17. j = mid - 1; //左边
18. k = mid + 1;
19. output[mid] = input[0];
20. for(i = 1 ; i < n ; )               //按照一左一右的顺序
21. {
22. if(j >= 0)
23. output[j--] = input[i++];
24. if(k < n)
25. output[k++] = input[i++];
26. }
27. }

3、操作系统任务调度问题。操作系统任务分为系统任务和用户任务两种。其中，系统任务的优先级 < 50，用户任务的优先级 >= 50且 <= 255。优先级大于255的为非法任务，应予以剔除。现有一任务队列task[]，长度为n，task中的元素值表示任务的优先级，数值越小，优先级越高。函数scheduler实现如下功能，将task[] 中的任务按照系统任务、用户任务依次存放到 system\_task[] 数组和 user\_task[] 数组中（数组中元素的值是任务在task[] 数组中的下标），并且优先级高的任务排在前面，优先级相同的任务按照入队顺序排列（即先入队的任务排在前面），数组元素为-1表示结束。  
      例如：task[] = {0, 30, 155, 1, 80, 300, 170, 40, 99}    system\_task[] = {0, 3, 1, 7, -1}    user\_task[] = {4, 8, 2, 6, -1}

1. void scheduler(int task[] , int n , int system\_task[] , int user\_task[])
2. {
3. int i , j , k , index\_user , index\_system , temp;
4. j = k = 0;
5. index\_user = index\_system = 0;
6. for(i = 0 ; i < n ; ++i)
7. {
8. if(task[i] > 255)
9. {
10. printf("This is illegal task!\n");
11. }
12. else if(task[i] >= 50 && task[i] <= 255)      //用户任务
13. {
14. user\_task[index\_user++] = i;              //存储用户任务的下标
15. }
16. else         //系统任务
17. {
18. system\_task[index\_system++] = i;          //存储系统任务的下标
19. }
20. }//for
21. //对每个类型组中的任务进行优先级排序
22. for(i = 0 ; i < index\_user - 1 ; ++i)        //冒泡排序
23. {
24. for(j = 0 ; j < index\_user - i - 1 ; ++j)
25. {
26. if( task[user\_task[j]] > task[user\_task[j+1]] )
27. {
28. temp = user\_task[j];
29. user\_task[j] = user\_task[j+1];
30. user\_task[j+1] = temp;
31. }
32. }
33. }//for
34. if(index\_user < n)
35. user\_task[index\_user] = -1;
37. for(i = 0 ; i < index\_system - 1 ; ++i)        //冒泡排序
38. {
39. for(j = 0 ; j < index\_system - i - 1 ; ++j)
40. {
41. if( task[system\_task[j]] > task[system\_task[j+1]] )
42. {
43. temp = system\_task[j];
44. system\_task[j] = system\_task[j+1];
45. system\_task[j+1] = temp;
46. }
47. }
48. }//for
49. if(index\_system < n)
50. system\_task[index\_system] = -1;
51. }