**2016编程题**

1. 建立一个结构体包含学生信息（学号，成绩），使用结构体数组和结构体指针，输入200个学生信息，以成绩从低到高排序，输出最高成绩的学生的学号和成绩（最高成绩可能有多名学生）。
2. **输入20个数（整型或浮点型），逆序构建一个单向链表。**

#include<stdio.h>

#include<stdlib.h>

typedef struct Node{

float num;

struct Node \*next;

}Node;

int main()

{

freopen("b.txt","r",stdin);

Node \*s,\*head;

int i;

float t;

head = (Node \*)malloc(sizeof(Node));

head->next = NULL;

for(i = 0;i < 20;i ++)

{

scanf("%f",&t);

s = (Node \*)malloc(sizeof(Node));

s->num = t;

s->next = head->next; //头插法

head->next = s;

}

s = head->next;

while(s)

{

printf("%f\n",s->num);

s = s->next;

}

return 0;

}

**2015年编程题**

**编程一：编写一个函数，从字符串中寻找整个连续的数字字符，将其转化成整数并保存在arr整型数组里，溢出以-1作为标志。字符串以‘#’结束（如"uestc2015jsj123#"）**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int a[50];

int i = 0;

int str\_to\_int(char str[],int len)

{

int num = 0;

for(int j = 0; j < len; j++)

num = num\*10 + (str[j] - '0');

return num;

}

void int\_find(int a[],char \*t)

{

char \*p = t,\*q;

while(\*p != '#')

{

if(\*p >= '0' && \*p <= '9')

{

q = p + 1;

while(\*q >= '0' && \*q <= '9') q++;

a[i++] = str\_to\_int(p,q-p);

p = q; //取当前指针为新的首字符

}

else p++;

}

a[i] = -1;

}

int main()

{

char src[] = "uestc2015jsj123#";

int\_find(a,src);

int t = i;

for(int x = 0; x < t; x++)

printf("%d ",a[x]);

return 0;

}

**编程二：随机输入若干整数和若干浮点数（顺序也是随机的，最多100个），要求编写完整的程序，将整数按从大到小排列，浮点数按从小到大排列（浮点数排序可省略），并输出。如：输入10 12.3 12 5 52.1 3.65 88.6 输出：12 10 5 3.65 12.3 52.1 88.6**

#include<stdio.h>

#include<stdlib.h>

void sort\_int (int num[], int n)

{

for(int i = 0; i < n - 1; i++)

{

for(int j = i + 1; j <= n - 1; j++)

{

if(num[j] < num[i])

{

int temp = num[i];

num[i] = num[j];

num[j] = temp;

}

}

}

}

int main()

{

float temp;

int num\_int[100], count\_int=0, count\_foat=0;

float num\_float[100];

int n;

scanf("%d", &n);

for(int i = 0; i < n; i++)

{

scanf("%f", &temp);

if(temp == (int)temp)

num\_int[count\_int++] = (int)temp;

else

num\_float[count\_foat++] = temp;

}

sort\_int(num\_int, count\_int);

sort\_float(num\_float, count\_foat);

for(int i = 0; i < count\_int; i++)

printf("%d ", num\_int[i]);

for(int i = 0; i < count\_foat; i++)

printf("%f", num\_float[i]);

return 0;

}

**编程三：编写完整的程序，构造整数集合(其实就是一个整数链表)，并实现对该集合操作的若干功能：加入一个新元素，判断某元素是否在集合内，输出两个集合并集，输出两个集合交集，删除集合中某一元素。struct set{int numb; struct set \*next;}**

#include<stdio.h>

#include<stdlib.h>

struct set{

int numb;

struct set \*next;

};

//往集合中添加一个元素--头插法

void add(struct set \*L,int x)

{

struct set \*temp = (set \*)malloc(sizeof(set));

temp->numb = x;

temp->next = L->next;

L->next = temp;

}

//输出集合

void print(struct set \*L)

{

struct set \*temp = L->next;

while(temp)

{

printf("%d ",temp->numb);

temp = temp->next;

}

printf("\n");

}

//查找集合中的某一个元素

int isExitList(struct set \*L,int x)

{

struct set \*temp = L->next;

while(temp)

{

if(temp->numb == x) return 1;

temp = temp->next;

}

return 0;

}

//删除集合的某一个元素

void deleteList(struct set \*L,int x)

{

struct set \*temp = L->next;

struct set \*pre = L;

while(temp)

{

if(temp->numb == x) //集合元素是互异的，找到即结束循环

{

pre->next = temp->next;

delete(temp);

break;

}

else

{

pre = temp;

temp = temp->next;

}

}

}

//输出两个集合的并集

void interSectin(struct set \*L1,struct set \*L2)

{

struct set \*p1 = L1->next;

struct set \*p2 = L2->next;

int flag;

while(p1)

{

printf("%d ",p1->numb);

p1 = p1->next;

}

while(p2)

{

p1 = L1->next; //每次都得初始化的数据

flag = 0;

while(p1)

{

if(p1->numb == p2->numb)

{

flag = 1;

break;

}

p1 = p1->next;

}

if(flag == 0) printf("%d ",p2->numb);

p2 = p2->next;

}

}

//输出两个集合的交集

void Union(struct set \*L1,struct set \*L2)

{

struct set \*p1 = L1->next;

struct set \*p2 = L2->next;

int flag;

while(p1)

{

p2 = L2->next;

flag = 0;

while(p2)

{

if(p1->numb == p2->numb){

printf("%d ",p1->numb);

break; //找到即可结束循环，因为集合元素是互异的

}

p2 = p2->next;

}

p1 = p1->next;

}

}

int main()

{

struct set \*L1 = (set \*)malloc(sizeof(set));

struct set \*L2 = (set \*)malloc(sizeof(set));

L1->next = NULL;

L2->next = NULL;

int temp;

scanf("%d",&temp);

while(temp != -1)

{

add(L1,temp);

scanf("%d",&temp);

}

scanf("%d",&temp);

while(temp != -1)

{

add(L2,temp);

scanf("%d",&temp);

}

printf("输出交集：");

Union(L1,L2);

printf("\n");

printf("输出并集：");

interSectin(L1,L2);

return 0;

}

**2014年编程题**

1. **编写函数，实现字符串的逆序，不允许申请新的数组空间**

#include<stdio.h>

#include<stdlib.h>

int main()

{

char str[50]= "UESTC2017/3/12";

int len = 0;

char ch;

while(str[len] != '\0') len++;

for(int i = 0; i <= len / 2; i++)

{

ch = str[i];

str[i] = str[len - i - 1];

str[len - i - 1] = ch;

}

for(int i = 0; i < len; i++)

printf("%c",str[i]);

return 0;

}

1. **身份证长度是18，其中第7至14个数字是生日，编写完整程序判断2个身份证号的出生日期先后。**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int is\_Same(char st1[],char st2[])

{

int low = 6, high = 13;

while(low <= high && st1[low] == st2[low]) low++;

return st1[low] - st2[low];

}

int main()

{

char st1[19],st2[19];

gets(st1);

gets(st2);

int t = is\_Same(st1,st2);

if(t > 0) printf("第二个人生日先\n");

else if(t < 0) printf("第一个人生日先\n");

else printf("生日相同\n");

return 0;

}

1. **编写完整程序，计算1-x+x^2/2!-x^3/3!+…+x^n/n!，只允许有Main函数，不允许定义其他函数，x是单浮点数，n是整数。**

#include<stdio.h>

#include<stdlib.h>

int main()

{

double item = 1, sum = 1;

int n;

scanf("%d",&n);

int x = 1;

for(int i = 1; i <= n; i++)

{

item \*= -1\*1.0\*x/i;

sum += item;

}

printf("%f\n",sum) ;

return 0;

}

1. **编写完整程序，一个链表，找出其中数据项最大的结点，然后将其移动到链表尾部（结点node由整型data和节点指针next构成），不允许申请新的结点。**

#include<stdio.h>

#include<stdlib.h>

struct node{

int data;

struct node \*next;

};

void print(struct node \*L)

{

struct node \*temp = L->next;

while(temp)

{

printf("%d ",temp->data);

temp = temp->next;

}

printf("\n");

}

//头插法建立单链表

void init\_List(struct node \*L,int n)

{

int x;

for(int i = 1; i <= n; i++)

{

scanf("%d",&x);

struct node \*temp = (node \*)malloc(sizeof(node));

temp->data = x;

temp->next = L->next;

L->next = temp;

}

}

void max\_Node(struct node \*L)

{

struct node \*temp = L->next;

struct node \*pre = L;

struct node \*max = L->next;

while(temp)

{

if(temp->data > max->data)

{

max = temp;

}

pre = temp;

temp = temp->next;

}

//循环结束temp肯定为空，则pre指向链表的最后一个结点

int tempnum = pre->data;

pre->data = max->data;

max->data = tempnum;

}

int main()

{

int n;

struct node \*L = (node \*)malloc(sizeof(node));

L->next = NULL;

scanf("%d",&n);

init\_List(L,n);

print(L);

max\_Node(L);

print(L);

return 0;

}

**2013年编程题**

**1.把整数分解成素数 如90=2\*3\*3\*5**

#include<stdio.h>

int main()

{

int i,x;

while(scanf("%d",&x))

{

for(i = 2; i <= x; i++) //找<=x的质数

{

while(x != i) //若x=i,则输入的x是质数

{

if(x % i == 0) //若x可以被质数i整除

{

printf("%d ",i); //则质数i是其中一个质因数

x = x / i; //x=x/i 用整除后的x求解下一个质因数,若x=i(质数)则退出打印x,若不等且不能整除i,则i+1

}

else

break;

}

}

printf("%d \n",x);

}

return 0;

}

**2012年编程题**

1. **编写完整程序：利用2个函数对输入的两个分数进行加、减、乘、除四则运算和输出用分数表示的结果。(注：输入格式为：%ld/%ld%c%ld/%ld，输出格式为%ld/%ld)，例如：输入1/4+1/3，输出：7/12**

#include<stdio.h>

#include<stdlib.h>

long fz1,fm1,fz2,fm2,fz3,fm3;

char op;

long gcd(long m, long n)

{

int r;

while(n)

{

r = m % n;

m = n;

n = r;

}

return m;

}

void compute()

{

long gxz;

switch(op)

{

case '+':

case '-':

gxz = fm1\*fm2/gcd(fm1,fm2);

fz1 = gxz/fm1\*fz1;

fz2 = gxz/fm2\*fz2;

if(op == '+') fz3 = fz1 + fz2;

else fz3 = fz1 - fz2;

fm3 = gxz;

break;

case '\*':

fz3 = fz1 \* fz2;

fm3 = fm1 \* fm2;

break;

case '/':

fz3 = fz1 \* fm2;

fm3 = fm1 \* fz2;

break;

default:

exit(-1);

}

gxz = gcd(fz3,fm3);

fz3 /= gxz;

fm3 /= gxz;

printf("%ld/%ld\n",fz3,fm3);

}

int main()

{

while(scanf("%ld/%ld%c%ld/%ld",&fz1,&fm1,&op,&fz2,&fm2) != EOF)

{

compute();

}

return 0;

}

**2、编写函数，将单链表进行逆序，即表头变表尾，表尾变表头其中：**

**节点定义为：struct node{int num, struct node\*next};函数原型为：void turn(struct node\*head);**

#include<stdio.h>

#include<stdlib.h>

struct node{

int num;

struct node \*next;

};

void Init\_List(struct node \*head, int n)

{

int x;

struct node \*L = head;

//尾插法建立单链表

for(int i = 0; i < n; i++)

{

struct node \*temp = (node \*)malloc(sizeof(node));

scanf("%d",&x);

temp->num = x;

L->next = temp;

L = temp;

}

L->next = NULL;

}

//带头结点的链表逆序

void turn(struct node \*head)

{

struct node \*old\_head = head->next;

struct node \*new\_head = head->next;

head->next = NULL; //断开头结点

while(old\_head)

{

new\_head = old\_head->next; //获取old\_head的下一个节点的地址

old\_head->next = head->next; //将old\_head节点插入到头结点L后面，old\_head的后记节点指向head原有的节点

head->next = old\_head; //old\_head为头结点的第一个元素，头结点的后记节点指向old\_head

old\_head = new\_head; //old\_head为指向下一个需要交换的节点

}

}

void print(struct node \*head)

{

struct node \*temp = head->next;

while(temp)

{

printf("%d ",temp->num);

temp = temp->next;

}

printf("\n");

}

int main()

{

struct node \*head = (node \*)malloc(sizeof(node));

head->next = NULL;

int n;

scanf("%d",&n);

Init\_List(head, n);

print(head);

turn(head);

print(head);

return 0;

}

1. **编写完整程序：接收从键盘输入的仅由数字字符构成的字符串(假设字符串的最大长度为50)，统计并输出每个数字(0-9)的重复次数。**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int num[10];

int main()

{

char ch;

while((ch = getchar()) != '\n')

{

num[ch - '0'] ++;

}

for(int i = 0; i < 10; i++)

{

printf("%d %d\n",i,num[i]);

}

return 0;

}

1. **编写完整程序，采用结构数组和指向结构的指针，接收输入的100个学生信息(包括学号和C语言课程期末总成绩)，输出最高、最低成绩和分别对应的学号(可能有多个同学都是最高分，可能有多个同学都是最低分)。**

#include<stdio.h>

#include<stdlib.h>

const int maxn = 100;

struct Stu{

int num;

int grade;

};

int main()

{

freopen("a.txt","r",stdin);

Stu stu[maxn],\*p;

p = stu;

int min\_grade,max\_grade;

for(int i = 0; i < maxn; ++i)

{

scanf("%d%d",&p->num,&p->grade);

p++;

}

for(int i = 0; i < maxn; i++)

{

for(int j = i+1; j < maxn; j++)

{

if(stu[i].grade > stu[j].grade)

{

Stu temp = stu[i];

stu[i] = stu[j];

stu[j] = temp;

}

}

}

for(int i = 0; i < maxn; i++)

printf("%d %d\n",stu[i].num,stu[i].grade);

min\_grade = stu[0].grade;

max\_grade = stu[maxn - 1].grade;

printf("最低成绩为:%d\n",min\_grade);

printf("学号为：") ;

**p = stu;**

while(p->grade == min\_grade)

{

printf("%d\n",p->num);

**p++;**

}

printf("最高成绩为:%d\n",max\_grade);

printf("学号为：") ;

**p = (stu + maxn - 1);**

while(p->grade == max\_grade)

{

printf("%d ",p->num);

**p--;**

}

return 0;

}

**2015年保研编程题**

1. **编写一个完整的程序，使之能完成以下功能：从键盘中输入若干个整数，用链表储存这些输入的数，并要求存储的顺序与输入的顺序相反。**

**2.编写一个函数，把整数序列分成两个部分，使得左边部分都不大于右边部分，不需要排序。**

#include<stdio.h> //考查的是快排部分

#include<stdlib.h>

int partion(int arr[],int n)

{

int pos = 0;

int i = 0, j = n - 1;

int temp;

while(i != j)

{

while(arr[j] >= arr[pos] && i < j ) j--;

while(arr[i] <= arr[pos] && i < j) i++;

if(i < j)

{

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

temp = arr[i];

arr[i] = arr[pos];

arr[pos] = temp;

}

int main()

{

int a[10]={6,4,0,2,8,7,3,9,1,5};

partion(a,10);

for(int i = 0; i < 10; i++)

printf("%d ",a[i]);

return 0;

}

3、有两个整数数组A和B，它们分别有m、n个整数。并且都是按非递减序列，现将B数组插入A数组中，使得A数组中各元素不大于B数组中各元素，且还是非递减序列。

1. 两个递增有序整数数列链表La和Lb，将他们合并后，变成一个新的链表，要求该链表递减排序。（结点node由整型data和节点指针next构成）

#include<stdio.h>

#include<stdlib.h>

typedef struct node

{

int data;

struct node \*next;

}node;

//必须采用尾插法建立单链表

node \*init\_List(node \*head,int n)

{

int x;

node \*s = head;

for(int i = 1; i <= n; i++)

{

node \*temp = (node \*)malloc(sizeof(node));

scanf("%d",&x);

temp->data = x;

s->next = temp;

s = temp;

}

s->next = NULL;

return head;

}

node \*Union(node \*La, node \*Lb)

{

node \*p, \*q, \*s;

p = La->next;

q = Lb->next;

La->next = NULL;

free(Lb);

while(p != NULL && q != NULL){

if(p->data <= q->data) {

s = p->next;

p->next = La->next; /\* there is some problem \*/

La->next = p;

p = s;

}

else {

s = q->next;

q->next = La->next;

La->next = q;

q = s;

}

}

while(p != NULL) {

s = p->next;

p->next = La->next;

La->next = p;

p = s;

}

while(q != NULL) {

s = q->next;

q->next = La->next;

La->next = q;

q = s;

}

return La;

}

void print(node \*head)

{

node \*tmp = head->next;

while(tmp)

{

printf("%d ",tmp->data);

tmp = tmp->next;

}

printf("\n");

}

int main()

{

node \*L1,\*L2;

int n,m;

L1 = (node \*)malloc(sizeof(node));

L1->next = NULL;

printf("输入链表1的长度:");

scanf("%d",&m);

L1 = init\_List(L1,m);

L2 = (node \*)malloc(sizeof(node));

L2->next = NULL;

printf("输入链表2的长度:");

scanf("%d",&n);

L2 = init\_List(L2,n);

node \*new\_head = Union(L1,L2);

print(new\_head);

return 0;

}

1. 编写一个函数，删除链表中的最小值。（结点node由整型data和节点指针next构成）
2. 编写函数判断小括号是否匹配。(注这道题由于限定了只有小括号，故不需要栈来实现)
3. 对多个字符串进行字典排序
4. 编写一个函数，使之能完成以下功能：利用递归方法找出一个数组中的最大值和最小值，要求递归调用函数的格式如下：MinMaxValue(arr,n,&max,&min)，其中arr是给定的数组，n是数组的个数，max、min分别是最大值和最小值。
5. 有两字符数组s和t，求t在s中出现第一次的开始位置，如果没有则输出“No”，有则输出开始位置。

约瑟夫环，100个人围城圈，从第一个人开始，1~3报数，数到3退出，问剩下来的人编号是多少？

#include<stdio.h>

#include<stdlib.h>

int main()

{

int a[100];

int i,n,index;

for(i = 0; i < 100; i++)

a[i] = i + 1;

n = 100;

i = -1;

index = 0;

while(n > 1)

{

i++;

if(i == 100) i = -1;

if(a[i] == 0)

continue;

index++;

if(index % 3 == 0)

{

printf("第%d号离开\n",a[i]);

n--;

a[i] = 0;

}

}

for(i = 0; i < 100; i++)

if(a[i] != 0)

printf("最后为%d号",a[i]);

return 0;

}