程设板子

目录

30、最大连续子序列和

qsort

```
qsort(a,n,sizeof(a[0]),cmp); //数组a，大小为n

int cmp(const void *a ,const void *b)
{
    return *(int *)a > *(int *)b;//将数组升序排序
    //return *(int *)b < *(int *)a; 将数组降序排序
}

//对结构体进行快排
struct S{
    int cost,fee;
}stu[50010],ans[25];
//二级排序，先将fee降序排序，再将cost升序排序
int comp(const void *p1,const void *p2) {
    if( (*(struct S *)p2).fee != (*(struct S *)p1).fee)
        return (*(struct S *)p2).fee>(*(struct S*)p1).fee?1:-1;
    return (*(struct S *)p2).cost>(*(struct S*)p1).cost?-1:1;
}
qsort(stu,n,sizeof(stu[0]),comp);
//结构体快排可以用两次冒泡代替
```

二进制转十进制

```
#include<stdio.h>
#include<math.h>
#include<string.h>
char s[100];
int main() {
    int n,i,len;
    long long ans=0;
    scanf("%s",s);
    len=strlen(s);
    for(i=0;i<len;i++)
        ans+=(long long)(s[i]-'0')*pow(2,len-i-1);
    printf("%lld",ans);
    return 0;
}
```

十进制转二进制

```
#include <stdio.h>
int main(){
    int x; //x为十进制数
    scanf("%d",&x);
    int a[100];
    int count=0;
    do{
        a[count++]=x%2;
        x=x/2;
    }while(x!=0);//当商不为0时进行循环

    for(int i=count-1;i>=0;i--){
```

```
        printf("%d",a[i]);
    }
    return 0;
}
```

最大公约数

```
//多组数据输入，求数字ab的最大公约数
#include<stdio.h>

int main() {
    int a,b;
    while(scanf("%d%d",&a,&b) != EOF) {
        if(a < b) {
            int c;
            c = a;
            a = b;
            b = c;
        }//保证a>b
        while(a % b != 0) {
            int c = a%b;
            a = b;
            b = c;
        }
        printf("%d\n",b);
    }
    return 0;
}
```

向上/下取整

```
//输入t，a，b，求a/b。当t=1，向下取整。当t=2，向上取整。
#include<stdio.h>
#include<math.h>
int main() {
    int t;
    double a,b;
    scanf("%d %lf %lf",&t,&a,&b);
    if(t==1)
        printf("%d",(int)floor(a/b));
    else
        printf("%d",(int)ceil(a/b));
    return  0;
}
```

复数计算器

```
#include<stdio.h>
#include<math.h>

void print(double a,double b) {
    if(b >= 0) {
        printf("%.2lf+%.2lfi\n",a,b);
    }
    else{
```

```
        printf("%.2lf%.2lfi\n",a,b);
    }
    return;
}
int main() {
    double a,b,c,d;
    char e;
    while(scanf("%lf%lf%lf%lf %c",&a,&b,&c,&d,&e) != EOF) {
        if(e == '+') {
            print(a+c,b+d);
        }
        else if(e == '-') {
            print(a-c,b-d);
        }
        else if(e == '*') {
            print(a*c-b*d,b*c+a*d);
        }
        else {
            if(c == 0 && d == 0) {
                printf("??+??i\n");
            }
            else print((a*c+b*d)/(pow(c,2)+pow(d,2)),(b*c-
a*d)/(pow(c,2)+pow(d,2)));
        }
    }
    return 0;
}
```

闰年判断

```
#include<stdio.h>
int main() {
    int y;
    while(scanf("%d",&y)!=EOF){
        if((y % 4 == 0 && y % 100 != 0)||y % 400 == 0)
            printf("1\n");
        else
            printf("0\n");
    }
    return 0;
}
```

判断素数

```
#include <stdio.h>
#include <math.h>
int main()
{
    int i,a,flaga=0;
    scanf("%d",&a);
    for(i=2;i<sqrt(a);i++)
    {
        if(a%i==0)
        {
            flaga=1;
            break;
```

```
        }
    }
    if(flaga==1)
        printf("no");
    else
        printf("yes");
    return 0;
}
```

多项式求和（秦九韶算法）

```
#include <stdio.h>
int a[20];
int main(){
    int n,i,j,x;
    long long sum=0;
    scanf("%d %d",&n,&x);
    for(i=0;i<=n;i++)
        scanf("%d",&a[i]);
    for(i=n;i>=0;i--)
        sum=sum*x+a[i];
    printf("%lld",sum);
    return 0;
}
```

鸡兔同笼

```
#include<stdio.h>

int main()
{
    int h,f;
    scanf("%d%d",&h,&f);//h表示头的数量，f表示腿的数量
    printf("%d %d",h - (f-2*h) / 2,(f-2*h) / 2); //输出鸡的数量和兔的数量
    return 0;
}
```

水仙花数（“水仙花数”是指一个三位数，其各位数字的三次方和等于该数本身）

```
//输入一个范围[l,r]，输出范围内所有水仙花数，若没有，则输出no
#include <stdio.h>
int main()
{
    int start, end, i = 0, a, b, c, size = 0;
    while (scanf("%d %d", &start, &end) == 2)
    {
        for (i = start; i <= end; i++)
        {
            a = i / 100;
            b = i / 10 % 10;
            c = i % 10;
            //total = pow(c, 3) + pow(a, 3) + pow(b, 3);
            if ((a*a*a + b*b*b + c*c*c) == i)  //满足水仙花条件
            {
```

```
                if (size == 0)    //size=0输出第一个水仙花数
                {
                    printf("%d", i);
                }
                else      //size++输出第二。。第n个水仙花数
                {
                    printf(" %d", i);
                }
                size++;    //个数++;
            }
        }
        if (size == 0)    //范围内个数为0，则说明没有满足条件的
        {
            printf("no");
        }
        printf("\n");
    }
    return 0;
}
```

解一元二次方程（有确定边界的写法）

```
#include<stdio.h>
#include<math.h>

#define eps 1e-9

int equal(double a,double b) {
    if(a <= b + eps &&a >= b - eps) return 1;
    return 0;
}

int main() {
    double a,b,c,d;
    scanf("%lf%lf%lf",&a,&b,&c);
    if(a < 0) {
        a = -a;
        b = -b;
        c = -c;
    }
    d = pow(b,2)-4*a*c;
    if(equal(a,0.0)) printf("Not a quadratic");
    else if(equal(d,0.0)) {
        printf("Two equal roots:%10.5lf",(-b)/(2*a));
    }
    else if(d > 0) {
        printf("Two unequal real roots:%.6lf,%.6lf",((-b-sqrt(d))/(2*a)),(-
b+sqrt(d))/(2*a));
    }
    else printf("Two conjugate complex roots:%.5lf+%.5lfi %.5lf-%.5lfi",(-
b)/(2*a),sqrt(-d)/(2*a),(-b)/(2*a),sqrt(-d)/(2*a));
    return 0;
}
```

Ackman函数

```c
#include<stdio.h>
#include<math.h>

int a[5][100010];

long long Ack(int m,int n) {
    if(m == 0) return n+1;;
    if(n == 0) return Ack(m-1,1);
    return Ack(m-1,Ack(m,n-1));
}

int main(){
    int t;
    scanf("%d",&t);
    while(t --) {
        int m,n;
        scanf("%d%d",&m,&n);
        printf("%lld\n",Ack(m,n));
    }
    return 0;
}
```

汉诺塔

```c
#include<stdio.h>
#include<stdlib.h>

void move(char getone, char putone) {
    printf("%c-->%c\n", getone, putone);
}

void hanoit(int n, char a, char b, char c) {
    if(n == 1){
        move(a, c);
    } else {
        hanoit(n - 1, a, c, b);
        move(a, c);
        hanoit(n - 1, b, a, c);
    }

}

int main() {
    int m;

    scanf("%d", &m);
    hanoit(m, 'A', 'B', 'C');

    return 0;
}
```

二分法查找

```c
#include <stdio.h>
int a[100];
int main()
```

```
{
    int n,flag,l,r,mid,i,x;
    scanf("%d",&n);//数组大小n
    for(i=0;i<n;i++)
        scanf("%d",&a[i]); //数组a[n]
    while(~scanf("%d",&x)) //查找x，属于数组输出yes，否则输出no
    {
        flag=0;
        l=0;
        r=n-1;
        while(l<=r)
        {
            mid=(l+r)/2;
            if(a[mid]==x)
            {
                flag=1;
                break;
            }
            else if(a[mid]<x)
                l=mid+1;
            else
                r=mid-1;
        }
        if(flag==0)
            printf("no\n");
        else
            printf("yes\n");
    }
    return 0;
}
```

二分法解方程(例题为递减函数)

```
#include<stdio.h>
#include<math.h>

double f(double x) {
    return exp(-sqrt( x/10 ))/log(x/10);//函数方程
}

double Sol(double val,double u,double v) {
    double mid = (u + v)/2;
    if(v - u <= 0.000005) return mid;
    if(f(mid) < val) return Sol(val,u,mid);
    if(f(mid) > val) return Sol(val,mid,v);
    return mid;
}
int main() {
    double y;
    scanf("%lf",&y);
    printf("%.5lf",Sol(y,10,30));
    return 0;
}
/*
如果没有确定边界，则需要先找到初始边界
```

```
while(f(left)*f(right)>=0)
{
    if(f(left)==0)
    {
        printf("%.5f",left);
        return 0;
    }
    else if(f(right)==0)
    {
        printf("%.5f",right);
        return 0;
    }
    left*=2;
    right*=2;
}
```

求区间内元素平均值（前缀和）

```
#include<stdio.h>
#include<math.h>
long long b[1000010];
int a[1000010];

int main() {
    int m,n;
    scanf("%d%d",&m,&n);
    for(int i = 1;i <= n;i ++) {
        scanf("%d",&a[i]);
        b[i] = b[i-1] + a[i];
    }
    for(int i = 0;i < m;i ++) {
        int l,r;
        scanf("%d%d",&l,&r);
        printf("%lld\n",(b[r]-b[l-1])/(r-l+1));
    }
    return 0;
}
```

冒泡二级排序

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
#include<string.h>
int tem[2010],num[2010],score[2010];
int check[100000000];
int main()
{
    int n=0,j,i;
    while(scanf("%d %d",&num[n],&score[n])!=EOF)//先按score降序，再按num升序
    {
        tem[n]=n;
        n++;
    }
    for(i=0;i<n-1;i++)
    {
```

```
        for(j=i+1;j<n;j++)
        {
            if(score[tem[i]]<score[tem[j]])
            {
                tem[i]=tem[i]^tem[j];
                tem[j]=tem[i]^tem[j];
                tem[i]=tem[i]^tem[j];
            }
        }
    }
    for(i=0;i<n-1;i++)
    {
        for(j=i+1;j<n;j++)
        {
            if(score[tem[i]]==score[tem[j]]&&num[tem[i]]>num[tem[j]])
            {
                tem[i]=tem[i]^tem[j];
                tem[j]=tem[i]^tem[j];
                tem[i]=tem[i]^tem[j];
            }
        }
    }
    for(i=0;i<n;i++)
    {
        if(check[num[tem[i]]]==0)
        {
            printf("%d %d\n",num[tem[i]],score[tem[i]]);
            check[num[tem[i]]]++;
        }
    }
    return 0;
}
```

判断两个数字二进制下1的个数是否相同

```
#include<stdio.h>
#include<complex.h>

int main() {
    int T;
    scanf("%d",&T);
    while(T --) {
        int x,y,ans1=0,ans2=0;
        scanf("%d%d",&x,&y);
        while(x != 0) {
            ans1 ++;
            x = (x-1)&x;
        }
        while(y != 0) {
            ans2 ++;
            y = (y-1)&y;
        }
        if(ans1==ans2) printf("branch\n");
        else printf("in-order\n");
    }
}
```

求这一天是这一年的第几天

```
#include<stdio.h>
int main() {
    int y,m,d,ans = 0;
    scanf("%d%d%d",&y,&m,&d);//输入年月日
    int a[]={0,31,28,31,30,31,30,31,31,30,31,30,31};
    if(y%4==0&&y%100!=0||y%400==0) a[2]++;
    for(int i = 1;i<m;i ++) {
        ans += a[i];
    }
    ans += d;
    printf("%d",ans);
}
```

高精度加法

```
#include<stdio.h>
#include<string.h>
char s[10100],ss[10100];
int a[10100],b[10100];
int len;

int main()
{
    scanf("%s%s",s,ss);
    int l1 = strlen(s);
    int l2 = strlen(ss);
    if (l1 > l2)
        len = l1;
    else
        len = l2;
    for (int i = l1 - 1 ; i >= 0 ; i--)
        a[l1 - i - 1] = s[i] - '0';
    for (int i = l2 - 1 ; i >= 0 ; i--)
        b[l2 - i - 1] = ss[i] - '0';
    for (int i = 0 ; i < len ; i++)
    {
        a[i] = a[i] + b[i];
        a[i+1]+= a[i] / 10;
    }
    if (a[len] != 0) len++;
    while (a[len - 1] == 0 && len>1) len--;
    for (int i = len - 1 ;i >= 0 ;i--)
        printf("%d",a[i]);
    printf("\n");
    return 0;
}
```

高精度减法

```
#include <stdio.h>
#include <string.h>

char s[10100],ss[10100];
int a[10100],b[10100];
```

```
int len;


int main()
{
    scanf("%s%s",s,ss);
    int l1 = strlen(s);
    int l2 = strlen(ss);
    int flag = 0;

    memset(a,0,sizeof(a));
    memset(b,0,sizeof(b));

    if ( l1 < l2 || (strcmp(s,ss) < 0 && l1 == l2) )
    {
        flag = 1;
        for (int i = l2 - 1 ; i >= 0 ; i--)
            a[l2 - i - 1] = ss[i] - '0';
        for (int i = l1 - 1 ; i >= 0 ; i--)
            b[l1 - i - 1] = s[i] - '0';
    }
    else
    {
        for (int i = l1 - 1 ; i >= 0 ; i--)
            a[l1 - i - 1] = s[i] - '0';
        for (int i = l2 - 1 ; i >= 0 ; i--)
            b[l2 - i - 1] = ss[i] - '0';
    }
    if (l1 > l2)
        len = l1;
    else
        len = l2;
    for (int i = 0 ; i < len ; i++)
    {
        a[i] = a[i] - b[i];
        if (a[i] < 0)
        {
            a[i+1]-=1;
            a[i]+=10;
        }
    }

    while (a[len - 1] == 0 && len>1) len--;
    if (flag == 1) printf("-");
    for (int i = len - 1 ;i >= 0 ;i--)
        printf("%d",a[i]);
    printf("\n");
    return 0;
}
```

高精度乘法

```
#include<stdio.h>
#include<string.h>
char s1[50010],s2[50010];
int a[50010],b[50010],c[100010];
int main()
```

```
{
    int i,j,len;
    while(~scanf("%s %s",s1,s2))
    {
        if(s1[0]=='0'||s2[0]=='0')
            printf("0\n");
        else if(s1[0]=='-'&&s2[0]=='-')
        {
            for(i=0;i<strlen(s1)-1;i++)
                s1[i]=s1[i+1];
            s1[i]='\0';
            for(i=0;i<strlen(s2)-1;i++)
                s2[i]=s2[i+1];
            s2[i]='\0';
        }
        else if(s1[0]=='-')
        {
            printf("-");
            for(i=0;i<strlen(s1)-1;i++)
                s1[i]=s1[i+1];
            s1[i]='\0';
        }
        else if(s2[0]=='-')
        {
            printf("-");
            for(i=0;i<strlen(s2)-1;i++)
                s2[i]=s2[i+1];
            s2[i]='\0';
        }
        memset(a,0,sizeof(a));
        memset(b,0,sizeof(b));
        memset(c,0,sizeof(c));
        for(i=0;i<strlen(s1);i++)
            a[i]=s1[strlen(s1)-i-1]-'0';
        for(i=0;i<strlen(s2);i++)
            b[i]=s2[strlen(s2)-i-1]-'0';
        for(i=0;i<strlen(s1);i++)
        {
            for(j=0;j<strlen(s2);j++)
            {
                c[i+j]+=a[i]*b[j];
                c[i+j+1]+=c[i+j]/10;
                c[i+j]%=10;
            }
        }
        len=strlen(s1)+strlen(s2);
        while(c[len]==0) len--;
        for(i=len;i>=0;i--)
            printf("%d",c[i]);
        printf("\n");
    }
    return 0;
}
```

高精度除法

```
#include <stdio.h>
```

```
#include <string.h>

int len1,len2;
char s1[905],s2[905];
int re[905];

void sub()
{
    int i=0,j;
    while(1)
    {
        if(s1[i]=='0')  i++;
        else
        {
            j=i;
            break;
        }
    }
    for(;i<len2;i++)
        s1[i]=s1[i]-s2[i]+'0';
    for(i=len2-1;i>j;i--)            //低位开始检测是否小于0
        if(s1[i]<'0')
        {
            s1[i]+=10;
            s1[i-1]--;
        }
}

int main()
{
    int i,p;
    scanf("%s%s",s1,s2);
    len1=strlen(s1);
    len2=strlen(s2);
    if(len1<len2 || (len1==len2 && strncmp(s1,s2,len2)<0))   //如果a<b,直接输出0
        printf("0\n");
    p=0;
    while(1)
    {
        re[p]=0;
        while(strncmp(s1,s2,len2)>=0)       //一直进行减法，直到不能减为止
        {
            sub();
            re[p]++;
        }
        p++;
        if(len1==len2)  break;
        for(i=len2-1;i>=0;i--)              //在s2前面补0，以便进行减法运算
            s2[i+1]=s2[i];
        s2[0]='0';
        len2++;
        s2[len2]='\0';
    }
    i=0;
    while(1)
    {
        if(re[i]==0)    i++;
        else            break;
```

```
    }
    for(;i<p;i++)
        printf("%d",re[i]);
    return 0;
}
```

求阶乘(高精度)

```
#include<stdio.h>
#include<string.h>

int a[100000],n,i,y,xy[100000];

int main() {
    scanf("%d",&n);
    a[0]=1;
    a[1]=1;
    for (y=1;y<=n;y++) {
        memset(xy,0,sizeof(xy));
        xy[0]=a[0];
        for (i=1;i<=a[0];i++) {
          xy[i]+=a[i]*y;
          xy[i+1]=xy[i]/10;
          xy[i]%=10;
        }
        while (xy[xy[0]+1]>0) {
            xy[xy[0]+2]=xy[xy[0]+1]/10;
            xy[xy[0]+1]%=10;
            xy[0]++;
        }
        for (i=1;i<=xy[0];i++) a[i]=xy[i];
        a[0]=xy[0];
    }
    for (i=a[0];i>=1;i--) printf("%d",a[i]);
    return 0;
}
```

DFS迷宫

迷宫为m×n格（表示有m行、n列），其中有可走的也有不可走的，用1表示可以走，0表示不可以走。入口为第1行第1列（即左上角），出口为第 m 行第 n 列（即右下角），保证这两个点位都是1。请判断是否能走出迷宫。

如果能，请计算通过迷宫要行走的距离（规定只有一条出路且每一步的距离为1）。

如果不能，请计算可以到达的离起点直线距离最远的点

```
#include<stdio.h>
#include<math.h>

int a[30][30],b[30][30],m,n,yes,ans,c = 1,d = 1,l;

void dfs (int x,int y) {
    if(x == m && y == n) {
        yes = 1;
```

```
    }
    if(yes == 1) return ;
    if( b[x+1][y] == 0 && a[x+1][y] == 1) {
        ans ++;
        b[x+1][y] = 1;
        if(l < pow(x,2) + pow(y-1,2)) {
            l = pow(x,2) + pow(y-1,2);
            c = x+1;
            d = y;
        }
        dfs(x+1,y);
        if(yes == 1) return ;
        ans --;
    }
    if( b[x][y+1] == 0 && a[x][y+1] == 1) {
        ans ++;
        b[x][y+1] = 1;
        if(l < pow(x-1,2) + pow(y,2)) {
            l = pow(x-1,2) + pow(y,2);
            c = x;
            d = y+1;
        }
        dfs(x,y+1);
        if(yes == 1) return;
        ans --;
    }
    if( b[x-1][y] == 0 && a[x-1][y] == 1) {
        ans ++;
        b[x-1][y] = 1;
        dfs(x-1,y);
        if(yes == 1) return;
        ans --;
    }
    if( b[x][y-1] == 0 && a[x][y-1] == 1) {
        ans ++;
        b[x][y-1] = 1;
        dfs(x,y-1);
        if(yes == 1) return;
        ans--;
    }
    return;
}

int main() {
    scanf("%d%d",&m,&n);
    for(int i = 1;i <= m;i ++) {
        for(int j = 1;j <= n;j ++) {
            scanf("%d",&a[i][j]);
        }
    }
    dfs(1,1);
    if(yes == 1) printf("%d",ans);
    else printf("(%d,%d)",c,d);
    return 0;
}
```

求a[i]-a[j]的最大值

```
#include<stdio.h>

int max(int x,int y) {
    if(x < y) return y;
    return x;
}

int main() {
    int T;
    scanf("%d",&T);
    while (T --) {
        int n,ma = -1000000,ans = -1000000;
        scanf("%d",&n);
        for(int i = 1;i <= n;i ++) {
            int t;
            scanf("%d",&t);
            ans = max(ans,ma - t);
            ma = max(ma,t);
        }
        printf("%d\n",ans);
    }
}
```

RMQ问题（求区间内最大的元素）

```
//给定长度为N的序列，M个询问，每次询问两个数字A,B，要求求出属于A到B这段区间内的最大数。
#include<stdio.h>
#define max(a,b) a>b?a:b
int a[200010];
int dp[200010][32];
int RMQ(int l,int r)
{
    int k=0;
    while(1<<(k+1)<=r-l+1)
        k++;
    return max(dp[l][k],dp[r-(1<<k)+1][k]);
}
int main()
{
    int n,m,l,r,i,j;
    scanf("%d",&n);
    for(i=1;i<=n;i++)
        scanf("%d",&a[i]);
    for(i=1;i<=n;i++)
        dp[i][0]=a[i];
    for(j=1;(1<<j)<=n;j++)
        for(i=1;i+(1<<j)-1<=n;i++)
            dp[i][j]=max(dp[i][j-1],dp[i+(1<<(j-1))][j-1]);
    scanf("%d",&m);
    for(i=0;i<m;i++)
    {
        scanf("%d %d",&l,&r);
        printf("%d\n",RMQ(l,r));
    }
    return 0;
}
```

动态规划之考试得分

请通过这道题来感受一下动态规划的基本思想:

在一场竞赛中有n道题目，每道题都会对应一个分值，你可以选择任意一道题作答，比如你选择了x分的题目，那么在作答完毕（假设一定可以得分）后，你将获得x分，但是这场比赛中分值**等于**x−1和x+1的其他题目就会消失，那么这场比赛中你最多可以得到多少分?

输入第一个数为题目总数n，接下来为n个n个整数a1，a2，a3，表示各个题目的得分。输出你可以得到的最高分。

```
#include<stdio.h>
#include<limits.h>
int max(int a, int b){
    return a > b ? a :b;
}
int a[100010],f[100010],b[100010];
int main()
{
    int n,i,x=0,ans=0;
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        scanf("%d",&a[i]);
        b[a[i]]++;
        x=max(a[i],x);
    }
    f[1]=b[1];
    for(i=2;i<=x;i++)
    {
        f[i]=max(f[i-1],f[i-2]+i*b[i]);
    }
    printf("%d",max(f[x],f[x-1]));
    return 0;
}
```

最大连续子序列和

```
/*
input:-2 11 -4 13 -5 -2
output: 20 (11-4+13)
*/
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

#define maxn 10010
int A[maxn], dp[maxn];    // A[i] 存放序列，dp[i] 存放以 A[i] 为结尾的连续序列的最大和

// 求较大值
int max(int a, int b) {
    return a>b ? a : b;
}

int main() {
    int n, i, k;
```

```
    scanf("%d", &n);
    for(i=0; i<n; ++i) {        // 输入序列
        scanf("%d", &A[i]);
    }
    dp[0] = A[0];                // 边界
    for(i=1; i<n; ++i) {
        // 状态转移方程
        dp[i] = max(A[i], dp[i-1] + A[i]);
    }
    // 求最大连续子序列和
    k = dp[0];
    for(i=1; i<n; ++i) {
        if(dp[i] > k) {
            k = dp[i];
        }
    }
    printf("%d\n", k);        // 输出

    return 0;
}
```