# 2016蓝桥杯c-c++B组省赛试题及解析

**第一题**

煤球数目

有一堆煤球，堆成三角棱锥形。具体：

第一层放1个，

第二层3个（排列成三角形），

第三层6个（排列成三角形），

第四层10个（排列成三角形），

....

如果一共有100层，共有多少个煤球？

请填表示煤球总数目的数字。

注意：你提交的应该是一个整数，不要填写任何多余的内容或说明性文字。

**答案：171700**

**#include<stdio.h>**

**int main()**

**{**

**int a[101] ={0};**

**for(int i = 1 ; i < 101 ; i ++)**

**a[i] = a[i-1] + i;**

**int ans = 0;**

**for(int j = 1 ; j < 101 ; j ++)**

**ans += a[j];**

**printf("%d\n",ans);**

**return 0;**

**}**

**第二题**

生日蜡烛

某君从某年开始每年都举办一次生日party，并且每次都要吹熄与年龄相同根数的蜡烛。

现在算起来，他一共吹熄了236根蜡烛。

请问，他从多少岁开始过生日party的？

请填写他开始过生日party的年龄数。

注意：你提交的应该是一个整数，不要填写任何多余的内容或说明性文字。

**答案：26**

**#include<stdio.h>**

**int main()**

**{**

**int start,end;**

**for(start = 1 ; start < 236 ; start ++)**

**{**

**for( end = start ; end < 236 ; end ++ )**

**{**

**int sum = 0;**

**for(int i = start; i <= end; i ++)**

**sum += i;**

**if( sum == 236)**

**{**

**printf("start : %d end : %d\n",start,end);**

**}**

**}**

**}**

**return 0;**

**}**

**第三题**

凑算式

B DEF

A + --- + ------- = 10

C GHI

（如果显示有问题，可以参见【图1.jpg】）

这个算式中A~I代表1~9的数字，不同的字母代表不同的数字。

比如：

6+8/3+952/714 就是一种解法，

5+3/1+972/486 是另一种解法。

这个算式一共有多少种解法？

注意：你提交应该是个整数，不要填写任何多余的内容或说明性文字。

**答案：29**

**#include<stdio.h>**

**int ans = 0;**

**int num[10];**

**bool visit[10];**

**void Solve()**

**{**

**double sum = num[0] + (double)num[1] / num[2] + (double)(num[3]\*100+num[4]\*10+num[5])/(num[6]\*100+num[7]\*10+num[8]);**

**if(sum == 10)**

**{**

**ans ++;**

**}**

**}**

**void dfs(int index)**

**{**

**if(index == 9)**

**{**

**Solve();**

**return ;**

**}**

**for(int i = 1 ; i < 10 ; i ++)**

**{**

**if(!visit[i])**

**{**

**visit[i] = true;**

**num[index] = i;**

**dfs(index+1);**

**visit[i] = false;**

**}**

**}**

**}**

**int main()**

**{**

**dfs(0);**

**printf("%d\n",ans);**

**return 0;**

**}**

**第四题**

快速排序

排序在各种场合经常被用到。

快速排序是十分常用的高效率的算法。

其思想是：先选一个“标尺”，

用它把整个队列过一遍筛子，

以保证：其左边的元素都不大于它，其右边的元素都不小于它。

这样，排序问题就被分割为两个子区间。

再分别对子区间排序就可以了。

下面的代码是一种实现，请分析并填写划线部分缺少的代码。

#include <stdio.h>

void swap(int a[], int i, int j)

{

int t = a[i];

a[i] = a[j];

a[j] = t;

}

int partition(int a[], int p, int r)

{

int i = p;

int j = r + 1;

int x = a[p];

while(1){

while(i<r && a[++i]<x);

while(a[--j]>x);

if(i>=j) break;

swap(a,i,j);

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

return j;

}

void quicksort(int a[], int p, int r)

{

if(p<r){

int q = partition(a,p,r);

quicksort(a,p,q-1);

quicksort(a,q+1,r);

}

}

int main()

{

int i;

int a[] = {5,13,6,24,2,8,19,27,6,12,1,17};

int N = 12;

quicksort(a, 0, N-1);

for(i=0; i<N; i++) printf("%d ", a[i]);

printf("\n");

return 0;

}

注意：只填写缺少的内容，不要书写任何题面已有代码或说明性文字。

**答案：**swap(a,p,j)****

****第五题****

**抽签**

**X星球要派出一个5人组成的观察团前往W星。**

**其中：**

**A国最多可以派出4人。**

**B国最多可以派出2人。**

**C国最多可以派出2人。**

**....**

**那么最终派往W星的观察团会有多少种国别的不同组合呢？**

**下面的程序解决了这个问题。**

**数组a[] 中既是每个国家可以派出的最多的名额。**

**程序执行结果为：**

**DEFFF**

**CEFFF**

**CDFFF**

**CDEFF**

**CCFFF**

**CCEFF**

**CCDFF**

**CCDEF**

**BEFFF**

**BDFFF**

**BDEFF**

**BCFFF**

**BCEFF**

**BCDFF**

**BCDEF**

**....**

**(以下省略，总共101行)**

**#include <stdio.h>**

**#define N 6**

**#define M 5**

**#define BUF 1024**

**void f(int a[], int k, int m, char b[])**

**{**

**int i,j;**

**if(k==N){**

**b[M] = 0;**

**if(m==0) printf("%s\n",b);**

**return;**

**}**

**for(i=0; i<=a[k]; i++){**

**for(j=0; j<i; j++) b[M-m+j] = k+'A';**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; //填空位置**

**}**

**}**

**int main()**

**{**

**int a[N] = {4,2,2,1,1,3};**

**char b[BUF];**

**f(a,0,M,b);**

**return 0;**

**}**

**仔细阅读代码，填写划线部分缺少的内容。**

**注意：不要填写任何已有内容或说明性文字。**

****答案f(a,k+1,m-j,b)或f(a,k+1,m-i,b)****

****第六题****

**方格填数**

**如下的10个格子**

**+--+--+--+**

**| | | |**

**+--+--+--+--+**

**| | | | |**

**+--+--+--+--+**

**| | | |**

**+--+--+--+**

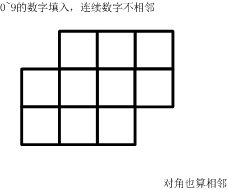
**（如果显示有问题，也可以参看【图1.jpg】）**

**填入0~9的数字。要求：连续的两个数字不能相邻。**

**（左右、上下、对角都算相邻）**

**一共有多少种可能的填数方案？**

**请填写表示方案数目的整数。**

**注意：你提交的应该是一个整数，不要填写任何多余的内容或说明性文字。**

**图1.jpg**

****答案是:1580****

****#include <stdio.h>****

****#include <math.h>****

****int flag[3][4]; //表示哪些可以填数****

****int mpt[3][4]; //填数****

****bool visit[10];****

****int ans = 0;****

****void init() //初始化****

****{****

****int i,j;****

****for(i = 0 ; i < 3 ; i ++)****

****for(j = 0 ; j < 4 ; j ++)****

****flag[i][j] = 1;****

****flag[0][0] = 0;****

****flag[2][3] = 0;****

****}****

****void Solve()****

****{****

****int dir[8][2] = { 0,1,0,-1,1,0,-1,0,1,1,1,-1,-1,1,-1,-1};****

****int book = true;****

****for(int i = 0 ; i < 3 ; i ++)****

****{****

****for(int j = 0 ; j < 4; j ++)****

****{****

****//判断每个数周围是否满足****

****if(flag[i][j] == 0)continue;****

****for( int k = 0 ; k < 8 ; k ++)****

****{****

****int x,y;****

****x = i + dir[k][0];****

****y = j + dir[k][1];****

****if(x < 0 || x >= 3 || y < 0 || y >= 4 || flag[x][y] == 0) continue;****

****if(abs(mpt[x][y] - mpt[i][j]) == 1) book = false;****

****}****

****}****

****}****

****if(book) ans ++;****

****}****

****void dfs(int index)****

****{****

****int x,y;****

****x = index / 4;****

****y = index % 4;****

****if( x == 3)****

****{****

****Solve();****

****return;****

****}****

****if(flag[x][y])****

****{****

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

****{****

****if(!visit[i])****

****{****

****visit[i] = true;****

****mpt[x][y] = i;****

****dfs(index+1);****

****visit[i] = false;****

****}****

****}****

****}****

****else****

****{****

****dfs(index+1);****

****}****

****}****

****int main()****

****{****

****init();****

****dfs(0);****

****printf("%d\n",ans);****

****return 0;****

****}****

**第七题**

剪邮票

如【图1.jpg】, 有12张连在一起的12生肖的邮票。

现在你要从中剪下5张来，要求必须是连着的。

（仅仅连接一个角不算相连）

比如，【图2.jpg】，【图3.jpg】中，粉红色所示部分就是合格的剪取。

请你计算，一共有多少种不同的剪取方法。

请填写表示方案数目的整数。

注意：你提交的应该是一个整数，不要填写任何多余的内容或说明性文字。

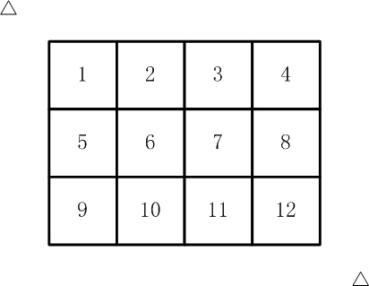
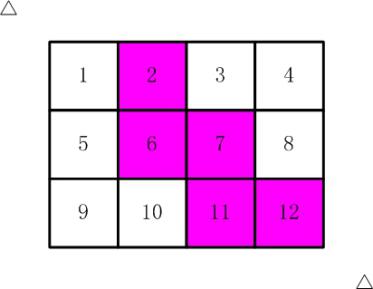
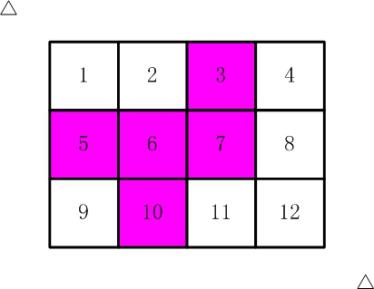


图1.jpg

图2.jpg

图3.jpg

**答案：116**

**#include <stdio.h>**

**#include <string.h>**

**int mpt[3][4];**

**int mpt\_visit[3][4];**

**int num[6];**

**int have[13];**

**int visit[13];**

**int ans = 0;**

**int Count = 0;**

**void init()**

**{**

**int k = 1;**

**for(int i = 0 ; i < 3 ; i ++)**

**for(int j = 0 ; j < 4 ; j ++)**

**{**

**mpt[i][j] = k;**

**k ++;**

**}**

**}**

**int dir[4][2] = {0,1,0,-1,-1,0,1,0};**

**//判断五个数是否能连在一起**

**void dfs\_find(int x,int y)**

**{**

**for(int i = 0 ; i < 4 ; i++)**

**{**

**int tx,ty;**

**tx = x + dir[i][0];**

**ty = y + dir[i][1];**

**if(tx < 0 || tx >= 3 || ty < 0 || ty >= 4) continue;**

**if(have[mpt[tx][ty]] == 0 || mpt\_visit[tx][ty])continue;**

**mpt\_visit[tx][ty] = 1;**

**Count ++;**

**dfs\_find(tx,ty);**

**}**

**}**

**void Solve()**

**{**

**int i;**

**memset(have,0,sizeof(have));**

**memset(mpt\_visit,0,sizeof(mpt\_visit));**

**for(i = 1; i < 6 ; i ++) have[num[i]] = 1;**

**for(i = 0 ; i < 12 ; i ++)**

**{**

**int x,y;**

**x = i / 4;**

**y = i % 4;**

**if(have[mpt[x][y]])**

**{**

**Count = 1;**

**mpt\_visit[x][y] =1;**

**dfs\_find(x,y);**

**break;**

**}**

**}**

**if(Count == 5)**

**{**

**ans ++;**

**}**

**}**

**//创建5个数的组合**

**void dfs\_creat(int index)**

**{**

**if(index == 6)**

**{**

**Solve();**

**return;**

**}**

**for(int i = num[index-1] + 1; i < 13 ; i ++)**

**{**

**if(!visit[i])**

**{**

**visit[i] = true;**

**num[index] = i;**

**dfs\_creat(index+1);**

**visit[i] = false;**

**}**

**}**

**}**

**int main()**

**{**

**init();**

**dfs\_creat(1);**

**printf("%d\n",ans);**

**return 0;**

**}**

**第八题**

四平方和

四平方和定理，又称为拉格朗日定理：

每个正整数都可以表示为至多4个正整数的平方和。

如果把0包括进去，就正好可以表示为4个数的平方和。

比如：

5 = 0^2 + 0^2 + 1^2 + 2^2

7 = 1^2 + 1^2 + 1^2 + 2^2

（^符号表示乘方的意思）

对于一个给定的正整数，可能存在多种平方和的表示法。

要求你对4个数排序：

0 <= a <= b <= c <= d

并对所有的可能表示法按 a,b,c,d 为联合主键升序排列，最后输出第一个表示法

程序输入为一个正整数N (N<5000000)

要求输出4个非负整数，按从小到大排序，中间用空格分开

例如，输入：

5

则程序应该输出：

0 0 1 2

再例如，输入：

12

则程序应该输出：

0 2 2 2

再例如，输入：

773535

则程序应该输出：

1 1 267 838

资源约定：

峰值内存消耗 < 256M

CPU消耗 < 3000ms

请严格按要求输出，不要画蛇添足地打印类似：“请您输入...” 的多余内容。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

注意: main函数需要返回0

注意: 只使用ANSI C/ANSI C++ 标准，不要调用依赖于编译环境或操作系统的特殊函数。

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

提交时，注意选择所期望的编译器类型。

**答案：**

**方法一：**

**#include <stdio.h>**

**#include <math.h>**

**int main()**

**{**

**int n;**

**int flag = false;**

**scanf("%d",&n);**

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

**{**

**for(int j = 0 ; j \* j <= n ; j ++){**

**for(int k = 0 ; k \* k <= n ; k ++)**

**{**

**int temp = n - i\*i - j\*j - k\*k;**

**double l = sqrt((double) temp);**

**if(l == (int)l )**

**{**

**printf("%d %d %d %d\n",i,j,k,(int)l);**

**flag = true;**

**break;**

**}**

**}**

**if(flag)break;**

**}**

**if(flag)break;**

**}**

**return 0;**

**}**

**方法二：**

**#include <stdio.h>**

**#include <math.h>**

**int mpt[5000010] ={0}; //mpt[i] = 1表示i 能够用两个完全平方数相加而得。**

**int n;**

**void init()**

**{**

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

**for(int j = 0 ; j\*j <=n ; j ++)**

**if(i\*i+j\*j <= n) mpt[i\*i+j\*j] = 1;**

**}**

**int main()**

**{**

**int flag = false;**

**scanf("%d",&n);**

**init();**

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

**{**

**for(int j = 0 ; j \* j <= n ; j ++){**

**if(mpt[n - i\*i - j\*j] == 0) continue; //如果剩下的差用两个完全平方数不能组合出来就不继续**

**for(int k = 0 ; k \* k <= n ; k ++)**

**{**

**int temp = n - i\*i - j\*j - k\*k;**

**double l = sqrt((double) temp);**

**if(l == (int)l )**

**{**

**printf("%d %d %d %d\n",i,j,k,(int)l);**

**flag = true;**

**break;**

**}**

**}**

**if(flag)break;**

**}**

**if(flag)break;**

**}**

**return 0;**

**}**

**第九题**

交换瓶子

有N个瓶子，编号 1 ~ N，放在架子上。

比如有5个瓶子：

2 1 3 5 4

要求每次拿起2个瓶子，交换它们的位置。

经过若干次后，使得瓶子的序号为：

1 2 3 4 5

对于这么简单的情况，显然，至少需要交换2次就可以复位。

如果瓶子更多呢？你可以通过编程来解决。

输入格式为两行：

第一行: 一个正整数N（N<10000）, 表示瓶子的数目

第二行：N个正整数，用空格分开，表示瓶子目前的排列情况。

输出数据为一行一个正整数，表示至少交换多少次，才能完成排序。

例如，输入：

5

3 1 2 5 4

程序应该输出：

3

再例如，输入：

5

5 4 3 2 1

程序应该输出：

2

资源约定：

峰值内存消耗 < 256M

CPU消耗 < 1000ms

请严格按要求输出，不要画蛇添足地打印类似：“请您输入...” 的多余内容。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

注意: main函数需要返回0

注意: 只使用ANSI C/ANSI C++ 标准，不要调用依赖于编译环境或操作系统的特殊函数。

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

提交时，注意选择所期望的编译器类型。

**第十题**

最大比例

X星球的某个大奖赛设了M级奖励。每个级别的奖金是一个正整数。

并且，相邻的两个级别间的比例是个固定值。

也就是说：所有级别的奖金数构成了一个等比数列。比如：

16,24,36,54

其等比值为：3/2

现在，我们随机调查了一些获奖者的奖金数。

请你据此推算可能的最大的等比值。

输入格式：

第一行为数字 N (0<N<100)，表示接下的一行包含N个正整数

第二行N个正整数Xi(Xi<1 000 000 000 000)，用空格分开。每个整数表示调查到的某人的奖金数额

要求输出：

一个形如A/B的分数，要求A、B互质。表示可能的最大比例系数

测试数据保证了输入格式正确，并且最大比例是存在的。

例如，输入：

3

1250 200 32

程序应该输出：

25/4

再例如，输入：

4

3125 32 32 200

程序应该输出：

5/2

再例如，输入：

3

549755813888 524288 2

程序应该输出：

4/1

资源约定：

峰值内存消耗 < 256M

CPU消耗 < 3000ms

请严格按要求输出，不要画蛇添足地打印类似：“请您输入...” 的多余内容。

所有代码放在同一个源文件中，调试通过后，拷贝提交该源码。

注意: main函数需要返回0

注意: 只使用ANSI C/ANSI C++ 标准，不要调用依赖于编译环境或操作系统的特殊函数。

注意: 所有依赖的函数必须明确地在源文件中 #include <xxx>， 不能通过工程设置而省略常用头文件。

提交时，注意选择所期望的编译器类型。

**答案：**

**#include <stdio.h>**

**#include <algorithm>**

**#include <queue>**

**using namespace std;**

**#define LL long long**

**struct fs**

**{**

**LL up,down;**

**};**

**int n;**

**LL arr[110];**

**fs Fs[110];**

**bool cmp(LL a,LL b)**

**{**

**return a > b;**

**}**

**LL Gcd(LL a,LL b)**

**{**

**if( b == 0 )return a;**

**return Gcd(b,a%b);**

**}**

**LL Get(LL a, LL b)**

**{**

**if( a < b) a ^= b ^= a ^= b;**

**LL v[30];**

**queue<LL>team;**

**if( a == b || a / b == a) return b;**

**v[0] = a, v[1] = b;**

**v[2] = a / b;**

**int top = 3,i,j;**

**team.push(a/b);**

**while(team.size())**

**{**

**LL now = team.front();**

**team.pop();**

**for(i = 0 ; i < top ; i ++)**

**{**

**LL temp = (v[i] > now) ? v[i] / now : now / v[i];**

**bool find = false;**

**for(j = 0 ; j < top ; j ++)**

**if( v[j] == temp) find = true;**

**if(find == true) continue;**

**team.push(temp);**

**v[top++] = temp;**

**}**

**}**

**LL ans = v[0];**

**for(i = 0 ; i < top ; i ++)**

**if(v[i] != 1)**

**{**

**ans = v[i];**

**break;**

**}**

**for(i = 0 ; i < top ; i ++)**

**if( v[i] < ans && v[i] != 1) ans = v[i];**

**return ans;**

**}**

**int main()**

**{**

**int i,j;**

**scanf("%d",&n);**

**for(i = 0 ; i < n ; i ++) scanf("%lld",&arr[i]);**

**sort(arr,arr+n,cmp);**

**int top = 1;**

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

**if(arr[i] != arr[i-1]) arr[top++] = arr[i];**

**n = top;**

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

**{**

**LL gcd = Gcd(arr[i],arr[i+1]);**

**Fs[i].up = arr[i] / gcd;**

**Fs[i].down = arr[i+1] / gcd;**

**}**

**LL x = Fs[0].up;**

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

**x = Get(x,Fs[i].up);**

**LL y = Fs[0].down;**

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

**y = Get(y,Fs[i].down);**

**printf("%lld/%lld\n",x,y);**

**return 0;**

**}**