**一、单项选择题**

**1.**下列说法不正确的是：

A.SATA硬盘的速度速度大约为500Mbps/s

B.读取18XDVD光盘数据的速度为1Gbps

C.前兆以太网的数据读取速度为1Gpbs

D.读取DDR3内存数据的速度为100Gbps

**2.**（）不能用于Linux中的进程通信

A.共享内存

B.命名管道

C.信号量

D.临界区

**3.**设在内存中有P1,P2,P3三道程序，并按照P1,P2,P3的优先级次序运行，其中内部计算和IO操作时间由下表给出（CPU计算和IO资源都只能同时由一个程序占用）：

P1:计算60ms---》IO 80ms---》计算20ms

P2:计算120ms---》IO 40ms---》计算40ms

P3:计算40ms---》IO 80ms---》计算40ms

完成三道程序比单道运行节省的时间是（）

A.80ms

B.120ms

C.160ms

D.200ms

**4.**两个等价线程并发的执行下列程序，a为全局变量，初始为0，假设printf、++、--操作都是原子性的，则输出不肯哪个是（）

void foo() {

    if(a <= 0) {

        a++;

    }

    else {

        a--;

    }

    printf("%d", a);

}

A.01

B.10

C.12

D.22

**5.**给定fun函数如下，那么fun(10)的输出结果是（）

int fun(int x) {

    return (x==1) ? 1 : (x + fun(x-1));

}

A.0

B.10

C.55

D.3628800

**6.**在c++程序中，如果一个整型变量频繁使用，最好将他定义为（）

A.auto

B.extern

C.static

D.register

**7.**长为n的字符串中匹配长度为m的子串的复杂度为（）

A.O(N)

B.O(M+N)

C.O(N+LOGM)

D.O(M+LOGN)

**8.**判断一包含n个整数a[]中是否存在i、j、k满足a[i] + a[j] = a[k]的时间复杂度为（）

A.O(n) B.O(n^2) C.O(nlog(n)) D.O(n^2log(n))

**9.**三次射击能中一次的概率是0.95，请问一次射击能中的概率是多少？  
A.0.63

B.0.5

C.\*\*

D.0.85

**10.**下列序排算法中最坏复杂度不是n(n-1)/2的是\_

A.快速序排 B.冒泡序排 C.直接插入序排 D.堆序排

**二、不定向选择题**

**1.**以下哪些进程状态转换是正确的()

A.就绪到运行 B.运行到就绪 C.运行到阻塞 D.阻塞到运行 E.阻塞到就绪

**2.**一个栈的入栈数列为：1、2、3、4、5、6；下列哪个是可能的出栈顺序。（选项不记得）

**3.**下列哪些代码可以使得a和b交换数值。（选项不记得）

**4.**A和B晚上无聊就开始数星星。每次只能数K个（20<=k<=30）A和B轮流数。最后谁把星星数完谁就获胜，那么当星星数量为多少时候A必胜？（选项不记得）

**三、填空问答题**

**1.**给你一个整型数组A[N]，完成一个小程序代码（20行之内），使得A[N]逆向，即原数组为1，2，3，4，逆向之后为4，3，2，1

void revense(int \* a,int n) {

}

**2.**自选调度方面的问题，题目很长，就是给你三个线程，分别采用先来先分配的策略和最短执行之间的调度策略，然后计算每个线程从提交到执行完成的时间。题目实在太长，还有几个表格。考察的是操作系统里面作业调度算法先进先出和最短作业优先。

**3.**有 个苦逼的上班族，他每天忘记定闹钟的概率为0.2，上班堵车的概率为0.5，如果他既没定闹钟上班又堵车那他迟到的概率为1.0，如果他定了闹钟但是上班 堵车那他迟到的概率为0.9，如果他没定闹钟但是上班不堵车他迟到的概率为0.8，如果他既定了闹钟上班又不堵车那他迟到的概率为0.0，那么求出他在 60天里上班迟到的期望。

**4.**战报交流：战场上不同的位置有N个战士（n>4），每个战士知道当 前的一些战况，现在需要这n个战士通过通话交流，互相传达自己知道的战况信息，每次通话，可以让通话的双方知道对方的所有情报，设计算法，使用最少的通话 次数，是的战场上的n个士兵知道所有的战况信息，不需要写程序代码，得出最少的通话次数。

**5.**有N个人，其中一个明星和n-1个群众，群众都认识明星，明星不认识任何群众，群众和群众之间的认识关系不知道，现在如果你是机器人R2T2，你每次问一个人是否认识另外一个人的代价为O(1)，试设计一种算法找出明星，并给出时间复杂度（没有复杂度不得分）。

解答： 这个问题等价于找未知序列数中的最小数，我们将reg这个函数等价为以下过程：，如果i认识j，记作i大于等于j,同样j不一定大于等于i,满足要求，i 不认识j记作i<j，对明星k,他不认识所有人，则k是其中最小的数，且满足其余的人都认识他，也就是其余的人都大于等于k.这样问题就被转换了。 就拿N=5来说，首先有数组S[5]={A,B,C,D,E}这5个变量，里边存放着随机数，求是否存在唯一最小数，如果存在位置在S中的哪里。（楼主这 里是这个意思，按我的理解题中这个最小数一定是存在且唯一的）

int finds(S,N)

{

    int flag=0;//用于判定是否有明星，即当前最小数另外出现几次

    int temp=0;//存放最小数在S中的位置

    for(i=1;i<N;i++)

   ｛

      if(!reg(S[i],S[temp])//如果temp标号的数小于i标号的数

     ｛

         temp=i;

         flag=0;//更换怀疑对象（最小数）时，标记清零

      ｝

      elseif(reg(S[temp]，S[i])//如果temp里存放的确实是唯一最小数是不会跑进这里来的

      {

           flag++;

`     }

    ｝

    if(flag>0) return -1;//表示没有明星,例如所有的数都相等

    return temp;//返回明星在S中的位置

}

**四、综合题**

有 一个淘宝商户，在某城市有n个仓库，每个仓库的储货量不同，现在要通过货物运输，将每次仓库的储货量变成一致的，n个仓库之间的运输线路围城一个圈，即 1->2->3->4->...->n->1->...，货物只能通过连接的仓库运输，设计最小的运送成本 （运货量\*路程）达到淘宝商户的要求，并写出代码。

解答：这个题目类似的题目有：

题目：[http://www.lydsy.com/JudgeOnline/problem.php?id=1045](http://www.lydsy.com/JudgeOnline/problem.php?id=1045" \t "_blank)  
有n个小朋友坐成一圈，每人有ai个糖果。每人只能给左右两人传递糖果。每人每次传  
递一个糖果代价为1，求使所有人获得均等糖果的最小代价。  
分析：  
假设a1分给an的糖果数为k，则可以得到以下的信息：  
a1 a2 　a3　　　　　　　　 an-1 　　　　　　　　　　　　 an  
当前数目：a1-k a2　　　　　　　　 a3　　　　　　　　 an-1 　　　　　　　　　　　　 an+k  
所需代价：|a1-k-ave| |a1+a2-k-2\*ave| |a1+a2+a3-k-3\*ave||a1+..+a(n-1)-k-(n-1)\*ave| |k|  
以sum[i]表示从a1加到ai减掉i\*ave的和值，这以上可以化简为  
总代价 = |s1-k|+|s2-k|+...+|s(n-1)-k|+|k|  
不难看出：当k为s1...s(n-1)中的中位数的时候，所需的代价最小

代码转载于网络：

|  |
| --- |
|  |

#include <cstring>

#include <iostream>

#include <algorithm>

using namespace std;

const int X = 1000005;

typedef long long ll;

ll sum[X],a[X];

ll n;

ll Abs(ll x){

    return max(x,-x);

}

int main(){

    //freopen("sum.in","r",stdin);

    while(cin>>n){

        ll x;

        ll tot = 0;

        for(int i=1;i<=n;i++){

            scanf("%lld",&a[i]);

            tot += a[i];

        }

        ll ave = tot/n;

        for(int i=1;i<n;i++)

            sum[i] = a[i]+sum[i-1]-ave;

        sort(sum+1,sum+n);

        ll mid = sum[n/2];

        ll ans = Abs(mid);

        for(int i=1;i<n;i++)

            ans += Abs(sum[i]-mid);

        cout<<ans<<endl;

    }

    return 0;

}