选择题

第一题，**两台电脑在局域网中，机器为千兆网卡，一台作服务器里面有一张网页为 1K 字节，问另一台下载这个网页的速度。**

我答：我不知道 1K 是指 1024 还是 1000…不过按我的算法没区别，1000 000000/8/1k

我选了 10 000 张/秒

第二题，单链表插入一个节点的问题。在 p 指向的节点后插入一个 q 指向的节点。

我答：q->next=p->next;p->next=q;

之后乱序，我记不清楚题号了。

有一题，地图染色问题，每个国家用矩形表示，让相邻国家颜色不同。离散里面有

有一题，问快速排序达到最坏情况时间复杂度 n2 的原数数组的具体情形。见数据结构

有一题，很扯的…指针取址符号混乱，选项却很白痴。

有一题，入栈序列 1,2,3,4,5,..,n，第一个出栈的是 n，问第 i 个出栈的是多少。

我答：n-i+1

最后一题，给中缀和后缀表达式，求前缀表达式。

填空题

第一题：数组(a1,a2,a3,a4..,an)，删除任意一个的概率相同，问平均删除一个要移动多少个。

我答：(n-1)/2

第二题：**一个程序填空，程序大意是在数组里面找第二大的数。**

注：不难

第三题：大致如下一个程序片段：

```
void xxx(x)
{
    intcountx=0;
    while(x)
    {
        countx++;
        x=x&(x-1);
    }
    cout<<countx<<endl;
}
```

问 xxx(9999)输出什么。

我答：8，记得做 ACM 的时候碰到过那个式子，貌似关于排列的，具体意思忘记了，搞一下可以明白是 x 变成二进制，里面有多少个 1 就是答案。

第四题：大致如下一个代码

```
inta[3][2]={1,2,3,4,5,6};
int*p[3];
p[0]=a[1];
```

问*(p[0]+1)是个什么东西

我答：4，蛮基础嗯。

简答题

第一题：7 公斤米，50 克砝码，200 克砝码各一个，称 1350 克米问最少要多少次，并编程回答。

我答，6 次，可能一开始会想到 1350/250 + 2 = 7 次，说明贪心无效。我不知道我的方法是不是很笨，用了递推，或者你可以看成是动态规划。转化一下题目的意思就是 1 克和 4 克砝码，问多少次称出 27 克大米，F[N]代表 N 克大米最少需要多少次。

则有：

F[N]=min{F[N-1],F[N-4],F[N-5]}+1

代码如下：

```
intfindmin(int weight)
{
    int v= weight/50;
    int f[150];
    f[0]=0;f[1]=1;f[2]=2;f[3]=3;f[4]=1;
    if (v<5) return f[v];
    int i;
    for (i=5;i<=v;i++)
        f[i]=min(f[i-1]+1,f[i-4]+1,f[i-5]+1);
    return f[v];
}
```

注：我一开始愣了很久，我在想，称好的大米可以作为砝码来用吗？？这样就是另一种问题了吧。

附加：

如果天平能做为平衡工具的话，两次平分到 1750 克，然后两次量出 200 克，1750-400 就是 1350 克了。。。四次。。。。

解答题第一题：

第一次：200+50，称出 250g 第二次：200+250，称出 450 第三次：200+450，称出 650

共称出 1350g

第二题，**有 N 个蛋和 M 个篮子，把蛋放到 M 个篮子里，每个篮子都不能为空。另外，需要满足：任意一个小于 N 的正整数，都能由某几个篮子内蛋的数量相加的和得到。写出程序，使得输入一个(N,M)，输出所有可能的分配情况。** 我答：不能想出算出所有摆放方法的方法，期待 ACM 大牛路过。

（1. 先取 M 个蛋放入 M 个篮子（一个篮子一个蛋）

2.剩下的（N-M）个蛋按照 1，2，4，。。方式依次维持各个篮子中蛋的数量（要有一个篮子保持只有一个蛋），若最后的蛋不是 2 的方次，有多少放入一个篮子

3.取 L(L<=N)个蛋时，应按二进制编码值考虑，如 13 个蛋：13 的二进制码值是 1101，则取有 8 个、4 个和 1 个蛋的篮子即可。

另外：题目不完整，N 与 M 应该有数量关系：M<=N 且 N<2 的 M 次方）

解答 1

view plaincopy to clipboardprint?

```
/**
 * 假设 n>m 并且 n 小于 100
public class Test {
	private int m;
	private int n;
	private int eggs[];
	private int numAnswer;

	Test(){
		m=10;
		n=20;
		numAnswer=0;
		eggs =	new int[m];
		for(int i=0;i<m;i++){
			eggs[i]=0;
		}
	}
	private void fill(boolean [] state, int step, int sum){
		if(step>=m){
			state[sum] = true;
			return ;
		}
		fill(state,step+1,sum);
		fill(state,step+1,sum+eggs[step]);
	}

	/**
	 * 判断是否满足：任意一个小于 N 的正整数，都能由某几个篮子内蛋的数量相加的和得到
	 * 算法：暴力枚举所有篮子的组合
	 * @return
	 */
	private boolean judge(){
		boolean [] state = new boolean [n+1];
		for(int i=0;i<=n;i++){
			state[i] = false;
		}

		fill(state,0,0);

		for(int i=1;i<=n;i++){
			if(!state[i]){
```

```
                return false;
            }
        }
        return true;
    }

    /**
     * 给每个篮子分鸡蛋，升序（后一个篮子的鸡蛋必须不小于前一个篮子，避免重复计算）
     * @param pre  前一个篮子鸡蛋数
     * @param already  前 step 个篮子  已使用的鸡蛋数
     * @param step  第 step 个篮子
     */
    public void solve(int pre,int already, int step){
        if(step==m-1){
            //最后一个篮子
            eggs[m-1]=n-already;
            //不符合条件
            if(eggs[m-1]<pre)       return;

            //判断是否满足：任意一个小于 N 的正整数，都能由某几个篮子内蛋的数量相加的和得到
            if(judge()) {
                for(int i=0;i<m;i++){
                    System.out.print(eggs[i]+" ");
                }
                System.out.println();
                numAnswer++;
            }
            return ;
        }

        // 给第 step 个篮子装鸡蛋，pre  到  n-already  种可能
        for(int i=pre; i<=n-already; i++){
            eggs[step]=i;
            //递归
            solve(i,already+i,step+1);
        }
    }
    public static void main(String arg []    ){
        Test test = new Test();
        test.solve(1,0,0);
        System.out.println("可能情况的数量："+test.numAnswer);
    }
```

```
}
```

解答 2

```
using System;
using System.Collections.Generic;
using System.Text;
namespace CmpSplitEgg
{
   class Program
   {
   static void Main(string[] args)
   {
   SplitEgg();
   }

   public static bool SplitEgg()
   {
   // 初始化变量，差额 diffNum = 鸡蛋数 eggNum - 篮子数 basketNum
   int eggNum = 0, basketNum = 0, diffNum;
   // 输入鸡蛋数、篮子数
   Input(ref eggNum, ref basketNum);
   // 排列结果，并初始化
   int[] resultEggs = new int[basketNum];
   for(int i=0;i<basketNum;i++)
   {
   resultEggs[i] = 1;
   }

   // 差额 = 鸡蛋数 - 篮子数
   diffNum = eggNum - basketNum;
   if (diffNum < 0)
   {
   Console.WriteLine("You can't make N < M");
   return DoAgain() && SplitEgg();
   }
   else if (Math.Pow(2, basketNum) <= eggNum)
   {
   Console.WriteLine("You can't make N > 2^M");
   return DoAgain() && SplitEgg();
   }

   // 对任意一个小于 N 的数 总能使几个篮子里的鸡蛋总数等于它，则需要编号 n 的篮子
```

```
放的鸡蛋数<=前面的鸡蛋数总和+1
    // 基于 2 进制编码是能表示所有数字且位数最小的编码方式，上面条件由此推出
    // 假设组合为升序排列，第一个篮子必然为 1 个鸡蛋
    RandomLay(resultEggs, 1, eggNum);

    // 是否重新做一次
    return DoAgain() && SplitEgg();
    }

    /// <summary>
    /// 重新选择
    /// </summary>
    public static bool DoAgain()
    {
    Console.WriteLine();
    Console.WriteLine("if you want to enter the N and M again?Yes（Note: if not enter
'Y' or 'y', the application will quit...）");
    string choice = Console.ReadLine();
    return choice.ToLower() == "y";
    }

    /// <summary>
    /// 输入
    /// </summary>
    /// <param name="eggNum">鸡蛋数量</param>
    /// <param name="basketNum">篮子数量</param>
    public static void Input(ref int eggNum, ref int basketNum)
    {
    while (true)
    {
    try
    {
    Console.WriteLine("Please enter the egg number N:");
    eggNum = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine("Please enter the basket number M:");
    basketNum = Convert.ToInt32(Console.ReadLine());
    break;
    }
    catch (Exception)
    {
    Console.WriteLine("Enter error: please input integer!");
    Console.WriteLine();
    continue;
    }
```

```
}
}

/// <summary>
/// 随即放置鸡蛋
/// </summary>
/// <param name="result">结果</param>
/// <param name="beginIndex">开始索引</param>
/// <param name="total">鸡蛋数</param>
public static void RandomLay(int[] result, int index, int total)
{
// iMax 为 index 对应可取鸡蛋数上限
int iMax = 1, basketNum = result.Length;
for (int j = 0; j < index; j++)
{
iMax += result[j];
}

// 复制
int[] copyResult = new int[basketNum];
for (int i = 0; i < basketNum; i++)
{
copyResult[i] = result[i];
}

// 结束条件 1：已为最后一个篮子
if (index == basketNum - 1)
{
int mBasket = total - iMax + 1;
if (mBasket <= iMax)
{
copyResult[index] = mBasket;
Console.Write("Split solution: ");
foreach (int res in copyResult)
Console.Write(res + " ");
Console.WriteLine();
}
return;
}
for (int ii = copyResult[index - 1]; ii <= iMax; ii++)
{
// 结束条件 2：当前至少需要鸡蛋数
int nowNum = ii * (basketNum - index) + iMax - 1;
// 表示无法再按升序放置鸡蛋
```

```
    if (nowNum > total)
    break;
    copyResult[index] = ii;
    RandomLay(copyResult, index + 1, total);
    }
    }
    }
}
```

解答 3

[code=C/C++][/code]

```
#include<iostream>
#include<math.h>
#include<malloc.h>
#include<fstream>
using namespace std;

struct solution
{
int *ptr;
struct solution *next;
};
typedef struct solution solu;

int* first(int n,int m); //计算出第一种组合
solu* others(int n,int m,solu *head,solu *prior); //计算出其他组合
int sum(int n,int *p); //计算前 n-1 个篮子里的蛋数和
bool only(solu *head,int *p,int m); //检查组和是否满足要求
int ways; //全局变量，保存组合的方法数

void main()
{
int n=0,m=0,i=0,k=0;
solu *head=NULL;
solu *temp=NULL;
LABLE: cout<<"输入鸡蛋数 N=";
cin>>n;
cout<<"输入篮子数 M=";
cin>>m;
if(m<=0||n<=0||m>n||(double)n>=pow(2.0,m)) //对 m,n 的约束
{
cout<<"输入不合法！"<<endl;
goto LABLE;
}
cout<<"正在计算..."<<endl;
```

```
head=others(n,m,head,NULL); //调用 others 开始计算
temp=head;
ofstream file("D:\\egg.txt"); //结果保存着这个目录下
cout<<"共有"<<ways<<"种组合方式："<<endl;
file<<"共有"<<ways<<"种组合方式："<<endl;
k=ways;
while(temp!=NULL&&ways)
{
cout<<"方式"<<k-ways+1<<"："<<endl;
file<<"方式"<<k-ways+1<<"："<<endl;
for(i=0;i<m;i++)
{
cout<<*(temp->ptr+i)<<" ";
file<<*(temp->ptr+i)<<" ";
}
delete[] temp->ptr;
temp=temp->next;
cout<<endl;
file<<endl;
ways--;
}
file.close();
cout<<"操作结果保存在 D://egg.txt，您可以查看或删除之。";
cin>>i;
}

int sum(int n,int *p) //计算前 n-1 个篮子里的总蛋数
{
int total=0;
for(int i=0;i<n;i++)
{
total+=*(p+i);
}
return total;
}
int* first(int n,int m)
{
int *p,i=0,temp1=0,temp2=0;
p=(int *)malloc(m*sizeof(int));
for(i=0;i<m;i++) //每个篮子里放一个蛋
{
*(p+i)=1;
}
//下面的分配满足的条件：
```

```
//“总能找到几个篮子，使里面鸡蛋的和等于任意一个小于 n 的正整数”
//下面的 if～else 语句完成一种组合，升序排列，并使后面的篮里的蛋尽量多
if(n-m>m-1)
//剩下的蛋数大于前面 m-1 个篮子里的蛋数和，
{
*(p+m-1)+=m-1;
while(sum(m,p)<n) //还有蛋剩余
{
temp1=n-sum(m,p); //剩蛋数
for(i=m;i>0;)
{
temp2=sum(i-1,p); //第 i 个篮子前面的所有篮子里的蛋数的总和
if(*(p+i-1)<=temp2)
//第 i 个篮子里的蛋数小于等于前面篮子里蛋的总数，给这个篮里加蛋
//否则，见 else
{
if(temp1<=temp2-*(p+i-1)+1) //剩下的蛋可以全部放到第 i 个篮里，完毕
{*(p+i-1)+=temp1;break;}
else {*(p+i-1)+=temp2-*(p+i-1)+1;break;} //在第 i 个篮子放可能达到的最多蛋数
}
else i--; //检测前面那个篮子
}
}

}
else *(p+m-1)+=n-m;
//剩下的蛋数小于等于前面 m-1 个篮子里的蛋数和，
//把所有的蛋都放到最后一个篮里，完成一种组合。
return p;

}

solu* others(int n,int m,solu* head,solu *prior)
{
int i=0,j=0,k=0;
if(head==NULL) //还没有任何组合
{
solu *s=new(solu);
s->ptr=first(n,m); //调用 first()生成满足后面的值最大的升序序列
head=s;
head->next=NULL;
prior=head;
ways=1;
}
```

```
for(j=m-1;j>0;j--) //两重循环，开始计算其他组合
//原理是从后面的篮子里取出鸡蛋放入前面的篮子中
{
if(*(prior->ptr+j)==1) //后面的篮子里蛋数为1，跳出循环
break;
for(i=j-1;i>0;i--) //一个个往前挨
{
if(*(prior->ptr+j)-1>*(prior->ptr+i)) //后面的篮子减掉后不能比前面的少，保持升序排列
{
int *p=(int *)malloc(m*sizeof(int));
for(k=0;k<m;k++)
{
(*(p+k))=(*(prior->ptr+k));
}
(*(p+j))--;(*(p+i))++;
if(only(head,p,m)) //检查是否满足条件，满足则将结果添加到链表中
{
solu *stemp=new(solu);
stemp->ptr=p;
stemp->next=head->next;head->next=stemp;
head=others(n,m,head,stemp);
ways++;
}
else delete[] p;

}
else if(*(prior->ptr+j)-1==*(prior->ptr+i))
continue;
else
break;
}
}
return head;
}
bool only(solu *head,int *p,int m) //判断条件是否符合
{
solu *s=head;
int flag=0,i=0;
for(int k=0;k<m-1;k++)
{
if(*(p+k+1)<*(p+k)||*(p+k+1)>sum(k+1,p)+1) //两个条件：1、升序，2、后面的数必须小于等于前面的篮子总数和加1
return false;
```

```
}
while(s!=NULL) //判断是否有过相同的组合，有则返回 false
{
flag=0;
for(i=0;i<m;i++)
{
if(*(s->ptr+i)!=*(p+i))
{
flag=1;
break;
}
}
if(!flag)
{
return false;
}
s=s->next;
}
return true; //检查通过，返回 true
}
```

任意给定的 M 和 N, 假定鸡蛋已经全部按规定放好了，那么如果能取出 X 个鸡蛋（0<X< M）那么剩下的就是 M-X 个鸡蛋，也相当于取出了 M-X 个鸡蛋。所以只需要考虑能够取到 1 到 M/2 个鸡蛋即可。

又如有的哥们讲的 1,2,4,8.。。2^(k-1)这样数列比较特殊，有 k 位这样的数列,可从中取出若干个数相加得到任意小于 2^k-1 的数（因为 K 位这样的数列相加的和为 2^k - 1）,那么依照题意我们应该从这样的数列开始考虑。

现在有 N 个篮子，先拿掉一个篮子。那么这 N-1 个篮子按上面的方法放鸡蛋的话可表示出所有小于 2^(N-1)-1 的数，如果 2^(N-1)-1 > M/2 那么该题有解，否则无解。

情况一 M > 2^(N-1)-1 > M/2

给 N-1 个篮子中分别放 1,2,4,8....2^(N-2)个鸡蛋,剩下的鸡蛋全部放最后一个篮子里。

由于前 N-1 个篮子可表示任意小于 M/2 的数，所以这 N 个篮子可表示所有小于 M 的数。如果不考虑篮子的编号和顺序，此情况只有一种放法。

情况二 2^(N-1)-1 > M

按上面的方法还没放到第 N-1 个篮子鸡蛋就没了，那么这时的做法是，先按上面的方法放好所有鸡蛋。假设放到第 x 个篮子鸡蛋就没了，那么从 x+1 个篮子开始回头从 x 篮子里拿一个鸡蛋放入其中，然后是 x+2 篮子同样的处理，依次类推。如果 x 篮子中只剩一个鸡蛋了还有篮子是空的，那么从 x-1 篮子中取鸡蛋。依次类推放满所有篮子。按这种方法如果 N=M，则每个篮子放一个鸡蛋。如果不考虑篮子的编号和顺序则方法是很有限的，也就是将鸡蛋在几个篮子里倒来倒去。

可以看出此算法的复杂度只有 N，根本不需要递归什么的。老太太按这种方法操作也能很快得出一个解。

也可以看出给定若干个鸡蛋，至少需要多少个篮子才能满足题目要求，比如 100 个鸡蛋就最少需要 7 个篮子 500 个鸡蛋最少需要 9 个篮子，1000 个鸡蛋最少需要 10 个篮子

第三题，大意淘宝网的评论系统，原先只有一个评论表，对于现在大用户，大数据量，大访问量，请设计一个合理可行的架构来优化关于评论的数据库。

我答：哥蒙了，哥胡言乱语的。

附加题：前端设计师必答

第一题：图片默认为半透明，鼠标移上去变成不透明。

我注：img 标签 onfocus 和 onblur 的应用，注意这个透明的属性在 IE 和 FireFox 下是不同的。而且用 js 控制的时候，属性名也要注意…

第二题：一个输入框，和一个列表框，列表框里面有很多字符串，在输入框里面输入字符串时，列表框中字符串前缀是该字符串的做高亮或者其他显著表示。最后回车选择或者鼠标双击列表框选择。

我注：看上去要写不少东西啊……实在懒了。

总结：

基础偏多，大题很算法，很偏实际应用，前面不会不应该了，后面看造化，毕竟时间也不多。

最后：如果有错，请指正，仅给路人或未来想进淘宝的孩子或八卦的朋友做些参考。

淘宝 2011 校招南京东大宣讲笔试经验

王思璇 2010-11-10 17:58:22

淘宝 2011 校招南京东大宣讲笔试经验

昨天参加淘宝的宣讲会，参加笔试。

东大，六点半开始宣讲会，去得有点早，顺便熟悉了一下东大。

开始宣讲会，宣讲人是淘宝的七公。

淘宝是一家不错的公司，年轻活力，疯狂——或者用宣讲会视频中的学长所说的是全力以赴，

而不是尽力而为。
年轻，朝气，给年轻人舞台。
宣讲会中，淘宝好像特注重 C++和 Java 工程师的招聘，居然在宣讲会的职位较少 PPT 中没有系统运维的介绍，让人很是揪心。
笔试，分为几种试卷。C++和 Java，系统运维、安全、前端设计是在一份试卷上。PHP、数据库是在另外的试卷上。
感觉试卷分配的不是很合理，应该把系统运维和 PHP 数据库放在一起好一点。淘宝也许有淘宝的想法吧。
考了系统运维，试卷结构是基础题 40 分、方向性试题 40 分、综合题 20 分。
感觉前面的基础题像是杂烩，包括网络题、编码基础、IP 主机子网划分、线性结构、还有一个 C 的运算题吧。
前面感觉一般般，填空题的第五题 C 的执行，压根没做过，苦了我这个非计算机专业，又特想钻研网络的人了。
下面是方向性试题了，毫无疑问，挥起豪笔（淘宝送的宣传笔是圆珠笔，用自己的考试笔舒服一点），直奔系统运维。
两题，前面一题是一个 A 类网络的划分，好像是 10.13.16.0/20，应该是这个，记不清了，反正是一个 20 位的 submask，找出网号段，IP 段，子网掩码。。。这些网络划分的知识，比较简单了，应该拿下了。。。
下面一题，是对于搞安全的人是常识啦，举例 4 种 DNS 常见攻击方法。我的答案是 DNS 缓存中毒，DDOS 攻击，ARP 欺骗，DNS 重定向，域名攻击。
反正方向性试题考的是方向基础吧，不是很难，重在基础。拿到题目时，还愣了一下，IP 划分，好久没动过了，但想想全是基础，呵呵，easy，easy。
第三部分，是综合题了，但我觉得更像是为 C++和 Java 的人准备的，一道算法和数据处理的题目。技术类除前端的不用做，其他的都要做。
无奈，作为经管院的我说这题不会。
说说做前端的吧，题目没做，但看了一下，掌握好 HTML 语言，CSS，应该问题不大，而且，题目就是现在比较流行的图片缩放功能。购物网站上经常看到的鼠标放到
宝贝上，Popup 出一个图片层，这些代码，平时留意一下，应该没问题。
总体上，淘宝是一家挺人性化的公司，希望淘宝做大，实现大淘宝战略。葛优的话：“21 世纪——人才。。。”

2011 淘宝.net 面试之马后炮

王思璇　2010-11-10 17:55:27
首先，机会都是大家争取的，努力是自己奋斗的，
谋事在人，成事在天，努力过就好。给自己的安慰，呵呵。
今天去淘宝面试，在等待教室，看着宣传片，看见一个个硕士哥哥都好轻松，哈哈。有个在玩 ipad。一个个都有书看看的。
到了时间点，面试官叫我名字，然后跟着一个大哥哥一起进了个房间，面试官很和蔼，都是

带笑容的，感觉让人很舒服，面试官说我们聊聊吧，好惬意的。反正给人的感觉就是很好。
刚开始大哥哥都是对着简历问问，下面罗列分析下
第一：关于 google 的 HK 站的访问不稳定，你是怎么解决的，怎么上国际站的。
当时我蒙了，还真没去关注，我一般都是 HK 站不稳定，我就等等，然后可以用了，我就继续用的，还真没想到怎么上 COM 的站
呵呵，回来看了下，原来下面可以点 in english。就可以直接上国际站了。
分析：这点应该是考察你对一样事物的，刨根的能力，看你这个人是不是有专研的能力和思想，个人愚见。
第二：数据库索引的状态
第三：TCP/IP
第四：设计模式
你最擅长，实践，那几个原则
第五：学习的方法，经常上的一些网站
这个今天我就说。博客园和 CSDN。。。。忘记把 codeplex 说了。毕竟这个也是非常好的，好多开源组件。哎真是可惜
第六：最擅长的技术，
这个今天我没有表达明确，直接来个 webform 就好了。。还说了很多废话。现在想想真是不应该的哈。
分析：这个大家要小心的，千万不要说不了解的，要不然会死的很惨，呵呵
第七：数据库范式的理解，1,2,3
第八：软件工程，开发模式
瀑布、敏捷，，必要了解的，瀑布的过程，CMM5 层是哪五层，你具体的理解是什么。
第九：oo 思想的薄弱，（我目前的状态）
这个还是很薄弱的，一些思想类的书都没看，都是工具书。
面试官的感觉，有可能就是我这方面做的太少了。还是初级程序员做的，那种页面级别的工作。
这个在以后还是要提升的。
第十：string=a+b 几个对象
3 个
第十一：datatable 里面的 row 有几种状态
这个真没关注过，
第十二：开发用什么框架，有没有用其他的组件

实习笔试题
在进入我的淘宝页面时，此页面需要获取登录用户的相关信息，在访问量少的情况下，可以采用直接访问数据库的方式，但当访问量太高时，会导致数据库压力过高，因此通常采取的方法为将用户信息进行缓存，在用户数不多的情况下，这个方案还是提供了很多的帮助的，但用户数增多了一点后，出现的问题是缓存占了太多的内存，而经分析，原因是这些缓存中很多是不访问的用户信息。
1.1 请写一端存储用户信息的缓存实现代码，并实现当缓存到达一定大小后，如继续新增用户信息，则将最近不访问的用户信息从缓存中踢出；
1.2 由于我的淘宝是部署在多台机器上的，如用户每次访问不同的机器，以上方案会造成每

台机器都需要去数据库中加载此用户信息，请给出一个方案来避免此问题。

4 月 12 日，淘宝来玉泉曹西招实习生。首先介绍了下淘宝的发展和使用的技术，然后就开始进入正题了。

笔试，分基础部分和分类部分

基础部分：雷打不动的数据结构和算法，选择题都是一些树、排序之类的题，都忘了，T_T。大题有三
写一个 mergeSort。递归和 merge，基础，总算还记得…
找出数组中的所有和为某个值的所有组合。不会，T_T
谈谈对最近 IT 业的发展和自己的想法。说了下 4 月编程语言排行榜的变化和 Sun 被收购一系列的事情
分类部分：分 Java、C++、测试和系统，只做了 Java，挺简单的。就一个线程安全的 Collection 类不知道，原来是 Vector..
一直等了 2 个礼拜才等来了面试通知。今天晚上到淘宝 8 楼，2 个男的。先自我介绍。大部分的时间就讲了在 Trilogy 实习的事，把每个项目都很仔细的讲了，然后面试官针对某些细节提些问题。还有就问了 Java Collection 的一些类的问题，无非 HashMap、TreeMap、LinkedHashMap 的各自特点和异同，再讲讲自己对底层实现的钻研，我讲了一下 ArrayList 的继承结构，说具体源码实现还没看。面试结束，就说结果一周内会通知我

Java 和 c 语言的区别
java 是面向对象的语言，所以里面提供了很多的类，是目前最强大的开发语言，C 语言是高级语言中相对来讲比较低级的语言，是面向过程的语言，因为 C 语言执行速度快，执行效率高，能够对底层硬件进行操作，所以一般用来开发后台程序，而 java 平台移植性好，所以一般用来开发网络应用程序，照现在的情况来看，java 最有发展前途，是目前使用最多的开发语言

JAVA 与 C 语言的区别之我见
1. 语言背景：
C 语言是在单机时代应用非常广泛， c 语言是基于汇编语言和高级语言间的一种中级语言，它融合了高级语言的简单易用和汇编语言的执行效率。而 Java 是在研究电子消费产品开发平台和互联网应用的基础上实现的，它的许多语言特性也是从 c 语言那里沿用和发展，并且使面向对象更加自然和完善（如安全性和代码的移动性）。
2.语言跨平台：
C 语言不可以跨平台，JAVA 是不怕这一点的，因为 Java 可以跨平台，在 windows 和 unix 等系统上都可以很好的运行，这一点 Java 是很强大的。
3.指针：
指针是 c 语言最大的优点，它可以使用户几乎可以访问计算机的所有内存资源和其他部分资源（就是指那里打那里）。同时也是 c 语言程序最难掌握和调试的问题，并且给系统的安全性和稳定性带来很大的困难。 而 java 中没有指针的概念，尽管也有数组和对象的引用的概

念，但它的管理全部交给系统管理，这样限制了用户的资源的访问，但是也给 java 系统带来安全性和稳定性。JAVA 语言让编程者无法找到指针来直接访问内存无指针，并且增添了自动的内存管理功能，从而有效地防止了 c 语言中指针操作失误，如野指针所造成的系统崩溃。但也不是说 JAVA 没有指针，虚拟机内部还是使用了指针，只是外人不得使用而已。这有利于 Java 程序的安全

4.封装

在 java 中引入了 package 的概念，使面向对象和面向组件开发更加方便，而在 c 语言中没有 package 概念，需要其他方式来实现。Java 都能够实现面向对象思想（封装，继乘，多态）。而由于 c 语言为了照顾大量的 C 语言使用者，而兼容了 C，使得自身仅仅成为了带类的 C 语言，多多少少影响了其面向对象的彻底性！JAVA 则是完全的面向对象语言，它句法更清晰，规模更小，更易学。它是在对多种程序设计语言进行了深入细致研究的基础上，据弃了其他语言的不足之处，从根本上解决了 c 语言的固有缺陷。

5．数据类型及类

Java 是完全面向对象的语言，所有函数和变量部必须是类的一部分。除了基本数据类型之外，其余的都作为类对象，包括数组。对象将数据和方法结合起来，把它们封装在类中，这样每个对象都可实现自己的特点和行为。而 c 语言允许将函数和变量定义为全局的。

6．自动内存管理

Java 程序中所有的对象都是用 new 操作符建立在内存堆栈上， Java 自动进行无需内存回收操作，不需要程序员进行删除。而 c 语言中必须由程序贝释放内存资源，增加了程序设计者的负扔。Java 中当一个对象不被再用到时，无用内存回收器将给它加上标签以示删除。JAVA 里无用内存回收程序是以线程方式在后台运行的，利用空闲时间工作。

7. 字符串：

C 语言中声明字符串是用 char 声明的数组，而 JAVA 则调用 String 方法直接就可以声明一个字符串，很简便，这是 C 语言的不足之处！

Java 没有函数，作为一个比 c 语言更纯的面向对象的语言，Java 强迫开发人员把所有例行程序包括在类中，事实上，用方法实现例行程序可激励开发人员更好地组织编码。