



适应于请求段的内存分配方法是（ ）。

正确答案: D 你的答案: 空 (错误)

首次适应和最佳适应

固定分区和可变分区

首次适应和固定分区

最佳适应和可变分区

在虚拟存储系统中，若进程在内存中占三块(开始时为空，采用先进先出页面淘汰算法，当执行访问页号序列为 1、2、3、4、1、2、5、1、2、3、4、5、6 时，将产生（ ）次缺页中断。

正确答案: D 你的答案: 空 (错误)

7

8

9

10

操作系统采用缓冲技术,通过减少对 CPU 的（ ）次数,提高资源的利用率。

正确答案: A 你的答案: 空 (错误)

中断

访问

控制

依赖

下列选项中关于多级队列调度和多级反馈队列的调度的叙述中，正确的是（ ）。

正确答案: A 你的答案: 空 (错误)

多级反馈队列的调度中就绪队列的设置不是像多级队列调度一样按作业性质划分，而是按时间片的大小划分

堆积队列调度用到优先权，而多级反馈队列调度中没有用到优先权

多级队列调度中的进程固定在某一个队列中，而多级反馈队列调度中的进程不固定

堆积队列调度中每个队列按作业性质不同而采用不同的调度算法，而多级反馈队列调度中除了个别队列外，均采用相同的调度算法

下列的进程状态变化中，哪些是不可能发生的？

正确答案: C 你的答案: 空 (错误)

运行→就绪

运行→等待

等待→运行

等待→就绪





路由器收到一个数据包，数据包的目标地址是 202.65.17.4，该子网属于哪一个网段？

正确答案: B 你的答案: 空 (错误)

202.65.4.0/20

202.65.16.0/22

202.65.20.0/21

无正确选项

在牛网技术有限公司的局域网中,牛妹的私人电脑可以查看到同事的电脑,也成功登陆了 QQ,但无法访问到公司的站点"http://www.nowcoder.com/",请你协助牛妹查找 最有可能出现的问题的地方是？

正确答案: C 你的答案: 空 (错误)

UDP

DHCP

DNS

HTTP

浏览器

将一个 C 类网络划分为 3 个子网，每个子网最少要容纳 55 台主机，使用的子网掩码是？

正确答案: D 你的答案: 空 (错误)

255.255.255.252

255.255.255.248

255.255.255.224

255.255.255.192

在网络 7 层协议中，如果想使用 UDP 协议达到 TCP 协议的效果,可以在哪层做文章？

正确答案: C 你的答案: 空 (错误)

应用层

表示层

会话层

传输层

网络层

牛客网由于访问客户量的增长,原来的服务器不足以维持请求,经常发生宕机的突发情况,因此为了解决这个问题,CEO 决定新增加几台服务器,那么问题是这些接入 WEB 服务器第一次被访问到时，不同协议的发生顺序是下面中的_____。

正确答案: A 你的答案: 空 (错误)

ARP -> DNS -> HTTP

ARP -> HTTP -> DNS

DNS -> HTTP -> ARP

DNS -> ARP -> HTTP





HTTP -> ARP -> DNS

HTTP -> DNS -> ARP

叶某开发出了一款超级智能机器人，能对任何提问给出“是”或者“不是”的答案。现有 3 个这种机器人，其中有数量不定的(0 到 3 个)机器人发生了故障。如果正常机器人总是给出正确的答案，而故障机器人总是给出错误的回答。每一回合只能问任意一个机器人任意一个问题，那么至少需要_____回合才能确保区分出哪些机器人是正常，哪些是故障。

正确答案: C 你的答案: 空 (错误)

- 1
- 2
- 3
- 4
- 5
- 6

人患癌症的概率为 1/1000.假设有一台癌症诊断仪 S1，通过对它以往的诊断记录的分析，如果患者确实患有癌症它的确诊率为 90%,如果患者没有癌症，被诊断成癌症的概率是 10%。某人在被诊断为癌症后，他真正患癌症的概率为()

正确答案: C 你的答案: 空 (错误)

- 9/1000
- 1/1000
- 1/112
- 9/10

Linux 文件权限一共 10 位长度，分成四段，第三段表示的内容是（）。

正确答案: C 你的答案: 空 (错误)

- 文件类型
- 文件所有者的权限
- 文件所有者所在组的权限
- 其他用户的权限

在局域网络内的某台主机用 ping 命令测试网络连接时发现网络内部的主机都可以连同，而不能与公网连通，问题可能是（）。

正确答案: C 你的答案: 空 (错误)

- 主机 IP 设置有误
- 没有设置连接局域网的网关
- 局域网的网关或主机的网关设置有误
- 局域网 DNS 服务器设置有误

在 Linux 系统中，哪个文件你可以存储用于创建用户目录的系统用户默认文件？





正确答案: C 你的答案: 空 (错误)

```
/usr/tmp
/etc/default
/etc/skel
/etc/users
```

在 linux 编程中，以下哪个 TCP 的套接字选项与 nagle 算法的开启和关闭有关？

正确答案: B 你的答案: 空 (错误)

```
TCP_MAXSEG
TCP_NODELAY
TCP_SYNCNT
TCP_KEEPALIVE
```

哪个指令是将所有缓冲区的内容写入到磁盘？

正确答案: B 你的答案: 空 (错误)

```
save
sync
edbuff
flush
```

对一个含有 20 个元素的有序数组做二分查找，数组起始下标为 1，则查找 A[2] 的比较序列的下标为 ()

正确答案: B 你的答案: 空 (错误)

```
9, 5, 4, 2
10, 5, 3, 2
9, 6, 2
20, 10, 5, 3, 2
```

对数组 $A[] = \{4, 78, 3, 64, 32, 89, 43, 12\}$ 进行 Hash 存储时，选用 $H(K) = K \% 7$ 作为 Hash 函数，则 Hash 地址为 1 的元素有 () 个。

正确答案: C 你的答案: 空 (错误)

```
1
2
3
4
```

一棵 Huffman 树有 m 个叶结点，使用 `struct Node{Node *l,*r;int val;}` 结构来存储该树中的结点，一共会产生多少个 NULL 指针？ ()

正确答案: D 你的答案: 空 (错误)



$2m+1$

$4m$

$2m-1$

$2m$

今天的计算机课上，老师给同学们出了一道题：

输入 n 个数，请将其中的偶数的二进制反转。

eg: 输入 1 6 5

其中 6 是偶数，二进制表示为 110，反转后为 011，代表 3，所以最终输出 1 3 5.

小贱君最近脑子不怎么好使，想了半天也没想出来如何做，最后他向你寻求帮助了，帮帮可怜的小贱君吧！

```
1  #include <iostream>
2  using namespace std;
3  int flipNum(int x){
4      if (x & 1) return x;
5      int n = 0;
6      while (x){
7          n = (n << 1) | (x & 1);
8          x = x >> 1;
9      }
10     return n;
11 }
12 int main(){
13     int N;
14     int x;
15     while (cin >> N){
16         for (int i = 0; i < N; ++i){
17             cin >> x;
18             cout << flipNum(x);
19             if (i != N - 1) cout << " ";
20         }
21         cout << endl;
22     }
23 }
```

经过深思熟虑之后，小贱君打算去 **M** 国闯一闯，那是一个古老的东方国度，传说有很多高阶魔法师，他想成为一名伟大的魔法师，将来征服星辰大海。

经过千辛万苦，小贱君终于来到了 **M** 国，不幸的是刚进城门小贱君就被 **M** 国的守城士兵困在了一种叫做“困兽之斗”的阵法之中。

士兵对小贱君说：“看到漂浮在你身边的宝石了吗？彩虹连接的两颗宝石可以任意交换位置，你需要通过一系列交换后使得宝石组成的字符串的字典序最小。若不能破阵，那还是请回吧！”

小贱君观察了一下周围的宝石，只见每颗宝石上标有一个小写字母，而且有一些宝石上通过彩虹与其他宝石相连。

琢磨了半天，他终于搞懂了这个阵法的意思：

获取更多资料礼包！

微信关注：



若宝石系列为: dcba

其中有两道彩虹, 分别是(0,1),(1,2),代表第一个位置上的宝石可以和第二个位置上的宝石互换, 第二个位置上的宝石可以和第三个位置上的宝石互换, 最终可以得到字典序最小的宝石系列: bcda。

作为小贱君的死党, 你有什么方法帮助他破阵吗?

```
1  #include <iostream>
2  #include <vector>
3  #include <string>
4  #include <algorithm>
5
6  using namespace std;
7
8  int find(vector<int> &, int);
9  void merge(vector<int> &, int, int);
10
11 int main() {
12     string s;
13     while (cin >> s) {
14         int len = s.length();
15         vector<int> indexs(len, -1);
16         int cnt;
17         cin >> cnt;
18         for (int i = 0; i < cnt; ++i) {
19             int from, to;
20             cin >> from >> to;
21             int froot = find(indexs, from);
22             int troot = find(indexs, to);
23             if (froot != troot) {
24                 merge(indexs, froot, troot);
25             }
26         }
27         vector<vector<int>> mat(len);
28         for (int i = 0; i < indexs.size(); ++i) {
29             if (indexs[i] == -1) continue;
30             int root = find(indexs, i);
31             mat[root].push_back(i);
32         }
33         for (int i = 0; i < mat.size(); ++i) {
34             vector<char> chs;
35             for (int j = 0; j < mat[i].size(); ++j) {
36                 chs.push_back(s[mat[i][j]]);
37             }
38             sort(chs.begin(), chs.end());
```



```
39             for (int j = 0; j < mat[i].size(); ++j) {
40                 s[mat[i][j]] = chs[j];
41             }
42         }
43         cout << s << endl;
44     }
45     return 0;
46 }
47
48 int find(vector<int> &indexs, int i) {
49     int ret = i;
50     while (indexs[ret] >= 0) {
51         ret = indexs[ret];
52     }
53     int q;
54     for (int p = i; p != ret; p = q) {
55         q = indexs[p];
56         indexs[p] = ret;
57     }
58     return ret;
59 }
60
61 void merge(vector<int> &indexs, int i, int j) {
62     int t;
63     if (indexs[i] < indexs[j]) {
64         indexs[i] += indexs[j];
65         indexs[j] = i;
66     } else {
67         indexs[j] += indexs[i];
68         indexs[i] = j;
69     }
70 }
```

这是一个并查集问题，收集所有通过采用连通的节点，将这些节点按照字典序排序。

经过多次强攻之后，赫柏带领的军团不仅没能击败鲁卡斯，反而被鲁卡斯打得七零八落，赫柏终于体会到了高阶天之驱逐者的强大实力。

不过，赫柏最终还是找到了鲁卡斯的致命弱点，他发现鲁卡斯喜欢收集上古卷轴，因为上古卷轴能够让鲁卡斯获得神秘之力。

卢卡斯决定使用上古卷轴将卢卡斯引诱到绝域之门，利用绝域之门的力量消灭卢卡斯。

赫柏注意到卢卡斯喜欢收集不同的卷轴，如果总是捡到相同的上古卷轴，它的兴趣就会逐渐降低。

赫柏现在拥有 N 种不同的卷轴，每种卷轴有 A_i 个。现在他要将这 N 个卷轴分散在鲁卡斯领地到绝域之路的路上，每一种排列方式都有一个吸引值 **Charm**，吸引值越高，鲁卡斯被引诱到绝域之门的概率越高。

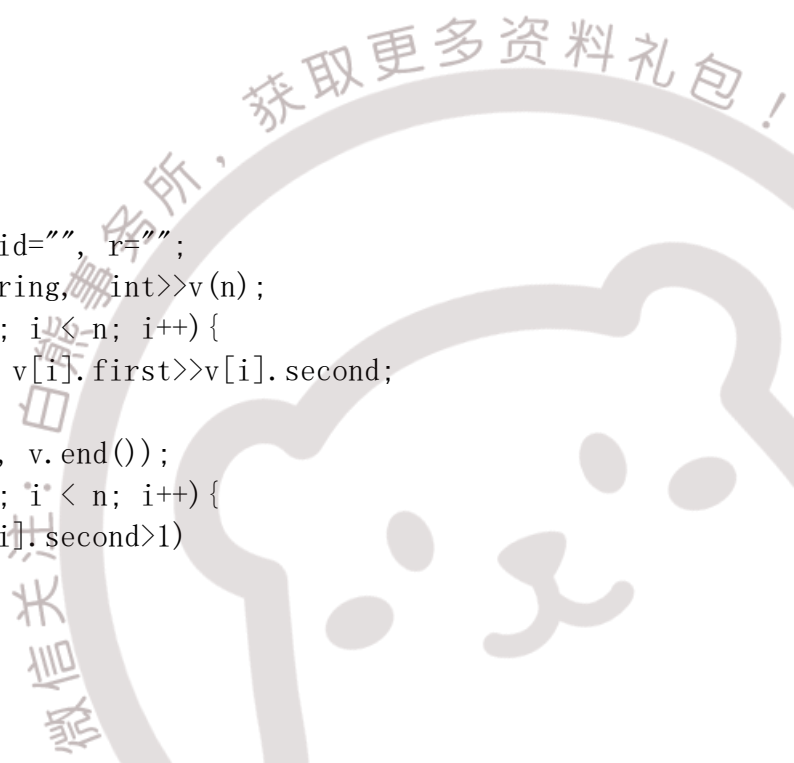


Charm=Sum of all $D(i)$, 其中 $D(i)=k-i$, i 为该排列中卷轴 i 的下标, k 为位于 i 后面且和 i 是同一种卷轴的卷轴下标。

现在所有的卷轴以<卷轴名称 数量>的格式给出, 你需要输出所有卷轴的排列顺序, 使得吸引值最大, 如果有多种排列方式满足条件, 输出按照名字排列字典序最小的一个。

每种字符串的吸引值只与它第一次出现和最后一次出现的位置有关, 所以我们可以先把所有字符串的首尾出现位置确定, 再把其余的字符串塞到中间就行了, 安排字符串位置时均按照字典序由小到大的顺序。

```
1  #include<iostream>
2  #include <sstream>
3  #include<stdio.h>
4  #include<queue>
5  #include<list>
6  #include<set>
7  #include<map>
8  #include<time.h>
9  #include<vector>
10 #include<stack>
11 #include<unordered_set>
12 #include<unordered_map>
13 #include<memory.h>
14 #include<algorithm>
15 #include<numeric>
16 #include<string>
17 #include<functional>
18 #include<limits.h>
19 #include<iterator>
20 using namespace std;
21 #pragma warning(disable:4996)
22
23
24 int main() {
25     int n;
26     while (cin >> n) {
27         string l="", mid="", r="";
28         vector<pair<string, int>>v(n);
29         for (int i = 0; i < n; i++) {
30             cin >> v[i].first>>v[i].second;
31         }
32         sort(v.begin(), v.end());
33         for (int i = 0; i < n; i++) {
34             if (v[i].second>1)
```





```
35         l += v[i].first + ' ', r += v[i].first + ' ',
36     }
37     for (int i = 0; i < n; i++){
38         while (v[i].second>0)
39             mid += v[i].first + ' ', v[i].second--;
40     }
41     string s = l + mid + r;
42     if (s.size())
43         s.pop_back();
44     cout << s << endl;
45     }
46 }
```

