一、单项选择（共 15 题，每题 2 分，共计 30 分，每题有且仅有一个正确选项）

1.完整的计算机系统应该包括（）

A.运算器、存储器、控制器　　B.外部设备和主机

C.主机和应用程序　　D.配套的硬件和软件系统

2.文件传输使用的协议是（）

A .SMTP　　B .FTP

C .UDP　　D .TELNET

3.根据 NAT 协议，下列地址中( )属于 C 类地址。

A. 10.1.56.23　　B. 172.15.34.128

C. 192.168.32.17　　D.172.128.45.34

4.在 Linux 系统中，cp 命令的作用是（ ）

A.更改文档或目录的日期时间　　B.创建一个新的目录

C.复制一个文件到另一个位置　　D.查看指定文件的内容

5.现有一个文件夹，其中包含 1000 张图片，图片的分辨率以 1240*720，1920*1080，1600*900 三个分辨率交替存储，每张图片均为 32 位图像，则这个文件夹最少要用（）GB 的 U 盘存储。

A. 2　　B.4　　C.6　　D.8

6.总共有 6 个不同的元素进栈，能得到（）种不同的出栈序列。

A. 123　　B.121　　C.132.　　D.130

7.已知一个根节点再第一层的完全二叉树的第 6 层有 8 个叶子节点，则该完全二叉树的至少有（）个节点。

A.39　　B.23　　C.111　　D.119

8.一个有 n 个顶点和 n 个边组成的无向图一定是（ ）

A.连通的　　B.无环的　　C.有环的　　D.不连通的

9.已知一个二叉树的中序遍历为 HBDIAEFCG，先序遍历为 ABHDICEFG，则该二叉树的后序遍历为（）

A.IHDBEFGCA　　B.IHDBFEGCA　　C.HIDBFGECA　　D.HIDBFEGCA

10.23|15+9^16&69 的结果是（）

A.31　　B.5　　C. 28　　D.3

11.若序列的原始状态为{73,94,46,20,77,61,51,28,81,58}，使用快速排序进行排序，以第一个记录为基准值，第一次划分结果（）

A.{20,73,58,46,77,28,51,61,81,94}　　B.{58,20,51,61,28,73,77,81,94,46}

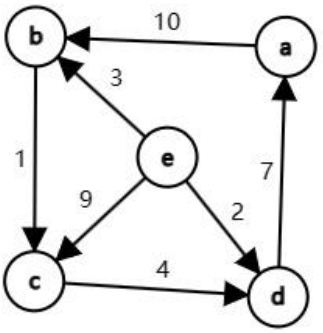
C.{46,20,61,51,28,58,73,94,77,81}　　D.{58,28,46,30,51,61,73,77,81,94}

12.求出下列算法的时间复杂度（）

```
int y=0;
while((y+1)*(y-1)<=n)
{
    y++;
}
```

A.$O(logn)$　　B.$O(n)$　　C.$O\left(n^{\frac{1}{2}}\right)$　　D.$O(N^2)$

13.已知有向图G如下图所示，使用kruskal算法求图G的最小生成树，加到最小生成树中的边依次是( )



A.(a,e),(c,e),(b,e),(b,c),(b,d)　　B.(b,c),(b,d),(b,e),(a,e),(c,e)

C,(a,e),(b,e),(c,e),(b,d),(b,c)　　D.(b,c),(e,d),(e,b),(c,d),(d,a)

14.设一批产品共10件，其中4件是次品，现进行放回抽样，即为取一个检查完放回后继续抽取，总共抽取三次，恰好其中两件为残次品的概率为（）

A.$\frac{3}{10}$　　B.$\frac{36}{125}$　　C.$\frac{1}{2}$　　D.$\frac{13}{25}$

15.假定编译器规定int和short长度范围分别为32位和16位，执行以下代码：

```
unsigned short x=65530;
unsigned int y=x;
```

得到的y的机器码（）

A. 0000 7FFAH　　B. 0000 FFFAH　　C. FFFF 7FFAH　　D. FFFF FFFAH

二、阅读程序（程序输入不超过数组或字符串定义的范围；判断题正确填√，错误填×；除特殊说明外，判断题 1.5 分，选择题 3 分，共计 40 分）

第一题

```
1.  #include <bits/stdc++.h>
2.  long long n, m, t, a[1000005];
3.  using namespace std;
4.      int main() {
5.      cin >> n >> m;
6.      while (m > 0) {
7.          t++;
8.          m--;
9.          if (m <= 0)
10.             break;
11.             a[t] = a[t - 1] + 1;
12.             while (m > (1 << n - a[t]) && a[t] <= n) {
13.                 m -= 1 << n - a[t];
14.                 a[t]++;
15.             }
16.         }
17.     if (t != 1)
18.             for (int i = 1; i < t; i++)
19.                  cout << a[i] << " ";
20.     else
21.             puts("0");
22.     return 0;
23. }
```

16. 第 21 行的"0"改成'0'，程序运行会报错（ ）。

17. 输入的 n 如果是负整数，程序运行不会报错（ ）。

18. 第 11 行的功能是对 a 数组求前缀和（ ）。

19. 输入的 n 最大可以到 1000000（ ）。

20. 输入 4 12， 输出（ ）。

A. 0　　B. 2 3　　C. 2 3 4　　D. 1 2 3

21. 当输入 n 为 3，m 为 1~8 之间的整数时，平均输出整数的个数(保留一位小数)是（ ）。

A. 1.0　　B. 1.6　　C 1.8　　2.0

第二题

```
1.  #include<iostream>
2.  #include<cstring>
3.  using namespace std;
4.  string a,b;
5.  int f[2010][2010];
6.  int Dfs(int i, int j){
7.      if(f[i][j]!=-1)
```

```
8.          return f[i][j];
9.      if(i==0)
10.         return f[i][j]=j;
11.     if(j==0)
12.         return f[i][j]=i;
13.     int c=1;
14.     if(a[i-1]==b[j-1])
15.         c=0;
16.     return f[i][j]=min(min(Dfs(i-1,j)+1,Dfs(i,j-1)+1),Dfs(i-1,j-1)+c);
17. }
18. int main(){
19.     cin>>a>>b;
20.     memset(f,-1,sizeof(f));
21.     int len1=a.length(),len2=b.length();
22.     Dfs(len1,len2);
23.     cout<<f[len1][len2];
24.     return 0;
25. }
```

注：输入的字符串只包含小写字母 a~z。

22. 第 7 行和第 20 行的-1 如果改成-2，程序运行结果可能不一样（ ）。

23. n 是输入两个字符串长度的最大值，程序的时间复杂度为（ ）。

24. 输出的最小值为-1（ ）。

25. 设 len1 和 len2 为执行完第 21 行后的值，则程序输出的最大值为 len1+len2（ ）。

26. 输入为 sfdqxbw　gfdgw，输出（ ）。

A. 2　　B. 3　　C. 4　　D. 5

27. 保证输入的两个字符串长度均为 100，且只包含小写字母 a~z，
第一个字符串为"acegikmoqsuwyace...uwy...ace...moq"，
第二个字符串为"abcdefghijklmnopqrstuvwxy...abc...wxyabcd"
（“...”表示省略了中间的字符），输出为（ ）。（4 分）

A. 4　　B. 13　　C. 50　　D. 96

第三题

```
1.  #include<iostream>
2.  #define mod 9901
3.  using namespace std;
4.  int a,b,sa,cal[10010][2],cnt=0,ans=1;
5.  int f(int ml,int nl){
6.      int s=1;
7.      while(nl>0){
8.          if(nl&1){
9.              s=s*ml%mod;
10.         }
11.         ml=ml*ml%mod;
```

```
        nl=nl>>1;
    }
    return s%mod;
}
int s(int x,int y){
    int res=0;
    y=y*b;
    if(x%mod==1){
        res=(y+1)%mod;
    }
    else{
        res=(f(x%mod,y+1)-1)%mod * f((x-1)%mod,mod-2)%mod;
    }
    return res%mod;
}
int main(){
    cin>>a>>b;
    if(a==0){
        cout<<0<<endl;
        return 0;
    }
    for(int i=2;i*i<=a;i++){
        if(a%i==0){
            cnt++;
            cal[cnt][0]=i;
            cal[cnt][1]=1;
            a=a/i;
            while(a%i==0){
                cal[cnt][1]++;
                a=a/i;
            }
        }
    }
    if(a!=1){
        cnt++;
        cal[cnt][0]=a;
        cal[cnt][1]=1;
    }
    for(int i=1;i<=cnt;i++){
        ans=ans*s(cal[i][0],cal[i][1])%mod;
    }
    cout << (ans%mod+mod)%mod;
    return 0;
}
```

28. 第 33 行的 i*i 改成 i,运行结果不变（ ）。
29. 第 2 行 9901 换成 1e9+7，运行结果不变（ ）。
30. cal[cnt][0]随着 cnt 的增大而增大（ ）。
31. 第 25 行删除%mod，结果不变（ ）。
32. 输入为 2　3，输出（ ）。

A.4　　B. 8　　C. 9　　D. 15

33. 输出为 4319，输入可能是以下哪组数据（ ）。

A.210　2　　B. 210　3　　C. 330　5　　D. 330　6

34. 输入为 217823　1，输出为（ ）。

A.1　B. 2　　C. 9901　D.22

三、**完善程序（单选题，每小题 3 分，共计 30 分）**

1、（奇数区间）给定一个长度为 n 的数列：$a_1, a_2, \cdots a_n$，如果其中一段连续的子序列 $a_i, a_{i+1}, \cdots a_j (i \le j)$中，奇数比偶数多，我们就称这个区间[i,j] 是奇数区间。求给定的 n 个数中有多少个奇数区间。

```
#include<bits/stdc++.h>
using namespace std;
const int off = 1e6 + 1;
const int maxn = 1e6 + 5;
int a[maxn], s[maxn], c[2 * maxn];
int n;
long long ans;
int lowbit(int x){
    return x & -x;
}
long long getSum(int x){
    long long res = 0;
    while(x > 0){
        res += c[x];
        ①     ;
    }
    return res;
}
void add(int x, int k){
    while(     ②     ){
        c[x] += k;
        ③     ;
    }
}
int main(){
    cin >> n;
    add(off, 1);
```

```
    for(int i = 1; i <= n; i++){
        cin >> a[i];
        s[i] =      ④      ;
        ans += getSum(      ⑤      );
        add(s[i] + off,1);
    }
    cout << ans << endl;
    return 0;
}
```

35. ①处应该填（ ）

A. x -= lowbit(x)

B. x = lowbit(x)

C. x += lowbit(x)

D. x--

36. ②处应该填（ ）

A. $x < off$

B. $x <= n$

C. $x! = 0$

D. $x <= 2 * off$

37. ③处应该填（ ）

A. x -= lowbit(x)

B. x = lowbit(x)

C. x += lowbit(x)

D. x++

38. ④处应该填（ ）

A. s[i - 1] + a[i] % 2 ? 1 : -1

B. s[i - 1] + a[i] % 2 == 1 ? -1 : 1

C. s[i - 1] + a[i] % 2 != 0 ? 1 : -1

D. s[i - 1] + !a[i] % 2 ? 1 : -1

39. ⑤处应该填（ ）

A. 0

B. $s[i] + off - 1$

C. $s[i - 1]$

D. $s[i] + off$

2、（**最近公共祖先**）已知一棵有根的多叉树，每次询问下求给定的两个点的最近公共祖先。其中输入的第一行 3 个整数为结点个数，询问次数，根节点序号。

祖先结点:指一个结点本身或者它父节点的祖先。**最近公共祖先：**在一棵有根树中，对于任意两个结点的公共祖先中距离两者最近的结点。

```
#include <bits/stdc++.h>
using namespace std;
const int MAXN = 500010;
vector<int> edge[MAXN];
int depth[MAXN], lg[MAXN];
int pa[MAXN][22];
int n, m, r, x, y;
void dfs(int fa, int sn, int dep) {
    depth[sn] = dep;
    pa[sn][0] = fa;
    for(int i = 1;      ①     ; i++) {
        pa[sn][i] = pa[pa[sn][i - 1]][i - 1];
    }
    for(int i = 0; i < edge[sn].size(); i++) {
        int fn = edge[sn][i];
        if (fn == fa) {
            continue;
        }
        dfs(     ②     );
    }
}
int solve(int x, int y) {
    if (depth[x] < depth[y]) {
        swap(x, y);
    }
    while(depth[x] > depth[y]) {
        x =      ③     ;
    }
    if(x == y) {
        return x;
    }
    for(int k = lg[depth[x]] - 1; k >= 0; --k) {
        if(     ④     ) {
            x = pa[x][k], y = pa[y][k];
        }
    }
    return pa[x][0];
}
int main() {
    cin >> n >> m >> r;
    for(int i = 1; i <= n; i++)
        lg[i] = lg[i - 1] + (1 << lg[i - 1] == i);
    for (int i = 0; i < n - 1; i++) {
        cin >> x >> y;
```

```
        edge[x].push_back(y);
        edge[y].push_back(x);
    }
    dfs(     ⑤     );
    while (m--) {
        cin >> x >> y;
        cout << solve(x, y) << endl;
    }
    return 0;
}
```

40. ①处应该填（ ）

A. $i < pa[sn].size()$

B. $i <= lg[depth[fn]$

C. $i < lg[dep]$

D. $i <= lg\left[depth[sn]\right]$

41. ②处应该填（ ）

A. $sn, fn, dep + 1$

B. $fn, sn, dep + 1$

C. sn, fn, dep

D. fn, sn, dep

42. ③处应该填（ ）

A. $lg\left[depth[x] - depth[y]\right]$

B. $pa[x]\left[lg\left[depth[x] - depth[y]\right] - 1\right]$

C. $pa[y][lg[depth[x]$

D. $pa[y][lg\left[depth[y] - depth[x]\right]$

43. ④处应该填（ ）

A. $pa[k][y] == pa[k][x]$

B. $!pa[x][k]$

C. $pa[x][k]! = pa[y][k]$

D. $pa[x][k] == pa[y][k]$

44. ⑤处应该填（ ）

A. $r, 0, 0$

B. $0, r, 0$

C. $r, 0, 1$

D. $0, r, 1$