CSP2024-S组初赛模拟

1. 【单项选择】第1-15题，每题2分，共30分
2. 【阅读程序1】第16-20题，判断每题2分，单选每题3分，共13分
3. 【阅读程序2】第21-25题，判断每题2分，单选每题3分，共13分
4. 【阅读程序2】第26-31题，判断每题2分，单选每题2分或3分，共14分
5. 【完善程序1】第32-35题，每题3分，共12分
6. 【完善程序2】第36-41题，每题3分，共18分

【单项选择】 第1-15题，每题2分，共30分

1. 请选出以下最大的数（ ）。
$(1067)_{10}$
$(10000110001)_2$
$(40A)_{16}$
$(1777)_8$

2. 已知一棵二叉树有 2018 个节点，则其中至多有（ ）个节点有 2 个子节点。
1008
1009
1006
1010

3. 已知数列递推式为：$a[n]=a[n-1]+12\times a[n-2]$, 初值 $a[0]=3,a[1]=26$，则数列的通项公式为（ ）
$a[n]=5\times 3^n-2\times(-4)^n$
$a[n]=5\times 4^n-2\times(-3)^n$
$a[n]=4^n+2\times(-3)^n$
$a[n]=5\times 3^n-2\times 4^n$

4.现有一段 8 分钟的视频文件，它的播放速度是每秒 24 帧图像，每帧图像是 一幅分辨率为 2048×1024 像素的 256 位真彩色图像。请问要存储这段原始无压缩视频，需要多大的存储空间？（）。
120GB
720GB
15360MB
900GB

5. 由 a, b, c, d 这4个字符构成的长度 $= 5$ 的字符串，其中 a, b, c 都至少出现一次的字符串有（）。
390
391
294
295

6. 将 6 个无标号的球放进 3 个有标号的盒子里，允许有盒子不放球，有（）种方案。
729
20
56
28

7. $[1, 2020]$ 这些数字中 **1. 数码 2 一共出现多少次？例如 2020 中 2 出现 2 次，222 中 2 出现 3 次** **2. 包含 2 的数字有多少个？例如 $12, 222$ 等都包含 2**
$624, 563$
$624, 421$
$621, 563$
$621, 421$

8. 下列算法中，（）不是稳定的排序算法
插入排序
冒泡排序
选择排序
归并排序

9. 下列哪个问题的求解过程不涉及贪心？（）

霍夫曼编码

KMP字符串查找

最小生成树

选择最少的线段覆盖给定区间 $[L, R]$

10. 二分图是指能将点集划分成两个部分，每一部分内的点之间没有边相连的简单无向图（简单图指没有重边）。那么，24 个点的二分图至多有（）条边。

24

48

144

72

11. 对一个 n 个顶点、m 条边的无向简单图用 Prim 算法求最小生成树，使用、不使用堆优化的情况下，则其时间复杂度分别为（）。

$O((n+m)logn), O(n^2+m)$

$O((n^2+m)logn), O(n^2+m)$

$O((n+m)logn), O((n+m)^2)$

$O((n^2+m)logn), O((n+m)^2)$

```
int n = 10;
int cnt = 0;
for(int s = 1;s < (1<<n);++s){
    for(int s1 = s;s1 != 0;s1 = (s1-1)&s){
        ++cnt;
    }
}
```

12. 上述程序段执行结束后，`cnt` 的值为（）。

59049

58025

59050

59048

13. 令根结点的高度为 1，则一棵含有 2021 个结点的二叉树的高度至少为（）。
10
11
12
2021

14. 两棵形态不同、大小为 n 的二叉树 T_1, T_2，已知 T_1 的前序遍历和 T_2 的前序遍历完全相同，并且 T_1 的后序遍历和 T_2 的后序遍历完全相同，则 n 最小为：
2
4
1
3

15.将数组 $[8, 23, 4, 16, 77]$ 中的元素按从大到小的顺序排列，每次可以交换任意两个元素，最少需要交换（）次。
5
4
6
3

【阅读程序1】第16-20题，判断每题2分，单选每题3分，共13分

```
int n;
int l[MAXN], r[MAXN], sz[MAXN];
int cnt[MAXN], dep[MAXN];
int v[MAXN][MAXN];
void dfs1(int u){
    sz[u] = 1;
    if(l[u] != -1){
        dep[l[u]] = dep[u] + 1;
        dfs_sz(l[u]);
        sz[u] += sz[l[u]];
    }
    if(r[u] != -1){
        dep[r[u]] = dep[u] + 1;
        dfs_sz(r[u]);
        sz[u] += sz[r[u]];
    }
}

bool chk(int x, int y){
    if(x == -1 && y == -1) return 1;
```

```
    if(x == -1 || y == -1) return 0;
    ++cnt[x]; ++cnt[y];
    ++v[x][y];
    return chk(l[x],r[y]) && chk(r[x],l[y]);
}

int main(){
    cin>>n;
    for(int i=1;i<=n;++i) cin>>l[i]>>r[i];

    dep[1] = 1;
    dfs1(1);
    int ans = 0;
    for(int i=1;i<=n;++i){
        if(chk(l[i],r[i])){
            ans = max(ans, sz[i]);
        }
    }
    cout<<ans;
    return 0;
}
```

输入为一棵 1 为根的二叉树，给定每个点的左右子节点编号 $l[1\sim n]/r[1\sim n]$，如果没有以 `-1` 指代。

16. 对于任意输入，程序运行结束时，任意点 $1\le i\le n$ 满足 $cnt[i]\le dep[i]-1$
对
错

17. 对于任意输入，程序运行结束时，对于任意2个点 $1\le x,y\le n$，$v[x][y]\le 1$
对
错

18. 对于任意输入，程序的最坏时间复杂度为
$O(n^2)$
$O(nlogn)$
$O(n)$
$O(2^n)$

19. 若输入为 $n=1023$ 的满二叉树，则 $\sum cnt[i]$ 的值为：
8194
8192
4096
4098

20. 若输入为 $n=11$ 的完全二叉树，则 $\sum cnt[i]$ 的值为：

15

16

17

18

【阅读程序2】第21-25题，判断每题2分，单选每题3分，共13分

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef unsigned long long ull;
ull n,k;
string s;
int main(){
    cin>>n>>k;
    for(int i=n;i>=1;i--){
        ull w = 1ll<<(i-1);
        if(k < w){
            s += '0';
        }
        else{
            s += '1';
            k -= w;
            k = w - 1 - k;
        }
    }
    cout<<s<<endl;
    return 0;
}
```

输入 n 为大于 1 的整数，k 为 $[0,2^n)$ 范围内整数。

21. 把程序第10行改为 `ll w = 1ll<<(i-1)`，输出结果必定不变。

对

错

22. 若想要程序正常完成功能，n 最大值为 63。

对

错

23. 当输入为 $n = 64, k = 2^{64} - 1$ 时，输出为（）。

`1000000000000000000000000000000000000000000000000000000000000000`

`0000000000000000000000000000000000000000000000000000000000000000`

`1111111111111111111111111111111111111111111111111111111111111111`

`0000000000000000000000000000000000000000000000000000000000000001`

24. 若输入为 $n = 8, k = 100$，输出为（）。

`01100100`

`01010110`

`01100110`

`10010110`

25. 若输出为 `0001111010110001`，输入的 k 为（）。

649

7857

352

5342

【阅读程序3】第26-31题，判断每题2分，单选每题2分或3分，共14分

```
const int MAXN = 1024;
int cnt1[MAXN][MAXN], cnt2[MAXN][MAXN];
int n = 1024;
for(int i=1;i<n;++i){
    //__builtin_clz(x) 返回x二进制从最高位开始连续0的个数
    int d = __builtin_clz(i^(i-1));
    for(int x=0;x<n;++x){
        cnt1[x][d]++;
        for(int y=0;y<n;++y){
            cnt2[x][y] += (__builtin_clz(x^y) == d);
        }
    }
}
```

26. 对于任意 $0 \le x, y < n$，$\sum_j cnt2[x][j] = \sum_j cnt2[y][j]$ 成立

对

错

27. 对于任意 $0 \le x, y < n, 0 \le j < 32$，$cnt1[x][j] = cnt1[y][j]$ 成立
对
错

28. $d \neq 0$ 恒成立
对
错

29（本题2分）. $\sum_j cnt1[0][j]$ 的大小（ ）
1024
1023
2047
2048

30（本题3分）. $\sum_j cnt2[n-1][j]$ 的大小（ ）
4096
5120
5603
5632

31（本题3分）. $\sum_x \sum_j cnt2[x][j]$ 的大小（ ）
5242880
5737472
5766656
5767168

【完善程序1】第32-35题，每题3分，共12分

N 皇后问题

输入一个 $N \times N$ 的棋盘，`*` 表示可放，`.` 表示不可放。在棋盘上放置 N 个互不攻击的皇后，求方案数。

搜索剪枝，利用位运算维护每列和每个对角线的状态，处理出每行合法的位置状态，遍历 合法位置放置皇后。

```
1 #include<bits/stdc++.h>
```

```cpp
using namespace std;
int N,U;
char s[15];
int ban[15];
int ans = 0;
void dfs(int r, int col, int ld, int rd){//当前在第r行，列状态col，对角线状态ld/rd
    int s = ?32;
    if(r == N){
        ans += __builtin_popcount(s);//__builtin_popcount(s)返回s二进制中1的数量
        return;
    }

    while(s){
        int j = ?33;//放置皇后
        dfs(r+1, col^j, ?34);
        s ^= j;
    }
}
int main(){
    cin>>N;
    U = ?35;
    for(int i=1;i<=N;i++){
        cin>>s;
        for(int j=0;j<N;j++){
            if(s[j]=='.') ban[i] |= (1<<j);
        }
    }
    dfs(1,0,0,0);
    cout<<ans;
    return 0;
}
```

32.

- `U^(col|ld|rd|ban[r])`
- `(~col)|(~ld)|(~rd)|(~ban[r])`
- `U&(~(col|ld|rd|ban[r]))`
- `(~col)&(~ld)&(~rd)&(~ban[r])`

33.

- `s & (-s)`
- `s ^ (-s)`
- `s & (s - 1)`
- `s ^ (s - 1)`

34.

`(ld^j)>>1, (rd^j)>>1`

`(ld^j)<<1, (rd^j)>>1`

`ld^(1<<j), rd^(1<<j)`

`ld^(1<<j), rd^(1>>j)`

35.

`N-1`

`(1<<(N+1))-1`

`1<<N`

`(1<<N) - 1`

【完善程序2】第36-41题，每题3分，共18分

树上 k 级祖先，给一棵有根树（1 号点为根），首先长链剖分预处理，在每个链头点处：存储这条链自上而下的所有节点信息、以及这个点自下而上的相同长度祖先信息。

当查询 x 的 k 级祖先时，先把 x 跳到 k 第一个二进制位代表的数的位置（倍增预处理），这样就可以把剩下的距离缩减到 $\lfloor \frac{k}{2} \rfloor$ 以下，可以用在链头处预处理的信息 $O(1)$ 回答询问。

查询保证 k 级祖先存在。

```
#include<bits/stdc++.h>
#define MAXN 100005
using namespace std;

int n,q;
vector<int> adj[MAXN];
int anc[MAXN][21];
int son[MAXN], h[MAXN], dep[MAXN], top[MAXN];

vector<int> up[MAXN], down[MAXN];
int dfn[MAXN], tt = 0;

void dfs1(int u){
    for(int j=1;j<=20;j++) anc[u][j] = anc[anc[u][j-1]][j-1];
    for(int v: adj[u]){
        dep[v] = dep[u] + 1;
        anc[v][0] = u;
        dfs1(v);
        if(?36){
            son[u] = v;
            h[u] = h[v] + 1;
        }
    }
}
```

```cpp

void dfs2(int u, int x){
    dfn[u] = ++tt;
    top[u] = x;
    ?37;

    if(!son[u]) return;
    dfs2(son[u], x);

    for(int v: adj[u]){
        if(v == son[u]) continue;
        dfs2(v,v);
        int len = ?38, cur = v;
        for(int i=0;i<len;i++){
            ?39;
            cur = anc[cur][0];
            if(cur==0) break;
        }
    }
}

int query(int x, int k){
    if(k==0) return x;
    int j = log2(k);
    x = anc[x][j];
    k -= (1<<j);

    int y = top[x];
    if(dep[x] - dep[y] >= k) return ?40;
    else return ?41;
}

int main(){
    cin>>n>>q;
    int u;
    for(int i=2;i<=n;i++){
        cin>>u;
        adj[u].push_back(i);
    }

    dep[1] = 1;
    dfs1(1);
    dfs2(1, 1);

    int x, k;
    while(q--){
        cin>>x>>k;
        cout<<query(x,k)<<'\n';
    }
    return 0;
}
```

36.

`dep[v] > dep[son[u]]`

`h[v] < h[son[u]]`

`h[v] + 1 > h[u]`

`dep[v] < dep[son[u]]`

38.

`down[x].push_back(u)`

`up[u].push_back(x)`

`down[u].push_back(x)`

`up[x].push_back(u)`

39.

`dep[v]`

`h[v]`

`down[v].size()`

`up[v].size()`

39.

`up[v].push_back(cur)`

`up[cur].push_back(v)`

`down[v].push_back(cur)`

`down[cur].push_back(v)`

40.

`down[y][dep[x] - dep[y] - k]`

`up[y][k - dep[x] + dep[y]]`

`down[y][dep[x] + dep[y] - k]`

`up[y][k - dep[x] - dep[y]]`

41.
`down[y][dep[x] - dep[y] - k]`
`up[y][k - dep[x] + dep[y]]`
`down[y][dep[x] + dep[y] - k]`
`up[y][k - dep[x] - dep[y]]`