A.password

把给出的数字转换成二进制，按题意处理即可，可能出现密码为空的情况。

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<algorithm>
#include<cmath>
#include<queue>
using namespace std;
typedef long long ll;
int main()
{
	int n,a;
	string s="";
	scanf("%d",&n);
	while(n--)
	{
		scanf("%d",&a);
		int num=0,k=a%8;
		for(int i=0;i<3;i++)
		{
			if(a%2) num++;
			a/=2;
		}
		if(num%2==a) s+='0'+k;
	}
	cout<<s<<endl;
	return 0;
}
```

B.watcher

推导，根据二叉树结构找到规律。

```
#include<iostream>
```

```
#include<cmath>
using namespace std;
int main()
{
	int n,m;
	cin>>n>>m;
	int p=1,t=2;
	while(m>=t)
	{
		p++;
		t*=2;
	}
	cout<<(int)(pow(2,n-1)-pow(2,n-p)+1)<<" "<<p<<endl;
	return 0;
}
```

C.decompose

质因数分解模板题。

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include<cmath>
using namespace std;
int ans[100];
int main() {
	int n;
	scanf("%d", &n);
	int x;
	while(n--)
	{
		scanf("%d",&x);
		int cnt=0,end=sqrt(x);
		for(int i=2;i<=end;i++)
		{
			if(x%i==0)
			{
				ans[cnt++]=i;
				while(x%i==0) x/=i;
			}
```

```
            }
            if(x!=1) ans[cnt++]=x;
            for(int i=0;i<cnt-1;i++)
                printf("%d ",ans[i]);
            printf("%d\n",ans[cnt-1]);
        }
        return 0;
}
```

D.cableway

用 dfs 或者并查集求出连通块个数-1 既是答案。

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<vector>
using namespace std;
const int N=10000+10;
vector<int> g[N];
bool vis[N];
void dfs(int u)
{
        vis[u]=true;
        for(int i=0;i<g[u].size();i++)
            if(!vis[g[u][i]]) dfs(g[u][i]);
}
int main()
{
        int n,m;
        scanf("%d%d",&n,&m);
        int u,v;
        for(int i=0;i<m;i++)
        {
            scanf("%d%d",&u,&v);
            g[u].push_back(v);
            g[v].push_back(u);
        }
        memset(vis,false,sizeof(vis));
        int ans=0;
        for(int i=1;i<=n;i++)
```

```
            if(!vis[i])
            {
                dfs(i);
                ans++;
            }
        printf("%d\n",ans-1);
        return 0;
}
```

E.pancake

用单调栈来维护最大值，数据量不是很大，效率高的话可能用其他 O(nlogn)的方法解决。

```
#include<iostream>
#include<cstring>
#include<cstdio>
#include<ctime>
#include<cstdlib>
using namespace std;
const int N=1e6+10;
int p[N];
int main()
{
        int T;
        scanf("%d",&T);
        int a,b;
        p[0]=0;
        int top=1;
        while(T--)
        {
            scanf("%d",&a);
            if(a==1)
            {
                scanf("%d",&b);
                p[top]=max(p[top-1],b);
                top++;
            }
            else
            {
                printf("%d\n",p[top-1]);
                top--;
```

```
            }
        }
        return 0;
}
```

F.exfibonacci

可以发现数列第一项为 1，后面第 n 项的值就是 2^(n-2)。然后用快速幂解决。

```
#include<iostream>
#include<cstdio>
using namespace std;
typedef long long ll;
const ll MOD=1e9+7;
ll expow(ll a,ll b)
{
        ll ret=1;
        while(b)
        {
                if(b&1)
                {
                        ret*=a;
                        ret%=MOD;
                }
                a*=a;
                a%=MOD;
                b>>=1;
        }
        return ret;
}
int main()
{
        int T;
        ll n;
        scanf("%d",&T);
        while(T--)
        {
                scanf("%lld",&n);
                if(n<=2) printf("1\n");
                else printf("%lld\n",expow(2,n-2));
        }
```

```
    return 0;
}
```

G.knowledge

枚举区间长度，尺取求最大值，或者其他解决方法。注意最大值为负数时，输出 0。

```
#include<iostream>
#include<cstdio>
#include<cstring>
using namespace std;
const int N=1e4+10;
int a[N];
int main()
{
    int n,K;
    scanf("%d%d",&n,&K);
    for(int i=0;i<n;i++)
        scanf("%d",&a[i]);
    int ans=0;
    for(int k=1;k<=K;k++)
    {
        int l=0,r=1,t=a[0];
        ans=max(ans,t);
        while(r<n)
        {
            if(r-l>=k) t-=a[l++];
            t+=a[r++];
            ans=max(ans,t);
        }
    }
    printf("%d\n",ans);
    return 0;
}
```

H.escape

动态规划，当无法到达第 N 个石块时，输出 0。

```
#include<iostream>
#include<cstring>
#include<cstdio>
#include<ctime>
#include<cstdlib>
using namespace std;
const int N=1e4+10;
const int MOD=1e9+7;
int dp[N];
bool vis[N];
int main()
{
	int n,k,m,a;
	scanf("%d%d%d",&n,&k,&m);
	memset(vis,false,sizeof(vis));
	for(int i=0;i<m;i++)
	{
		scanf("%d",&a);
		vis[a]=true;
	}
	dp[1]=1;
	for(int i=2;i<=n;i++)
	{
		dp[i]=0;
		for(int j=1;j<=k;j++)
		{
			if(i-j<1) break;
			if(vis[i-j]) continue;
			dp[i]=(dp[i]+dp[i-j])%MOD;
		}
	}
	printf("%d\n",dp[n]);
	return 0;
}
```

I.database

简单模拟，如果熟练使用 vector 会很简单。

```
#include<iostream>
#include<cstdio>
#include<algorithm>
#include<vector>
using namespace std;
const int N=1000+10;
vector<string> list;
int main()
{
	int T;
	cin>>T;
	string a,s;
	int p1,p2;
	while(T--)
	{
		cin>>a;
		if(a[0]=='i')
		{
			cin>>p1>>s;
			list.insert(list.begin()+p1-1,s);
		}
		else if(a[0]=='d')
		{
			cin>>p1;
			list.erase(list.begin()+p1-1);
		}
		else if(a[0]=='s')
		{
			cin>>p1>>p2;
			sort(list.begin()+p1-1,list.begin()+p2);
		}
		else if(a[0]=='f')
		{
			cin>>s;
			for(int i=0;i<s.length();i++)
				s[i]=tolower(s[i]);
			for(int i=0;i<list.size();i++)
			{
				string s0=list[i];
				for(int j=0;j<s0.length();j++)
					s0[j]=tolower(s0[j]);
				if(s0.find(s)!=string::npos)
					cout<<list[i]<<endl;
			}
```

```
            }
        }
        return 0;
}
```

J.cave

较复杂的 bfs 搜索，详见代码。

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<algorithm>
#include<cmath>
#include<queue>
using namespace std;
typedef long long ll;
const int N=100+10;
const int dx[]={0,0,-1,1};
const int dy[]={-1,1,0,0};
struct node
{
        int x,y,z;
        node(){}
        node(int a,int b,int c)
        {
                x=a;y=b;z=c;
        }
};
int g[N][N][N];
int dist[N][N][N];
int n,m;
node judge1(node a,int i)
{
        node ret=node(-1,-1,-1);
        int x=a.x+dx[i],y=a.y+dy[i],z=a.z;
        if(g[x][y][z]==1)
        {
                if(g[x][y][z+1]!=1&&g[a.x][a.y][a.z+1]!=1)
                {
                        ret=node(x,y,z+1);
```

```
            }
        }
        else
        {
            if(g[x][y][z-1]==1)
            {
                ret=node(x,y,z);
            }
            else if(g[x][y][z-2]==1)
            {
                ret=node(x,y,z-1);
            }
            else if(g[x][y][z-3]==1)
            {
                ret=node(x,y,z-2);
            }
        }
        return ret;
}
node judge2(node a,int i)
{
        node ret=node(-1,-1,-1);
        int x=a.x+dx[i],y=a.y+dy[i],z=a.z;
        if(g[x][y][z]!=1)
        {
            if(g[x][y][z-1]==1)
            {
                ret=node(x,y,z);
            }
            else if(g[x][y][z-2]==1)
            {
                ret=node(x,y,z-1);
            }
        }
        return ret;
}
int bfs1(node s)
{
        queue<node> q;
        memset(dist,-1,sizeof(dist));
        q.push(s);
        dist[s.x][s.y][s.z]=0;
        while(!q.empty())
        {
```

```
            node u=q.front();
            q.pop();
            for(int i=0;i<4;i++)
            {
                node v=judge1(u,i);
                if(v.x<=0||v.y<=0||v.z<=0||v.x>n||v.y>n||v.z>n) continue;
                if(dist[v.x][v.y][v.z]!=-1) continue;
                dist[v.x][v.y][v.z]=dist[u.x][u.y][u.z]+1;
                if(g[v.x][v.y][v.z]==3) return dist[v.x][v.y][v.z];
                q.push(v);
            }
        }
}
int bfs2(node s)
{
        queue<node> q;
        memset(dist,-1,sizeof(dist));
        q.push(s);
        dist[s.x][s.y][s.z]=0;
        while(!q.empty())
        {
            node u=q.front();
            q.pop();
            for(int i=0;i<4;i++)
            {
                node v=judge2(u,i);
                if(v.x<=0||v.y<=0||v.z<=0||v.x>n||v.y>n||v.z>n) continue;
                if(dist[v.x][v.y][v.z]!=-1) continue;
                dist[v.x][v.y][v.z]=dist[u.x][u.y][u.z]+1;
                if(g[v.x][v.y][v.z]==4) return dist[v.x][v.y][v.z];
                q.push(v);
            }
        }
}
int main()
{
        node a,b;
        scanf("%d%d",&n,&m);
        int x,y,z;
        memset(g,0,sizeof(g));
        for(int i=0;i<m;i++)
        {
            scanf("%d%d%d",&x,&y,&z);
            g[x][y][z]=1;
```

```
        }
        scanf("%d%d%d",&a.x,&a.y,&a.z);
        scanf("%d%d%d",&b.x,&b.y,&b.z);
        g[a.x][a.y][a.z]=2;
        g[b.x][b.y][b.z]=3;
        int e;
        scanf("%d",&e);
        while(e--)
        {
            scanf("%d%d%d",&x,&y,&z);
            g[x][y][z]=4;
        }
        int ans=bfs1(a)+bfs2(b);
        printf("%d\n",ans);
        return 0;
}
```