FINSALS KING C++

```
vector<vector<int>>v(8,vector<int>(8));
const int dx[]= {-1,-1,-1, 0,0, 1,1,1};
const int dy[]= {-1, 0, 1, -1,1, -1,0,1};
bool inside(int x, int y)
{
 return x>=0&&y>=0&&x<8&&y<8;
}
void addqueen(int col,int lin)
{
 for(int t=0; t<8; t++)
 for(int i=1;; i++)
 {
 int nx=lin+dx[t]*i;
 int ny=col+dy[t]*i;
 if(!inside(nx,ny))
 break;
v[nx][ny]|=1;

v[nx][ny]| 1;
 if((v[nx][ny]&2))
 break;
 }
}
void addrook(int col,int lin)
{
 for(int t:
 {
})
```

```
 for(int i=1;; i++)
 {
 int nx=lin+dx[t]*i;
 int ny=col+dy[t]*i;
 if(!inside(nx,ny))
 break;
 v[nx][ny]|=1;
 if((v[nx][ny]&2))
 break;
 }
}
void addbishop(int col,int lin)
{
 for(int t:
 {
 0,2,5,7
 })
 for(int i=1;; i++)
 {
 int nx=lin+dx[t]*i;
 int ny=col+dy[t]*i;
 if(!inside(nx,ny))
 break;
 v[nx][ny]|=1;
 if((v[nx][ny]&2))
 break;
 }
}
void addknight(int col,int lin)
```

```
{
 static const int dx[]= {-2,-2,-1,-1, 1,1, 2,2};
 static const int dy[]= {-1, 1,-2, 2,-2,2,-1,1};
 for(int i=0; i<8; i++)
 {
 int nx=dx[i]+lin;
 int ny=dy[i]+col;
 if(inside(nx,ny))
 v[nx][ny]|=1;
 }
}
void addpawn(int col,int lin)
{
 for(int t:{5,7})
 {
 int nx=dx[t]+lin;
 int ny=dy[t]+col;
 if(inside(nx,ny))
 v[nx][ny]|=1;
 }
}
string find_king_status(string s)
{
 int lin,col;
for(int i=0; i<s size(); i+=4)
for(int i=0; i<s.size(); i+=4)
 if(s[i]!='K')
 v[s[i+2]-'1'][s[i+1]-'a']=2;
 for(int i=0; i<s.size(); i+=4)
```

```
if(s[i]=='K')
col=s[i+1]-'a', lin=s[i+2]-'1';
else if (s[i]=='Q')
addqueen(s[i+1]-'a',s[i+2]-'1');
else if (s[i]=='R')
addrook(s[i+1]-'a',s[i+2]-'1');
else if(s[i]=='B')
addbishop(s[i+1]-'a',s[i+2]-'1');
else if(s[i]=='N')
addknight(s[i+1]-'a',s[i+2]-'1');
else if(s[i]=='P')
addpawn(s[i+1]-'a',s[i+2]-'1');
/*cout<<lin<<" "<<col<<'\n';
for(int i=0;i<8;i++,cout<<endl)
for(int j=0;j<8;j++)
cout<<v[i][j]<<" ";*/
if(v[lin][col]==0)
{
int sm=0,ct=0;
for(int i=0;i<8;i++)
{
int nx=dx[i]+lin;
int ny=dy[i]+col;
if(inside(nx,ny))
{
ct++;
if((v[nx][ny])&1)
sm++;
}
```

```
}
if(sm==ct)
return "STALEMATE";
return "SAFE";
}
int sm=0,ct=0;
for(int i=0;i<8;i++)
{
int nx=dx[i]+lin;
int ny=dy[i]+col;
if(inside(nx,ny))
{
ct++;
if((v[nx][ny])&1)
sm++;
}
}
if(sm==ct)
return "CHECKMATE";
return "CHECK";
}
```