Type ManageNode struct{

IP: string,

Port: uint64,

//当前集群所有节点的ip

Nodes:[]string,

//当前集群领导者的ip地址

Leader:string

}

Type ManageNode interface{

newManageNode(IP string,port uint64)(\*ManageNode,error)

//FirstNode(NodeIP string) (uint64,bool)

//NewNode(NodeIP string) (uint64,bool)

//OldNode(NodeIP string) (uint64,bool)

//JoinCluster（IP string）（uint64,bool）

}

Func main(){

输入本机登陆的IP和port（或者自动获取）；

初始化入口节点：EntryNode，\_:=newManageNode（ip,port）；

端口开始监听（节点加入集群的请求）；

设置一个定时器（定时为领导者发送心跳的时间间隔）；

for{

select{

case 收到请求：

switch{

case 第一个加入集群的新节点：

append(EntryNode.Nodes,conn.ip)

EntryNode.leader=conn.ip

允许加入集群，回复消息告知节点

case 新节点：

//集群中领导者下线不允许加入集群，影响投票选举的结果

if EntryNode.leader==” “{不允许加入，回复消息}

else{append（EntryNode.Nodes,conn.ip）;允许加入，回复消息}

case 旧节点：

允许加入，回复消息

case 领导者心跳：

重置定时器

}

局域网广播节点加入集群的消息

case 定时器超时：

EntryNode.leader=” ”

重置定时器

}

}

}