**合约接口说明**

一共有两个表，chainTable是（name，sum）储存账户余额

historyTable是（name，event，number）储存账户历史消息

name、event是string或者bytes32类型 ； sum、number是int

emit XXX (“………OK"); 是日志，有返回信息，操作成功/失败

**findChainTable(string name) returns(bytes32, int)**

输入name 返回查询的name、余额

**findHistoryTable(string name) returns(bytes32[], bytes32[], int[])**

输入name 返回查询的name、干了什么、花/收了多少

**newAccount(string name)**

输入name创建新账户余额为0

日志：emit newAccountResult("newAccount\_OK");

**makeTransaction(string nameA, string nameB, string what, int number)**

**可以写死nameA，防止用别人账户给钱**

nameA因为what给了nameB number积分，

会同时更新chainTable表中的两个账户余额，还有historyTable中插入两条记录，nameA扣分，nameB加分

nameA的余额是否足够，不足够将不操作，并输出日志：

emit transactionResult("transaction\_OK");

emit transactionResult("transaction\_false");

**shouxin(string qiye, int number)**

输入企业的名字qiye和number，将同时更新chainTable和historyTable，qiye的积分将增加

日志：emit shouxinResult("授信完成");

**chengdui(string qiye, int number)**

输入企业的名字qiye和number，将同时更新chainTable和historyTable，qiye的积分将减少

qiye的余额是否足够，不足够将不操作，并输出日志：

emit chengduiResult("承兑成功");

emit chengduiResult("承兑失败");

**update(string name, int sum) 建议直接使用makeTransaction接口即可**

更新chainTable中的账户余额

emit UpdateResult("Update\_OK");

**insertHistory(string name, string what, int number) 建议直接使用makeTransaction接口即可**

输入

emit InsertResult("InsertHistory\_OK");