设计一个多线程程序，模拟银行业务办理。需求定义如下：

1）模拟客户可以随机办理银行提供的8种业务中的一种；期中办理时间规划为：（基准时间每个同学自行指定）

|  |  |  |
| --- | --- | --- |
| **业务** | **业务序号** | **办理时间范围** |
| 存款 | 1 | 0.5\*基准时间---1.5\*基准时间 |
| 取款 | 2 | 0.5\*基准时间---1.5\*基准时间 |
| 缴纳罚款 | 3 | 1.2\*基准时间---2\*基准时间 |
| 开通网银 | 4 | 5\*基准时间---8\*基准时间 |
| 交水电费 | 5 | 1.5\*基准时间---2\*基准时间 |
| 购买基金 | 6 | 2\*基准时间---3\*基准时间 |
| 转账汇款 | 7 | 3\*基准时间---4\*基准时间 |
| 个贷还款 | 8 | 2\*基准时间---4\*基准时间 |

2）程序模拟多个窗口服务，窗口服务具有一定的差异化，现在窗口的开放情况如下；

|  |  |  |
| --- | --- | --- |
| **窗口类型** | **能办理业务类型** | **说明** |
| A类窗口 | 1,2,3,4,5,6,7,8 | 所有客户可以办理 |
| B类窗口 | 1,2,4,5,7 | 所有客户可以办理 |
| V类窗口 | 1,2,3,4,5,6,7,8 | VIP客户优先 |

系统模拟窗口开放情况：

A类：1个

B类：2个

V类：1个

3）顾客模拟要求：

3.1）顾客分为普通客户和VIP客户，普通用户和VIP客户比例自行确定；

3.2）顾客到达随机模拟；

4）模拟程序模拟一天的营业情况，在完成当天的营业（模拟为上午8:00-下午17:00）后打印出客户列表，包括以下项目：

（客户名称，客户到达时间，客户办理业务类型，客户所用时间）

5）统计出以下数据：

5.1当天的所有顾客平均办理时间（从到达到办理完成时间定义为顾客办理时间）;

5.2不同业务在所有办理业务中所占的比例

6）仿真（\*需要评定优秀的同学需要提供此部分实现）：

6.1)程序能够参数化设定当天到达客户办理不同业务的比例（比如业务1占20%等）

6.2)程序能够模拟回答以下问题：

如果基准日开放的窗口情况为：

A类：1个

B类：2个

V类：1个

那么针对营业日B的情况，在相同顾客数量的情况

下，要如何开发窗口才能得到和基准日基本一致的

顾客平均服务时间；

|  |  |  |
| --- | --- | --- |
| **业务** | **基准日A** | **营业日B** |
| 存款 | 20% | 10% |
| 取款 | 20% | 10% |
| 缴纳罚款 | 10% | 5% |
| 开通网银 | 10% | 5% |
| 交水电费 | 5% | 5% |
| 购买基金 | 15% | 40% |
| 转账汇款 | 10% | 5% |
| 个贷还款 | 10% | 20% |

基本要求：

1）编码规范；

2）系统是多线程实现，并合理的进行了线程的互斥和同步设计；

3）业务类型使用Enum类型

4）程序中实现得各种存储都使用集合，不能使用数组；

设计核心问题：

1）程序整体结构设计

2）营业过程管理（开门，关门，营业分配过程）

3）用户到达模拟实现；

4）窗口服务分配实现；

5）仿真求解方法（考评的重点不是答案，而是为了求解答案而进行的设计方法）；

## **一、业务需求**

1．银行内有4个业务窗口：1个普通窗口、2个快速窗口、1个会员窗口

2．有三种对应类型的客户：会员客户、普通客户、快速客户（办理最常见5种业务如个人取款、交水电费业务的客户）

3．异步随机生成各种类型的客户，通过控制线程执行时间来确定各类型用户的比例为：

会员客户 ：普通客户 ：快速客户 = 1 ：2 ：4

4．客户办理每种业务所需时间有最大值和最小值限制，在该范围内随机设定每个客户办理业务所需的时间

5．各类型客户在其对应窗口按顺序依次办理业务。

6．当会员窗口没有客户等待办理业务的时候，会员窗口可以处理普通客户的业务；当普通窗口没有客户等待办理业务的时候，普通矿口可以处理快速客户的业务；一旦有对应的客户等待办理业务的时候，则优先处理对应客户的业务。

7. 模拟银行开门、关门

## **二、需求分析**

1．有三种对应类型的客户：会员客户、普通用户、快速用户，异步随机生成各种类型的用户。各个类型的用户在其对应的窗口办理相关的业务。

（1）每一个客户其实就是由银行的一个取号机器产生号码的方式来表示的。所以，应该要有一个号码管理器对象，让这个对象不断地产生号码，就等于随机生成了客户。

（2）由于有三类客户，每类客户的号码编排都是完全独立的，所以系统应该一共要产生三个号码管理器对象，各自管理一类用户的排队号码。这三个号码管理器对象统一由一个号码机器进行管理，这个号码机器在整个系统中始终只能有一个，所以要被设计成单例模式。

2．各类用户在对应窗口按顺序依次办理业务，服务窗口每次找号码管理器获取当前要被服务的号码。

## **三、系统执行流程**

1．银行部分

（1）初始化（包括基准时间）

（2）开启四个窗口线程

（3）等待开门时间开门

（4）开启窗口服务

（5）等待关门时间，期间窗口给对应客户提供相对应的服务

（6）到达关门时间关门

（7）停止各个窗口服务以及号码生成

（8）关门结账，打印各个窗口的业务以及计算相关数据

2．客户部分

（1）号码机取号

（2）等待服务

## **四、代码分析**

**1．NumberGenerator类：**

（1）号码生成器类，为号码机使用，生成一系列递增的号码。

定义一个List<Integer>集合存储生成的号码

定义一个List<String>集合存储客户的名字

定义一个List<LocalTime>集合存储客户到达的时间

定义一个List<BusinessType>集合存储客户办理业务的类型

上述集合记录的数据供客户线程和窗口服务线程使用。

（2）定义一个生成号码的方法generateServiceNumber（）和获取号码的方法getServiceNumber（）

（3）客户线程需要通过generateServiceNumber（）来访问lastNumber获得号码；窗口服务线程也需要通过getServiceNumber（）访问lastNumber。两个不同的线程要访问相同的共享数据lastNumber，因此两个方法都需要同步机制。

**2．NumberMachine类：**

（1）号码机类，为每个用户生成唯一的序号。

定义三个NumberGenerator类的变量，各自持有一个号码生成器对象，分别为普通客户、快速客户和会员客户进行排号。

（2）号码机在整个系统中只能有一个，所以要设成单例模式，单例模式是指私有的构造函数，保证该函数只能在这个类中调用。这个类之外的任何类都不能调用它，因为是私有的、不可见。

（3）对外提供一个public的静态方法，以供其它类来获得这个单例对象。就是说，任何其它类想要得到该类的对象，只能通过这个方法得到。

**3．ServiceWindow类**

（1）服务窗口类，对客户提供各种业务办理

定义一个独立的线程callNumber负责叫号（执行客户服务），根据办理业务类型把客户交给不同的服务方法进行办理，定义3个服务方法commonService、expressService和vipService分别为普通客户、快速客户和VIP客户服务。

（2）服务方法流程：

1）从号码机取号，如果有服务号码，就开始服务

2）取号时，同时取得客户姓名、客户到达时间、办理业务类型等信息

3）（模拟）确定为普通客户服务的时间(办理消耗时间)，这个时间是随机产生的，具体算法是：每种业务的最小服务时间加上一个随机整数

随机整数来自于该类业务的最大服务时间减去最小服务时间的差值范围。例如：存款服务的办理时间范围为0.5\*2至1.5\*2，即在1和3分钟之间，3-1=2，那么随机整数为new Random().nextInt(2)，这只能产生0,1两个值，少个2，因此把随机数上限加1，即new Random().nextInt(3)。

每种业务的最大服务时间和最小服务时间取自于服务时间类ServiceTime，该类存储了银行的开门和关门时间，以及办理每笔各类业务的时间范围。该类可以根据输入的基准时间，动态计算出办理每笔各类业务的时间范围。

4）记录开始办理的时间。

5）（模拟）使用Thread.sleep(办理消耗时间)来体现为客户服务的过程

6）计算客户从达到银行一直到业务办理完毕的所用时间，公式为：

（开始办理时间 - 客户达到时间）+ 办理消耗时间（分钟）

7）把各种客户信息组装到客户记录对象BusinessRecord对象中，然后放入customerRecordList集合中保存下来，供结束营业后统计报表使用

8）把业务笔数保存到businessNumberMap中，，供结束营业后统计报表使用

9）打印服务完成信息

10）回到第1步继续

如果没有从号码机取得号码：

1）普通客户调用快速客服服务方法，VIP客户服务方法直接调用普通客户服务方法，为其它类型的客户服务，即VIP和普通窗口空闲时，则可为快速客户服务

2）快速客户服务方法什么也不做

（3）定义打印报表的方法printCustomerReport

服务窗口每天营业结束后，都要进行扎帐工作，打印报表。该方法打印窗口的日工作量报表，内容包括客户名称、客户到达时间、客户办理业务类型、客户所用时间(分钟)。数据来自于集合customerRecordList，服务方法负责向集合中放入数据。

（4）定义返回服务的客户数量的方法getCustomerCount以及返回所有客户的办理时间总和的方法getCustomerTotalTime，统计当天的所有顾客平均办理时间时使用

（5）定义保存业务笔数的方法addToMap以及返回业务笔数集合的方法getBusinessNumber，当打印不同业务在所有办理业务中所占的比例时使用.

**4．CustomerRecord类**

客户记录类，服务窗口线程用来保存服务过的客户信息，用于报表处理

/\*\*

\* @author Anchovy

\* @date 2020/11/18 16:22

\* 操作过的用户信息记录，服务于生成记录表单

\*/

public class CustomerRecord {

//顾客名字

private String customerName;

//顾客到达时间

private String arrivingTime;

//顾客类型

private String businessType;

//用的时间

private float timeUsing;

//总共时间（从到达到办理业务完毕所用时间）

private float totalTime;

**5．BusinessType类**

/\*\*

\* @author Anchovy

\* @date 2020/11/18 16:06

\* 枚举类型，8种类型的业务

\*/

public enum BusinessType {

DEPOSIT,WITHDRAW,PAY\_PANALTY,OPEN\_ONLINEBANK,PAY\_UTILITIES,BUY\_FOUD,TRANSFER,LOAN\_PAYMENT;

@Override

public String toString() {

String name = null;

switch (this) {

case DEPOSIT :

name = "个人存款";

break;

case WITHDRAW :

name = "个人取款";

break;

case PAY\_PANALTY :

name = "缴纳罚款";

break;

case OPEN\_ONLINEBANK :

name = "开通网银" ;

break;

case PAY\_UTILITIES :

name = "交水电费" ;

break;

case BUY\_FOUD :

name = "购买基金" ;

break;

case TRANSFER :

name = "转账汇款" ;

break;

case LOAN\_PAYMENT :

name = "个贷还款" ;

break;

}

return name;

}

}

**6．CustomerType类**

/\*\*

\* @author Anchovy

\* @date 2020/11/18 16:01

\* 枚举类型，3种类型的客户

\*/

public enum CustomerType {

COMMON,QUICK,VIP;

@Override

public String toString(){

String name = null;

switch (this){

case COMMON:

name = "普通";

break;

case QUICK:

name = "快速";

break;

case VIP:

name = "会员";

break;

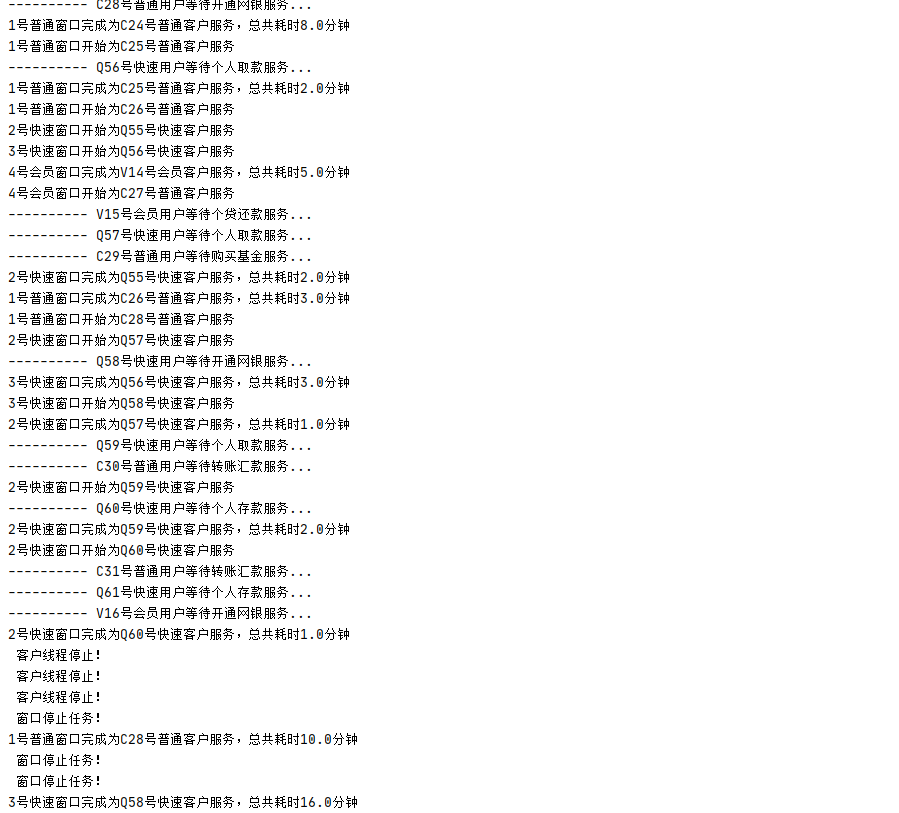
}

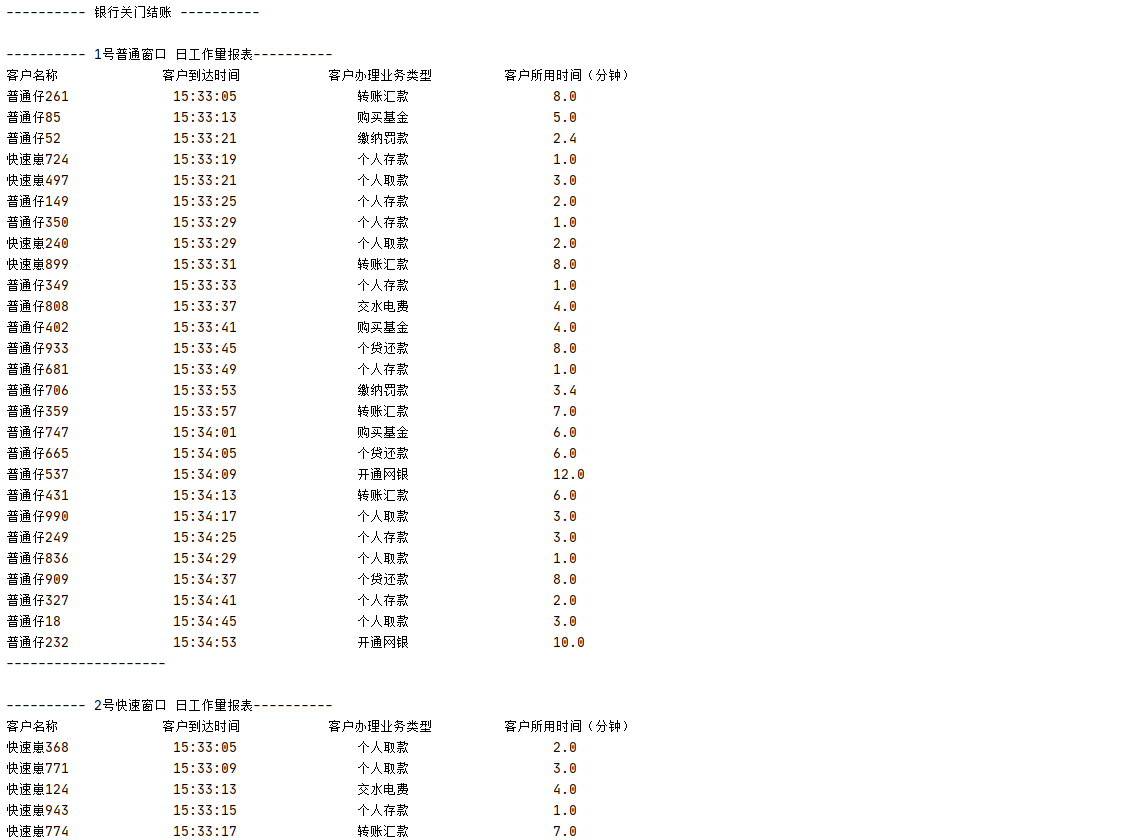
return name;

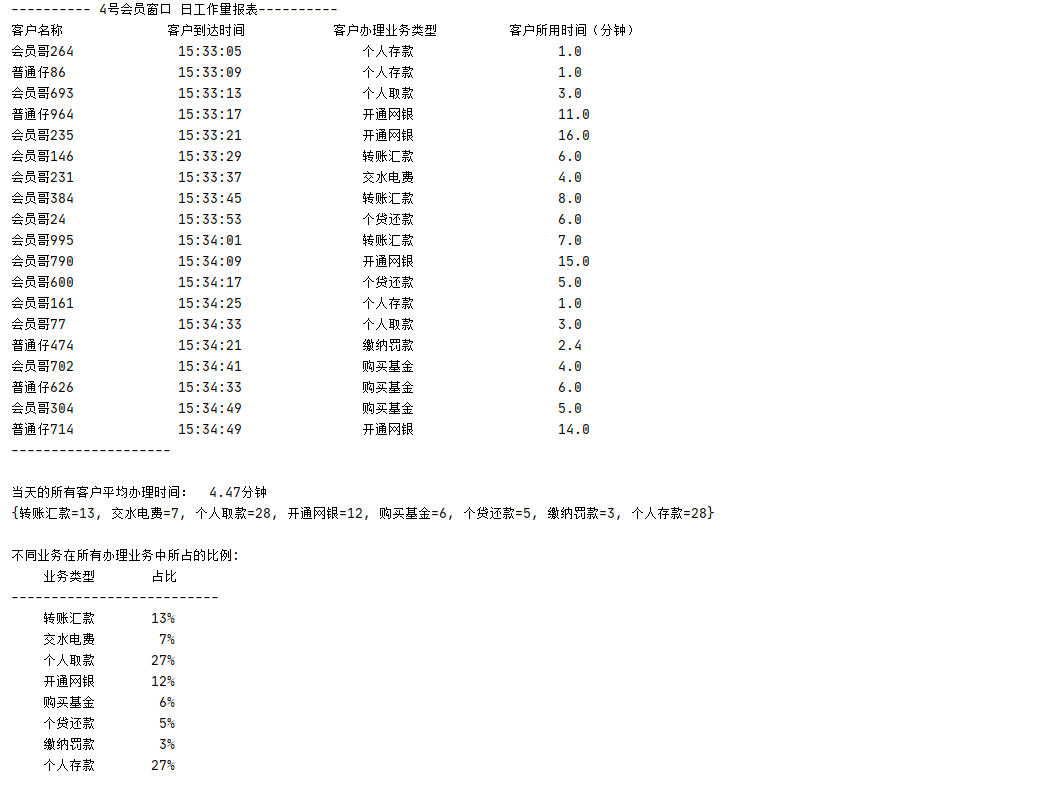
}

}

## **五、运行截图**







## **六、技术要点**

1、List、HashMap集合的使用

2、enum类型的使用

3、synchronized关键字的使用

4、java.util.Random类生成随机数

5、合并任意多个HashMap为一个：键不存在则插入新键值，键存在则累加相应的值

6、JDK1.8 LocalTime时间类的使用

7、JDK1.8 java.util.concurrent.Executors类创建线程，线程的启动和停止，相比Thread类，使用更方便

8、类的单例模式设计