在上一章中对在线考试系统进行需求分析，确定了整个系统的架构。本章将介绍系统的详细设计，包括数据库的详细设计以及一些关键模块的详细设计。

## 数据库详细设计

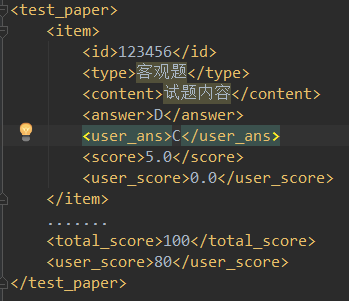
通过第三章对系统的整体进行设计和数据库的需求分析，我们确定采用MySQL数据库存储本系统涉及到的表格。管理员信息表 ADMIN\_TABLE，教师信息表 TEACHER\_TABLE，学生信息表 STU\_TABLE，知识点表KPOINT\_TABLE，题目表 QUESTION\_TABLE，试卷表TESTPAPER\_TABLE，试卷模板表 PAPERMODULE\_TABLE，课程表SUBJECT\_TABLE。由于篇幅的限制，下面只给出一些关键表的详细结构。

1. 题目表QUESTION\_TABLE、知识点KPOINT\_TABLE，课程表SUBJECT\_TABLE这三个表格都是由教师进行维护，每个题目都有所属课程的ID以及所要考察知识点的ID。表XXX是题目表的详细字段，每个题目都有唯一的标识ID，同时记录了录入该题目的创建者ID和录入时间，以及用于描述题目的一些字段，例如难度、内容、类型等。创建者在录入一个题目的同时，需要指定该题目的正确答案。对于客观题，题目答案中只需要直接给出答案内容，而对于客观题，需要给出测试数据以及标准代码。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| ID | BIGINT | 50 | 否 | 题目ID号 |
| CREATORID | BIGINT | 50 | 否 | 创建者ID |
| CREATEDATE | DATE |  | 否 | 创建时间 |
| DIFFICULITY | VARCHAR | 10 | 否 | 题目难度 |
| CONTENT | VARCHAR | 5000 | 否 | 题目内容 |
| TYPE | VARCHAR | 5 | 否 | 题目类型 |
| KPOINTID | BIGINT | 50 | 否 | 知识点ID |
| SUBJECTID | BIGINT | 20 | 否 | 课程ID |
| ANSWER | VARCHAR | 5000 | 否 | 题目答案 |
| SCORE | VARCHAR | 50 | 否 | 参考分值 |

1. 试卷表 TESTPAPER\_TABLE用于存放考卷信息，是根据预先设定的考试模板生成的，其中MODULEID为对应试卷模板ID号，ISCHECK表征试卷是否已经打分，其中TEST\_RESULT和TEST\_CONETNT字段采用XML的格式存储考卷内容以及考生答题后的试卷，格式如下。。所示。<test\_paper></test\_paper>为整个试卷，<item></item>用于标识一道题目，其中id对应的题目表中的ID号，type为题目的类型，content为题干，answer为题目的答案而user\_ans为考生给出的答案，score为改题目的分值，user\_score为考生得分。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段长度 | 是否为空 | 描述 |
| ID | VARCHAR | 50 | 否 | 试卷ID号 |
| TITLE | VARCHAR | 50 | 否 | 试卷名称 |
| STATE | VARCHAR | 10 | 否 | 是否有效 |
| SUBJECTID | VARCHAR | 20 | 否 | 课程ID |
| MODULEID | VARCHAR | 20 | 否 | 试卷模板ID |
| STUDENTID | VARCHAR | 20 | 否 | 考生ID |
| TEST\_RESULT | VARCHAR | 100000 | 否 | 提交试卷内容 |
| ISCHECK | VARCHAR | 10 | 是 | 是否已经评阅 |
| SOCRE | VARCHAR | 20 | 否 | 考试成绩 |
| PAPERREVIEW | VARCHAR | 1000 | 是 | 评阅详情 |



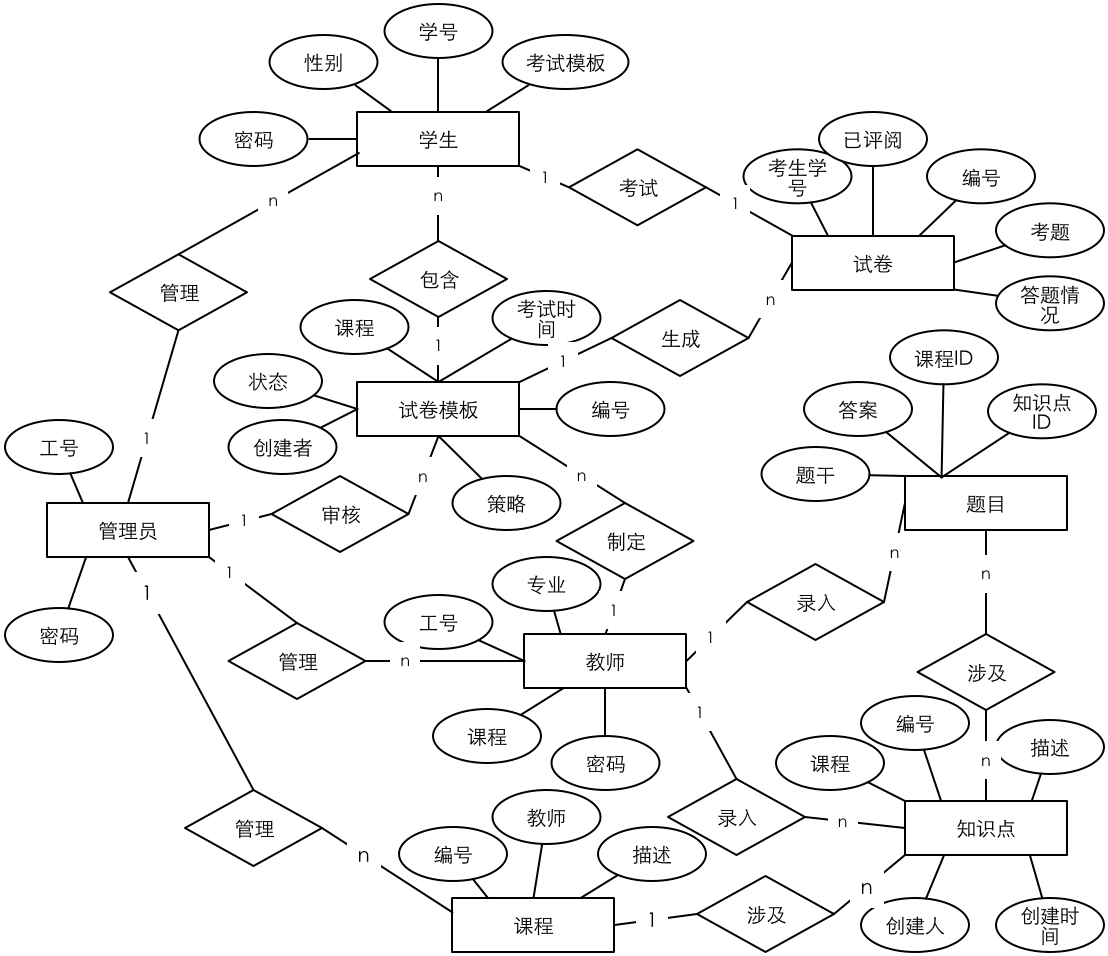
1. 试卷模板表 。教师制定的生成试卷的策略，POLICYCONTENT为组卷策略描述，仍采用XML格式进行描述。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段长度 | 是否为空 | 描述 |
| ID | VARCHAR | 50 | 否 | 模板ID号 |
| CREATORID | VARCHAR | 100 | 否 | 创建者ID |
| CREATEDATE | DATE | 7 | 否 | 创建时间 |
| TITLE | VARCHAR | 50 | 否 | 试卷名称 |
| STATE | VARCHAR | 10 | 否 | 是否审核通过 |
| SUBJECTID | VARCHAR | 20 | 否 | 课程ID |
| POLICYCONTENT | VARCHAR | 500 | 是 | 组卷策略 |

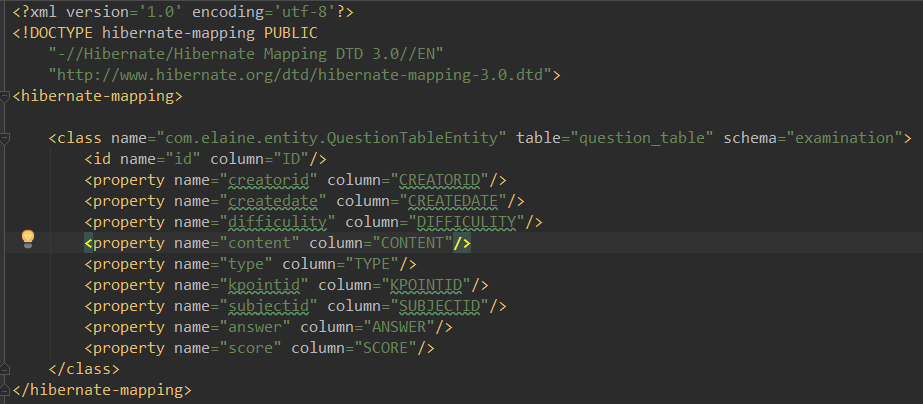
4.学生信息表，由学生自己通过学号注册或者管理员代为注册，并设定在线考试系统的登录密码。其中EXAM\_MODULE默认为null，只有当教师或者管理员将对应考试模板ID号添加到该字段，考生才能够参加对应的考试。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 字段长度 | 是否为空 | 描述 |
| ID | VARCHAR | 50 | 否 | 考生ID号 |
| PWD | VARCHAR | 50 | 是 | 密码 |
| NAME | VARCHAR | 50 | 否 | 考生姓名 |
| MAJOR | VARCHAR | 50 | 否 | 专业 |
| GRADE | VARCHAR | 50 | 否 | 年级 |
| GENDER | VARCHAR | 10 | 否 | 性别 |
| BIRTHDAY | DATE | 7 | 是 | 出生日期 |
| ID\_NUMBER | VARCHAR | 20 | 是 | 身份证号 |
| EMAIL | VARCHAR | 50 | 是 | 电子邮箱 |
| PHONE\_NO | VARCHAR | 50 | 是 | 联系电话 |
| PHOTO | VARCHAR | 200 | 否 | 电子照片地址 |
| EXAM\_MODULE | VARCHAR | 50 | 是 | 考试模板ID |

上面介绍了主要的数据表，为了更加全面的了解整体的数据结构，我们采用实体-关系图(Entity-Relationship Diagram)的方式进行描述，如。。所示。



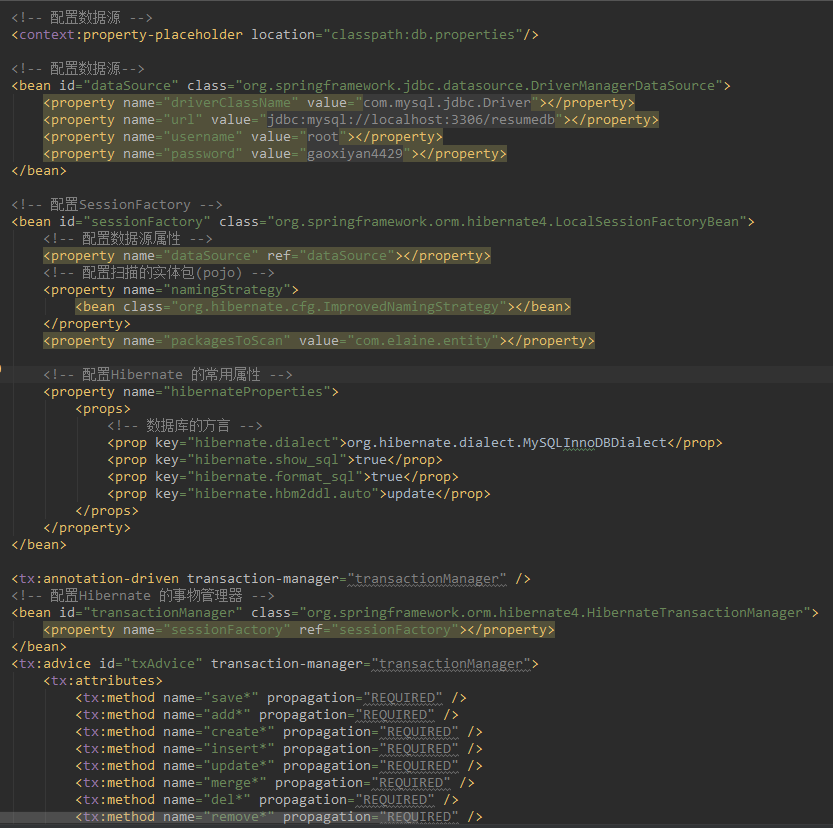
本文采用内Hibernate实现数据层的持久化，建立实体类与数据表之间的映射关系，从而避免开发人员采用JDBC的方式直接操作数据库，从而保证整个系统遵从面向对象的方式进行开发。Hibernate通过hbm.xml格式的文件将POLO类映射到数据库的表格中，不需要编写任何额外的代码。IntelliJ IDEA支持将数据库中的表格直接转化为POLO类，大大减少了开发的工作量。POLO类对应数据库表格，每个POLO类的实例对应数据库中的一行记录，POLO类的属性对应数据库表的字段。图。。就是QuestionEntity与QUESTION\_TABLE的映射文件question.hbm.xml。这个文件通过class name与table建立联系，并将POJO类的属性一一与table的字段对应，包括名称以及数据类型。之后对数据库中数据的操作就可以简单通过操作POJO类来实现。



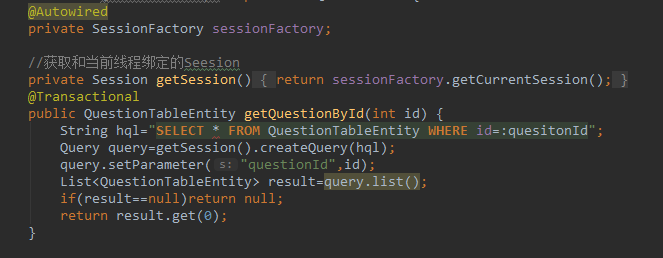
单纯使用Hibernate进行事务处理，每次对数据的操作都需要经历8个步骤。首先需要获取Configuration对象，接下来通过Configuration对象创建sessionFactory对象，并通过sessionFactory对象创建session，然后通过session开启事务，进行一些数据持久化操作，处理结束后需要提交事务，如果发生异常需要进行事务回滚，最终关闭session，对应代码如下：

**public void** getPersistence(){  
 Configuration configuration=**new** Configuration().configure();  
 SessionFactory sessionFactory=configuration.buildSessionFactory();  
 Session session=**null**;  
 **try**{  
 session=sessionFactory.openSession();  
 session.beginTransaction();  
 *//数据持久化操作* ......  
 session.getTransaction().commit();  
 }**catch**(Exception e){  
 e.printStackTrace();  
 *//进行事务回滚* session.getTransaction().rollback();  
 }**finally**{  
 **if**(session!=**null**){  
 **if**(session.isOpen()){  
 session.close()  
 }  
 }  
 }  
}

而用Spring对Hibernate进行管理时会大大简化这个步骤，Spring提供了HibernateTemplate类，这个类对Session等进行了封装，并且会自动提交事务，不需要开发者手动开启事务和提交事务，只需要将对象进行save操作即可。图X是Spring对Hibernate的配置。



在Spring配置文件中加入Hibernate相关的配置之后，对数据持久化的操作如下图所示。只需要在使用sessionFactory之前对其进行类型声明，后续可以通过getCurrentSession得到session对象，然后进行数据库的相关操作，Spring会进行后续的处理。



## 系统功能模块详细设计

### 系统登录详细设计

系统登录是整个系统的入口，用户需要输入用户名和密码，点击登录按钮提交。如果提交的信息没有验证通过，则会提示用户名或者密码错误，用户需要重新输入用户名和密码；否则系统会根据用户的账号判定用户的身份，并跳转到对应的系统页面，详细流程如。。。所示。



### 学生在线考试详细设计

学生在线考试是整个系统的核心模块，考生在登录考试系统后，需要点击在线考试选项，系统会判断该学生是否具备考试资格。如果学生不具备考试资格，则直接退出；如果具备考试资格，则系统会根据试卷模板生成试卷并开始计时。如果在考试过程中，考生主动点击提交按钮，考试开始之后半个小时才能够提交试卷，否则继续答题。另外，到达考试时限后，系统会自动提交试卷结束考试，流程如。。所示。



### 教师管理部分详细设计

## 系统关键模块设计

## 自动组卷

目前最常用的是基于随机抽题法的自动组卷，通过随机函数不断从题库中抽取试题直到组卷完成或者无法在试题库中找到满足条件的试题。这种算法时间复杂度较高，并且在组卷约束条件较多的情况下，经常会出现组卷失败。因此该种算法只适用于题库比较少并且组卷相对简单的场景。基于遗传算法的组卷算法是近几年刚发展起来的，该种方法的组卷效果很好，但是算法复杂度较高，运行速度慢，因此本系统选择在传统的随机抽题法的基础上进行改进，根据知识点、难度等条件，从题库中抽取出满足这些条件的试题，然后随机从这些题目中抽取试题进行组卷，最后存储到试卷库中。考试模板中会针对每一道题设定考察的知识点和难度，组卷时需要从取出其中一道题的要求，也就是考察的知识点和难度，然后从题库中找出满足条件的试题，并进行组卷，对应流程图如。。。所示。



### 自动评阅试卷

计算机语言类课程的考试题目可以分为两种类型客观题和程序设计题，其中客观题的批改比较简单，只需要将考生提交的答案与标准答案进行比对，如果相同则改题满分，否则为0分。而程序设计类题目的批改过程比较复杂。

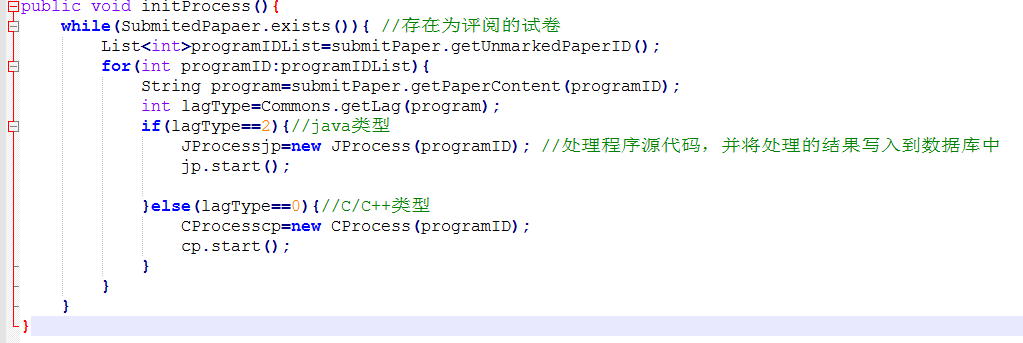
自从上世纪60年代以来，许多学者致力于研究基于文本答案的主观题自动批改技术，并设计开发出了多种自动阅卷系统：Beierle等人【Beierle C, Kulaa M, Widera M. Automatic Analysis of Programming Assignments[C]//DeLFI. 2003: 144-153.】提出了一种针对Prolog语言的课程作业的评阅方式AT(x), 通过分析学生提交的程序然后将评阅结果通过Web的方式反馈给学生。Singh[Singh R, Gulwani S, Solarlezama A, et al. Automated feedback generation for introductory programming assignments[J]. programming language design and implementation, 2013, 48(6): 15-26.] 等人开发的系统通过一个现有的常见错误表对学生提交程序中的简单小错误进行简单的修正。Jinrong[Li J, Pan W, Zhang R, et al. Design and implementation of semantic matching based automatic scoring system for C programming language[C]. international conference on e learning and games, 2010: 247-257.]等人则提出了一种针对语法和语义匹配的方式，通过将学生提交的程序与标准程序进行语义匹配来计算得分。Shamsi和Elnagar等人提出了一种针对java语言的评阅系统，该系统结合了Junit 框架和程序图表示两种方式，综合计算学生得分。以上这些方法适用于人数较少的场景并且只针对一种特定的计算机语言，同时采用的算法相对比较复杂，因此本文将根据计算机语言的特征采用一种相对比较简答的评阅方法。相比于自然语言，计算机语言具有以下特征【Li J, Pan W, Zhang R, et al. Design and implementation of semantic matching based automatic scoring system for C programming language[J]. Entertainment for Education. Digital Techniques and Systems, 2010: 247-257.】：

1. 程序的基本格式固定，语法和词法的约束都比较严格，格式相对比较统一。
2. 计算机语言考试的对应程序代码一般都比较短小，一般不超过40行。
3. 对于特定的题目，程序的整体框架大体相同。

目前针对多种语言的程序设计自动阅卷方法都是通过运行学生提交的程序，然后将程序的输出与标准答案进行比对，根据比较的结果得出相应的分数。这种简单通过比对输出结果来评判分数的方式显然是充分的，因此本文提出了一种综合程序输出和程序理解的方法如图。。所示，在某些情况下通过程序理解的方式来纠正通过程序输出计算得分的方式，使得最终的结果更加的公平、合理。该模块首先对读取数据库中未评阅试卷中的程序设计题，并对学生提交的程序进行编译，如果编译通过则运行程序，将测试用例文件作为程序的输入。如果程序运行超时或者出现异常，则将该程序与标准程序进行匹配计算得分。如果程序正常运行并且程序输出与标准输出的匹配率大于0，则按照测试用例通过的比例计算得分。程序未通过编译或者运行超时、运行出错等，都将进入程序匹配阶段。



在本系统中，学生提交试卷后，系统会直接读取学生提交的试卷答题信息，编译运行检查结果并将结果写入到数据库中。在编程部分的试题进行评阅时，需要调用服务器上安装的编译器来编译考生提交的源代码，并保证在编译提交代码发生问题后，系统不会出现致命错误。为了达到该目的，需要引入一个守护线程，负责启动评阅试卷的线程，每个线程负责启动对应语言的编译器对程序进行编译和运行。守护线程负责每个一段时间从数据库中检查是否有未评阅的试卷，如果有则启动线程对提交的结果进行评阅。守护线程的关键代码如下图所示：



如果源程序编译之后能够运行并产生结果，则将产生的结果与标准答案进行对比。若与标准答案完全一致，则满分；若部分正确，则按照正确的比例给分；若得分为0，则将学生提交的代码与标准代码进行比对评分；如果程序未通过编译，无法产生运行结果，则转入程序修复。

**C/C++评判线程**

评判线程首先启动GCC编译器对源代码进行编译，然后指定测试数据文件作为编译后.exe文件的输入，并得到运行结果。线程启动之后，首先调用compile函数对源代码进行编译，如果编译通过，则调用编译后的.exe文件并指定测试文件作为输入，运行结束之后将输出结果与标准结果进行比较，并按照测试数据通过的比例计算得分。如果分数为0或者程序编译未通过，则通过比较源代码与标准代码的相似度，根据相似度计算最终得分并在标准分数的基础上减去部分分数。线程对应的run()函数流程图如。。。所示。



run()函数的关键代码如下：



**JAVA评判线程**

JDK是java语言的软件开发工具包，包括了java语言运行的环境配置以及常用的API。JDK支持各种操作系统，安装之后配置环境变量即可使用。Java评判线程会启动Javac编译器对源代码进行编译生成Java字节码，然后由java虚拟机执行Java字节码，并将指定的测试文件作为输入，并得到程序运行结果。与C/C++进行程序编译的过程类似，JProcess启动之后，首先调用compile()函数对源代码进行编译，如果编译通过，调用execute()函数执行程序，并将测试文件作为输入，并将程序的输出结果与标准结果进行比对，按照正确的比例给分。如果测试结果正确率为0或者程序未通过编译，则通过计算提交程序与标准程序的相似度来给分，具体流程图如。。所示。

**程序相似度匹配**

**算法思想：**

假定学生提交的程序代码为S，标准程序代码为P。

1. 采用词法分析进行程序预处理

程序展示方式是多种多样的，变量名称的不同或者变量定义的顺序不同都将导致程序的表示方式不同，因此在进行程序匹配之前，需要对其进行归一化处理。本文采用词法分析的方式过滤掉注释等无关信息，并将程序映射为一个token串。其中不同的关键字对应不同的token，变量名等无关信息映射为相同的token，最终将程序转为token串。

1. token串分割

将token串分割成一个个token子串，每个token子串包含m个token，相邻两个token子串之间有m-1个token是重复的。

其中len(S)为token串S中包含token的个数，为第i个包含m个token的子串。

1. 计算token子串的哈希值

对token子串计算哈希值的目的就是将相似的子串尽可能映射到比较相近的值，而相差较远的token子串的哈希值也应该相差较大。另外一方面，将子串映射为一个值，大大降低了空间负责度。对应的hash函数定义如下：

其中 ，为一个token，为的分值。不同关键字对应不同的分值，变量类token的分值为1，关键字的分值为5，q=5.

1. 相似度计算
2. S对应的哈希值串为，P对应的哈希值串为，相似度计算公式如下：

其中为S中token子串的个数，为哈希值相同的token的个数。

|  |
| --- |
| 算法1 GeneratePaper |
| 1: |
| 2:  3:  4: while do  5: 从题目要求的list中取出第i个要求  从数据库中随机得出满足条件的questionList        question=questionList.get(randNum)  将question加入到试卷  End if  End while |