

廈門大學



信息学院软件工程系

《JAVA 程序设计》实验报告

实验九

姓名：陈澄

学号：32420212202930

学院：信息学院

专业：软件工程专业

完成时间：2023.04.23

一、实验目的及要求

实验目的：

- 熟悉字符串及正则表达式

实验要求：

- 按照题目要求写代码和实验报告。

二、实验题目及实现过程

题目 1：编写程序完成：

输出以下新闻片段中出现的单词（每个单词只输出一次）。

输出以下新闻片段中包含 the 的句子。

PITTSBURGH (AP) — Carnegie Mellon University will hire a researcher from the Library of Congress to help it decode a collection that includes two WWII German Enigma machines.

The university wants to encourage the study of 19th and 20th century computers, calculators, encryption machines and other materials related to the history of computer science.

“When we look back and we see this, we see who we remember,” Andrew Moore, dean of CMU’s School of Computer Science, said, adding his students are increasingly asking for courses about the history of the field. “We see people who took technology to save lives and save the world.”

Pamela McCorduck, a prolific author on the history and future of artificial intelligence and the widow of Joseph Traub, a renowned computer scientist and the former head of CMU’s Computer Science Department, permanently loaned to the university a collection of early computers, books and letters. The collection, anchored by a three-rotor and four-rotor Enigma machine, is on display in the Fine and Rare Book Room in CMU’s Hunt Library in Oakland. The gift makes CMU one of a few institutions in the United States with Enigma machines. Even fewer display them.

（一）实验环境（集成开发环境、jdk 版本、字符编码等）

集成开发环境：IntelliJ IDEA

jdk 版本：17.0.5

字符编码：ASCII

（二）实现过程（**本部分为主要评分依据**，请描述解题思路，比如总共设计几个类，各个类的用途、成员、主要方法等及其之间调用关系等）

1. 新建变量 `article` 用于获得用户输入的文本，通过 `while` 循环以及 `NextLine` 获取每一行的输入 `a`，再通过 `article=article+a`；将其拼接为一个字符串。

2. 创建函数 `print_all_words` 用于获得一个字符串中所有不同的单词，实现如下：

新建一个 `Pattern` 为 `[^a-zA-Z0-9]+` 的正则表达式，用于匹配所有非字母和数字的字符，通过 `matcher` 与所给字符串相匹配，通过 `replaceAll` 将其全部替换为空格，通过 `toLowerCase()` 转化为小写，再通过 `split` 去除空格并转化为字符串数组 `words`。到此以获得字符串中所有单词。

建立一个 `String` 型的 `Set` 并通过 `addAll` 方法将上述字符串数组保存在该 `Set` 中以完成去重操作。最后建立一个 `Int` 型变量 `count` 用于计数不重复的单词数量，通过循环输出该 `Set` 即可。

3. 创建函数 `print_sentence_contains` 用于输出字符串中包含某个特定字符串的句子，实现如下：

通过 `split("\\.")` 将获得的字符串分隔为一个个句子。对于每个句子使用与上个函数相似的处理方法分隔为一个个单词（如果直接使用 `contains` 方法容易出现类似于 `these` 中包含 `the` 的情况），再通过 `Arrays.asList` 方法转化为 `list`，调用 `list` 中的 `contains` 方法查找字符串。同样使用一个 `count` 计数后输出。

（三）过程截图（**本部分为主要评分依据**，一张全屏截图（必须）、若干运行结果展示图（可选），主要代码（可选））

```

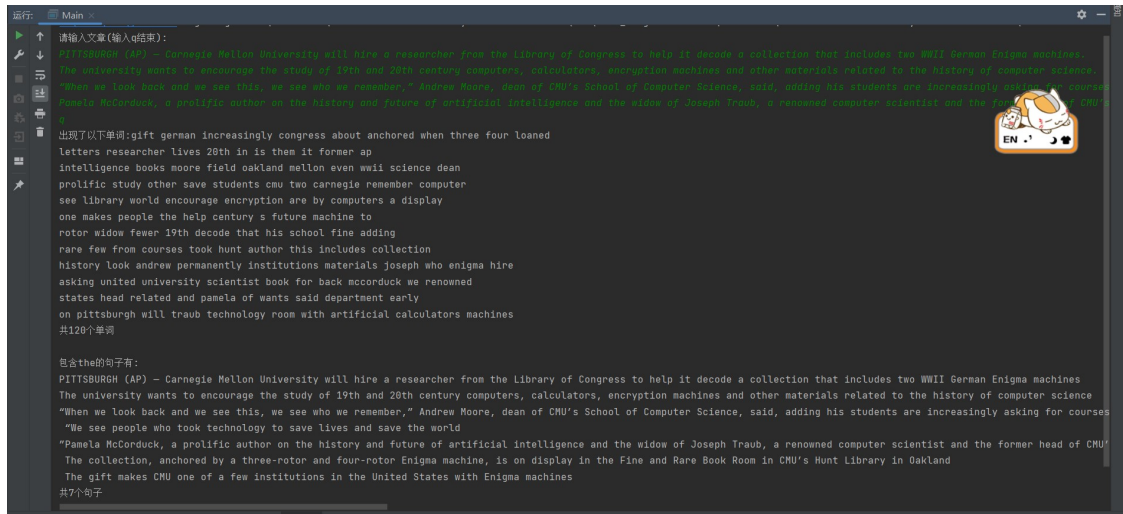
Main.java x
1  import java.util.*;
2  import java.util.regex.Pattern;
   0 个用法
3  public class Main {
   0 个用法
4  public static void main(String[] args) {
5      Scanner scanner = new Scanner(System.in);
6      String article = "";
7      System.out.println("请输入文章(输入q结束):");
8      while (true){
9          String a = scanner.nextLine();
10         if(!a.equals("q")) article=article+a;
11         else break;
12     }//获得输入
13     print_all_words(article);
14     print_sentence_contain(article, word: "the");
15 }
   1 个用法
16 static void print_all_words(String article){
17     Pattern PATTERN = Pattern.compile( regex: "[^a-zA-Z0-9]+"); //一个或多个除字母或数字以外所有符号的pattern
18     String[] words= PATTERN.matcher(article).replaceAll( replacement: " ").toLowerCase().split( regex: "\\s+"); //去掉标点符号转化为空格
19     Set<String> nonredundant_words = new HashSet<>(); //建立哈希表用于删除重复单词
20     nonredundant_words.addAll(Arrays.asList(words));
21     System.out.print("出现了以下单词:");
22     int count=0;
23     for (String s : nonredundant_words) {
24         System.out.print(s+" ");
25         count++;
26         if(count%10==0)System.out.print("\n");
27     }
28     System.out.println("共"+count+"个单词");

```

```

Main.java x
22     int count=0;
23     for (String s : nonredundant_words) {
24         System.out.print(s+" ");
25         count++;
26         if(count%10==0)System.out.print("\n");
27     }
28     System.out.println("共"+count+"个单词");
29     System.out.print("\n");
30 }
   1 个用法
31 @ static void print_sentence_contain(String s,String word){
32     Pattern PATTERN = Pattern.compile( regex: "[^a-zA-Z0-9]+");
33     String[] sentences = s.split( regex: "\\.");
34     System.out.println("包含"+word+"的句子有:");
35     int count=0;
36     for(String sentence:sentences){
37         String[] words = PATTERN.matcher(sentence).replaceAll( replacement: " ").toLowerCase().split( regex: "\\s+");
38         List<String> list = Arrays.asList(words);
39         if(list.contains(word)){
40             System.out.println(sentence);
41             count++;
42         }
43     }
44     System.out.println("共"+count+"个句子");
45 }
46 }

```



题目 2：用正则表达式对用户输入的用户名、密码、邮箱进行判断，若不满足输入要求则提示出错类型。

用户名要求：不能为空，只能由字母、数字和_组成，第一位不能为数字。

密码要求：不能为空，密码长度至少 8 位，由字母、数字、下划线组成。

邮箱要求：不能为空，需包含”@”符号。”@”符号后需要出现多个由”.”分割的词。

（一）实验环境（集成开发环境、jdk 版本、字符编码等）

集成开发环境：IntelliJ IDEA

jdk 版本：17.0.5

字符编码：ASCII

（二）实现过程（**本部分为主要评分依据**，请描述解题思路，比如总共设计几个类，各个类的用途、成员、主要方法等及其之间调用关系等）

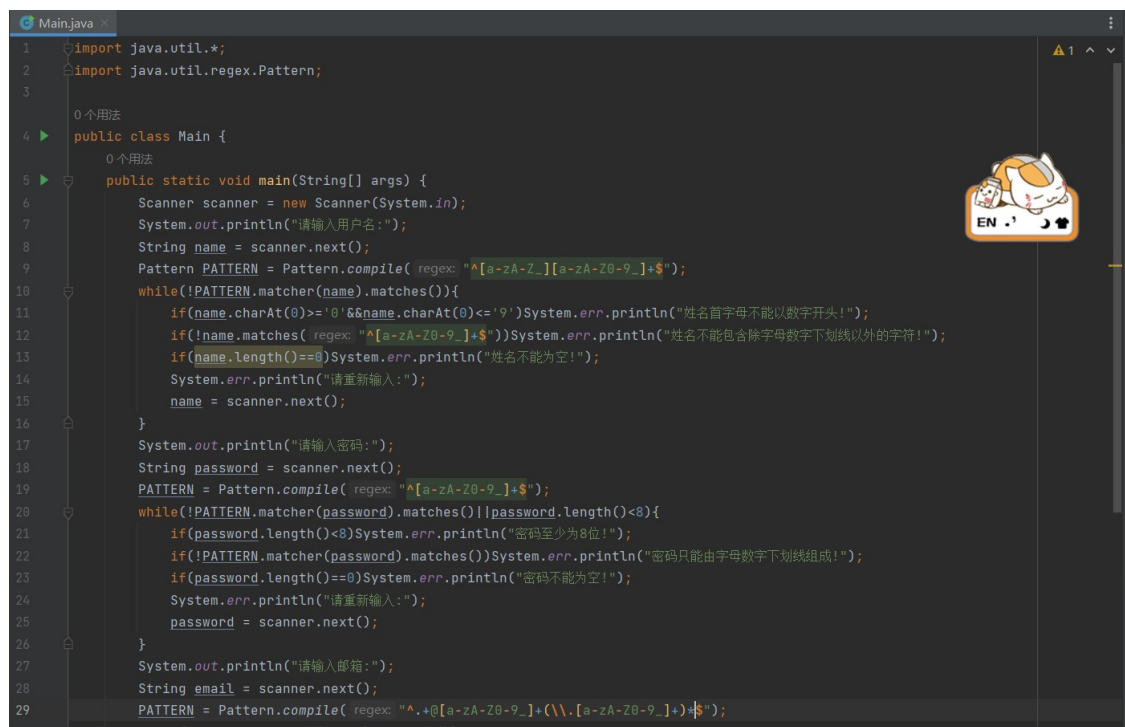
1. 姓名：通过正则表达式`^[a-zA-Z][a-zA-Z0-9_]+$`检测姓名是否合法，其中`^`代表首字符，`$`代表末字符，`[a-zA-Z]`代表首位只能为字母或下划线，`[a-zA-Z0-9_]`代表后几位可以为字母数字或下划线，`+`代表一个或多个。

若不合法则通过 while 语句重新让用户输入，其中对输入错误进行分类并反馈给用户，通过 `charAt(0)` 检测首位是否合法，通过 `^[a-zA-Z0-9_]+$` 检测姓名中是否包含其他符号，通过 `length` 检测姓名是否为空。

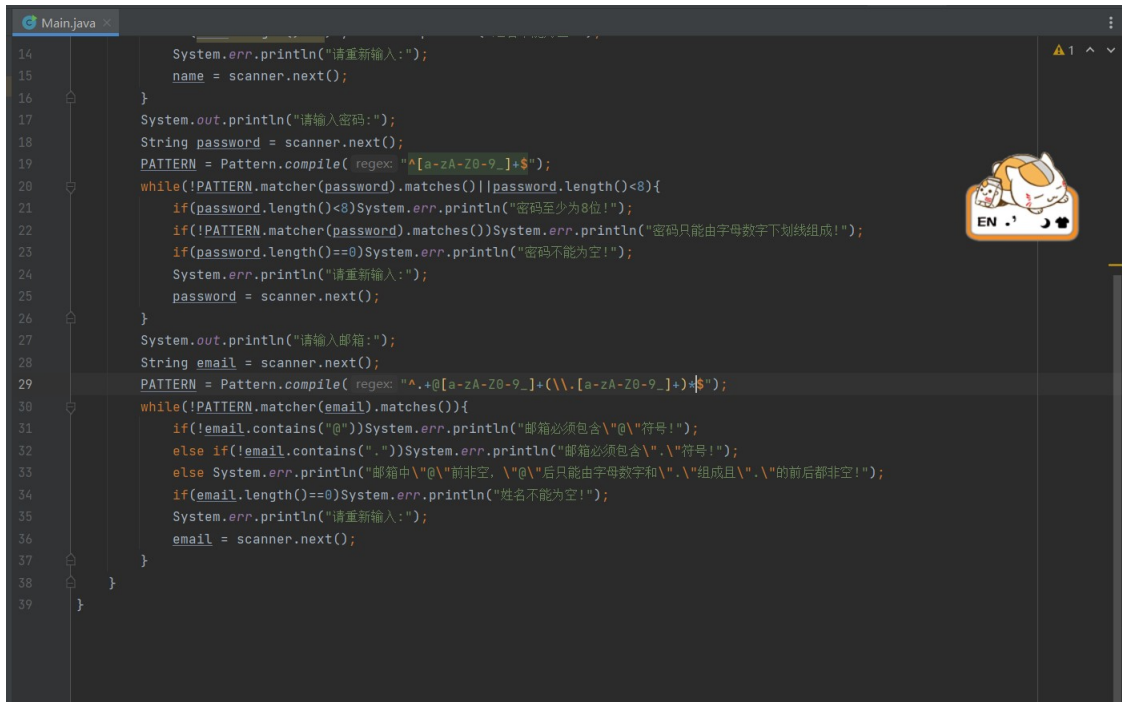
2. 密码：大致与姓名类似，正则表达式改为 `^[a-zA-Z0-9_]+$`，即只能由字母数字下划线构成，并在 while 中添加对密码长度的检测。

3. 邮箱：正则表达式为 `^.+@[a-zA-Z0-9_](\.[a-zA-Z0-9_]+)*$` 代表 @ 前面有一个或多个任意字符，@ 后面由字母数字下划线组成且 “.” 前后至少包含一个词，且词的长度非 0。

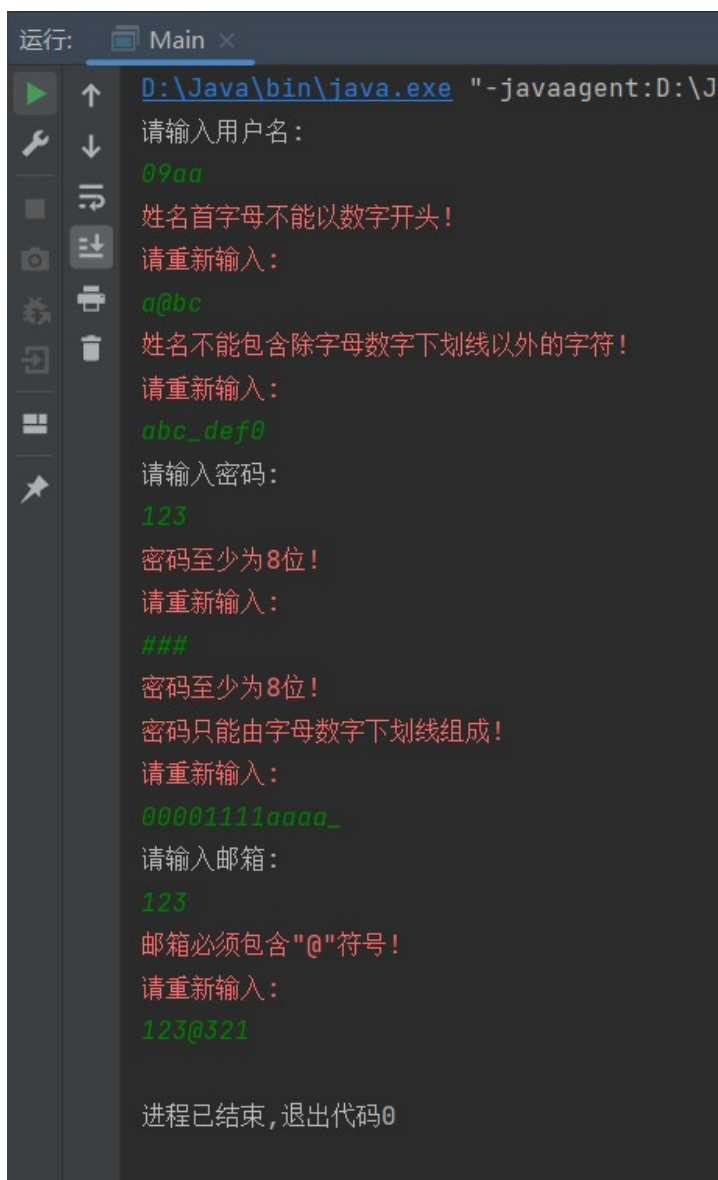
（三）过程截图（**本部分为主要评分依据**，一张全屏截图（必须）、若干运行结果展示图（可选），主要代码（可选））



```
1 import java.util.*;
2 import java.util.regex.Pattern;
3
4 public class Main {
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7         System.out.println("请输入用户名:");
8         String name = scanner.next();
9         Pattern PATTERN = Pattern.compile( regex: "^[a-zA-Z_][a-zA-Z0-9_]+$");
10        while(!PATTERN.matcher(name).matches()){
11            if(name.charAt(0)>='0' && name.charAt(0)<='9') System.err.println("姓名首字母不能以数字开头!");
12            if(!name.matches( regex: "^[a-zA-Z0-9_]+$")) System.err.println("姓名不能包含除字母数字下划线以外的字符!");
13            if(name.length()==0) System.err.println("姓名不能为空!");
14            System.err.println("请重新输入:");
15            name = scanner.next();
16        }
17        System.out.println("请输入密码:");
18        String password = scanner.next();
19        PATTERN = Pattern.compile( regex: "^[a-zA-Z0-9_]+$");
20        while(!PATTERN.matcher(password).matches() || password.length()<8){
21            if(password.length()<8) System.err.println("密码至少为8位!");
22            if(!PATTERN.matcher(password).matches()) System.err.println("密码只能由字母数字下划线组成!");
23            if(password.length()==0) System.err.println("密码不能为空!");
24            System.err.println("请重新输入:");
25            password = scanner.next();
26        }
27        System.out.println("请输入邮箱:");
28        String email = scanner.next();
29        PATTERN = Pattern.compile( regex: "^.+@[a-zA-Z0-9_](\\.[a-zA-Z0-9_]+)*$");
```



```
14      System.err.println("请重新输入:");
15      name = scanner.next();
16  }
17  System.out.println("请输入密码:");
18  String password = scanner.next();
19  PATTERN = Pattern.compile("^[a-zA-Z0-9_]+$");
20  while(!PATTERN.matcher(password).matches() || password.length() < 8){
21      if(password.length() < 8) System.err.println("密码至少为8位!");
22      if(!PATTERN.matcher(password).matches()) System.err.println("密码只能由字母数字下划线组成!");
23      if(password.length() == 0) System.err.println("密码不能为空!");
24      System.err.println("请重新输入:");
25      password = scanner.next();
26  }
27  System.out.println("请输入邮箱:");
28  String email = scanner.next();
29  PATTERN = Pattern.compile("^[a-zA-Z0-9_]+(\\.[a-zA-Z0-9_]+)*$");
30  while(!PATTERN.matcher(email).matches()){
31      if(!email.contains("@")) System.err.println("邮箱必须包含\"@\"符号!");
32      else if(!email.contains(".")) System.err.println("邮箱必须包含\".\"符号!");
33      else System.err.println("邮箱中\"@\"前非空, \"@\"后只能由字母数字和\".\"组成且\".\"的前后都非空!");
34      if(email.length() == 0) System.err.println("姓名不能为空!");
35      System.err.println("请重新输入:");
36      email = scanner.next();
37  }
38  }
39  }
```

```
运行: Main x
D:\Java\bin\java.exe "-javaagent:D:\J
请输入用户名:
09aa
姓名首字母不能以数字开头!
请重新输入:
a@bc
姓名不能包含除字母数字下划线以外的字符!
请重新输入:
abc_def0
请输入密码:
123
密码至少为8位!
请重新输入:
###
密码至少为8位!
密码只能由字母数字下划线组成!
请重新输入:
00001111aaaa_
请输入邮箱:
123
邮箱必须包含"@"符号!
请重新输入:
123@321

进程已结束,退出代码0
```

三、实验总结与心得记录

本部分根据实验过程的所得所想描述，记录可供以后复习回看 {可以记录调试过程遇到的问题，自己哪些知识点掌握不够，设计是否有缺陷（比如耗时？耗内存？）是否有亮点，是否有精妙的算法，或者设计模式的应用，可吐槽，也可与其他语言作适当对比。}（本部分不作为平时评分依据）

备注：

建议附带代码提交的方式：导出工程压缩包。

平时实验成绩以考查参与度为主，所有实验要求自己完成，一旦发现抄袭或者其他投机取巧，取消所有平时成绩