作业4

计算机程序设计简介：作业4

截止日期为2018年11月12日晚7点

目标

此作业将为您提供编写从文件获取输入并将输出写入文件的程序的经验。它还将为您提供更多使用条件和循环的练习。

资源和链接

下载zip文件并解压缩。
提交你的答案。
标记和反馈（如果可用）
用于检查程序的ClassTime程序的示例输出。

摘要
ClassTimes：
编写程序以从包含大学时间表数据的文件中提取有用信息。
反射。
写下你对这项任务的反思。

提交
您的ClassTimes.java版本
您的Reflection.txt文件

概观

该任务涉及一组练习和一个程序（ClassTimes），它从大学时间表信息文件中提取数据。

制备

下载作业4的zip文件并将其解压缩到主文件夹中的Assig4文件夹。它应包含要完成的Java程序的模板以及数据文件。仔细阅读整个作业，看看你需要做什么。

查看作业3的模型答案，并确保您了解程序的所有组件。另外，从使用文件的讲座中查看代码示例。
上课时间

大学时间表系统输出每个三个月所有课程的所有课程列表。然而，列表很长，使用它可以是繁琐的（例如找到特定课程的所有房间）到彻头彻尾的痛苦（例如，查找在特定房间中进行演讲的所有课程）。

您将完成ClassTimes计划，该计划将帮助用户更快地从（时间表的简化版本）中找到某些信息。该程序有9个方法，每个方法将从包含类时间表数据的文件中读取数据，并打印出（到文本窗格和/或文件）只是与用户相关的数据。所有方法都密切相关，并且具有非常相似的结构：遍历文件，读取每行的值并处理它们。

类时间表文件称为classdata.txt。

每门课程可能有很多不同的课程：它可能会在不同的日子在不同的时间在不同的房间进行几次讲座和/或教程和/或实验课程等。文件的每一行指定一个课程的一个课程或会话，因此每个课程可能有许多行。

此处显示了该文件的六行：

```
COMP102   Comp-Lab     Wed   1600   1700   CO219
COMP102   Comp-Lab     Wed   1600   1700   CO238
COMP102   Lecture      Mon   1100   1150   KKLT303
COMP102   Lecture      Wed   1100   1150   KKLT303
COMP102   Lecture      Fri   1100   1150   KKLT303
COMP102   Tutorial     Tue   1610   1700   AM104
```

这些行的格式如下：
第一个标记是课程代码
第二个标记是类的类型
第三个标记是一周中的某一天
第四和第五个令牌是会话的开始和结束时间（使用24小时时间）
最后一个标记是房间代码（房间代码的第一部分始终是建筑代码，例如Hugh McKenzie的“HM”。
请注意，数据基于实际课程时间表，但已经过修改和简化（并且今年不一定是最新的）。

核心
完成三种方法：
printCourse（String targetCourse）
该方法有一个参数 - 课程代码。它应该读取classdata文件，打印（到文本窗格）所有类类型，日期，开始和结束时间，以及给定课程的每个课程会话的空间（即，来自第一个令牌的所有行的数据）该行等于课程代码）。如果课程没有课程，则应打印出课程没有上课时间的消息。
它需要使用next（）读取每一行的第一个标记，然后可以使用nextLine（）读取该行的其余部分，或者分别读取该行的其余部分的每个值。您将需要使用String类中的一些方法。

printRoom（String targetRoom）
该方法有一个参数 - 房间代码。它应该打印出一个标题：“Classes in ...”并在其下面写一个虚线。然后它应该读取文件并打印出目标房间中每个课程的课程代码，课程类型，日期，开始时间和结束时间。然后它应该打印一条消息，说明房间里有多少课程，例如“KKLT302中没有预订课程”，或“VZ103中有25个课程”。

printAtStartTime（int startTime）
该方法有一个参数 - 开始时间。它应该读取课程时间表文件，打印出目标时间开始的每个课程的课程代码，房间，类型，日期和结束时间。最好分别读取每行上的六个令牌。

完成
完成三种方法。

printInRoomsOnDay（String targetRoom1，String targetRoom2，String targetDay）
该方法有三个参数 - 两个房间代码和一天。它应该打印出一个标题：例如“Tue in KK201或KK202 on Tue”并加下划线。然后它应该读取文件并打印出课程代码并开始当天目标房间中每个课程的开始时间。

bookingsFileForRoom（String targetRoom）
该方法有一个参数 - 房间代码。它应该构造一个新文件，其名称是房间代码，然后是“_Bookings.txt”[_eg_“HMLT205_Bookings.txt”]，其中包含该房间预订的类别列表。文件的第一行应指定预订的空间，例如：
预订房间HMLT205
在房间里预订的每个班级应该打印到三行的文件中，包括课程，时间和类型，然后是一个空行，_eg_
课程：ACCY130
时间：星期一1200-1250
会议：讲座

该方法应在完成文件写入后打印消息。
提示：首先使用UI.println将输出打印到UI窗口，然后将其写入文件。

inBuildingAfterTime（String targetBuilding，String targetDay，int targetStart）
该方法有三个参数 - 建筑代码，日期和开始时间。它应该打印指定建筑物中所有类（课程代码，类型和房间）的列表，并在指定的开始时间或之后开始。
请注意，每个房间名称的第一部分是大写的建筑代码。

挑战
完成两种方法。

double meanClassLength（String targetRoom）
该方法返回（不打印）指定房间中安排的所有课程的平均（平均）持续时间（以分钟为单位）。
提示：小心时间 - 它们不是普通的整数，如果房间里没有课程，也不会导致错误。

potentialDisruptions（String building，String targetDay，int targetStart，int targetEnd）
列出在给定日期在给定建筑物中具有类的所有课程（仅课程代码），使得课程的任何部分在给定时间之间。所涉及的每门课程只应列出一次，即使在该时间段内建筑物中有几个班级。请注意，数据文件按课程代码排序。
这种方法与完成问题之一类似，但有点复杂。

反馈
回答Reflection.txt文件中的以下问题，并确保提交。

ClassTimes中的一系列方法越来越棘手。 确定后者使其变得更加困难的原因。
classdata.txt中的行按课程代码排序。 如果它们被其他字段之一排序，哪些查询会更容易或更好输出？

Assignment 4

Introduction to Computer Program Design: Assignment 4

- Due 12 Nov 2018 7 pm

Goals

This assignment will give you experience in writing programs that get input from files and write output to files. It will also give you more practice using conditionals and loops.

Resources and links

- Download zip file and unzip it.
- Submit your answers.
- Marks and Feedback (When available)
- Example output of ClassTime program for checking your program.

Summary

- ClassTimes:
  ➡ Write program to extract useful information from a file containing the university timetable data.
- Reflection.
  ➡ Write up your reflections on this assignment.

To Submit

- Your version of `ClassTimes.java`
- Your `Reflection.txt` file

Overview

The assignment involves one set of exercises and one program (`ClassTimes`) that extracts data from a file of university timetable information.

Preparation

Download the zip file for assignment 4 and extract it to the `Assig4` folder in your home folder. It should contain templates for the Java program you are to complete, along with data files. Read through the whole assignment to see what you need to do.

Look at the model answers to assignment 3, and make sure you understand all the components of the programs. Also, go over the code examples from the lectures that used files.

Class Times

The university timetable system outputs a list of all the classes of all the courses in each trimester. However, the list

is very long, and using it can be anything from tedious (*eg* finding all the rooms for a particular course) to downright painful (*eg*, finding all the courses that have a lecture in a particular room).

You are to complete the `ClassTimes` program that will help a user find certain information from (a simplified version of) this timetable more quickly. The program has 9 methods, each of which will read the data from a file containing the class timetable data, and print out (to the text pane and/or to a file) just the data that is relevant to the user. All the methods are closely related and will have a very similar structure: loop through the file, reading the values on each line and processing them.

The class timetable file is called `classdata.txt`.

Each course may have lots of different classes: it may have several lectures and/or tutorials and/or laboratory sessions etc on different days at different times in different rooms. Each line of the file specifies one class or session for a course, so that there may be may be many lines for a each course.

Six of the lines of the file are shown here:

```
COMP102   Comp-Lab     Wed   1600   1700   CO219
COMP102   Comp-Lab     Wed   1600   1700   CO238
COMP102   Lecture      Mon   1100   1150   KKLT303
COMP102   Lecture      Wed   1100   1150   KKLT303
COMP102   Lecture      Fri   1100   1150   KKLT303
COMP102   Tutorial     Tue   1610   1700   AM104
```

The format of the lines is as follows:

- The first token is the course code
- The second token is the type of class
- The third token is the day of the week
- The fourth and fifth tokens are the start and end time of the session (using 24 hour time)
- The last token is the room code (The first part of the room code is always the building code, *eg* "HM" for Hugh McKenzie.

Note, the data is based on the real course timetable, but has been modified and simplified (and is not necessarily current for this year).

Core

Complete three methods:

- `printCourse(String targetCourse)`
  The method has one parameter - a course code. It should read the classdata file, printing out (to the text pane) all the class type, day, start and end times, and room for each class session of the given course (*ie*, data from all the lines where the first token on the line is equal to the course code). If there are no classes for the course, it

should print out a message that the course has no class times.
It will need to read the first token on each line using `next()`, and then can either read the rest of the line using `nextLine()` or read each value on the rest of the line separately. You will need to use some of the methods in the `String` class.

- `printRoom(String targetRoom)`
  The method has one parameter - a room code. It should print out a title: `"Classes in ..."` and write a dashed line under it. It should then read the file and print out the course code, class type, day, start time, and end time for each class session in the target room. It should then print a message saying how many classes were in the room, for example `"No courses are booked in KKLT302"`, or `"There are 25 classes in VZ103"`.

- `printAtStartTime(int startTime)`
  The method has one parameter - a start time. It should read the class timetable file, printing out the course code, room, type, day and end time for each class that starts at the target time. It will be best to read the six tokens on each line individually.

Completion

Complete three methods.

- `printInRoomsOnDay(String targetRoom1, String targetRoom2, String targetDay)`
  The method has three parameters - two room codes and a day. It should print out a title: *eg* `"Classes in KK201 or KK202 on Tue"` and underline it. It should then read the file and print out the course code and start time for each class session that is in either target room on the day.

- `bookingsFileForRoom(String targetRoom)`
  The method has one parameter - a room code. It should construct a new file, whose name is the room code, followed by `"_Bookings.txt"` [_eg_ `"HMLT205_Bookings.txt"`] with a list of the classes booked in that room. The first line of the file should specify what room the bookings are for, eg:
  `Bookings for room HMLT205`
  Each class booked in the room should be printed to the file on three lines with the course, the time, and the type, followed by a blank line, _eg_

  ```
  Course: ACCY130
  Time: Mon 1200-1250
  Session: Lecture
  ```

  The method should print a message when it has finished writing the file.
  Hint: Start by printing the output to the UI window using UI.println, and then make it write to the file.

- `inBuildingAfterTime(String targetBuilding, String targetDay, int targetStart)`
  The method has three parameters - a building code, a day, and a start time. It should print a list of all the

classes (course code, type, and room) that are in the specified building and start at or after the specified start time.
Note that the first part of every room name is the building code in uppercase.

Challenge

Complete two methods.

- `double meanClassLength(String targetRoom)`
  The method **returns** (not prints) the average (mean) duration in minutes of all classes scheduled in the specified room.
  Hints: be careful with the times - they are not ordinary integers, and do not cause an error if there are no classes in the room.

- `potentialDisruptions(String building, String targetDay, int targetStart, int targetEnd)`
  Lists all the courses (just the course code) that have classes in a given building on a given day such that any part of the class is between the given times. Each course involved should only be listed once, even if it has several classes in the building in the time period. Note that the data file is ordered by the course code.
  This method is similar to, but somewhat more complex than, one of the completion questions.

Reflection

Answer the following questions in the `Reflection.txt` file, and make sure you submit it.

1. The series of methods in ClassTimes were progressively more tricky. Identify what it was about the later ones that made them more difficult.
2. The lines in classdata.txt were sorted by the coursecode. What queries would be easier or have nicer output if they had been sorted by one of the other fields?

Print version | Backlinks |