**CMSC203**

**作业 #4**



**分配说明**

## 物业管理公司管理要出租的个别物业，并按月租金的百分比向他们收取管理费。属性不能相互重叠，并且每个属性必须在管理公司的图内。 编写一个应用程序，允许用户创建管理公司，并将公司管理的属性添加到其列表中。假设公司处理的最大属性数为 5。

**此分配测试的概念**

* 聚集
* 将对象传递到方法
* 阵列结构
* 对象作为数组的元素
* 处理数组元素
* 复制构造函数
* 朱尼特 ·泰斯 · 丁

**类**

**数据元素类** **=** **属性.java**

类*属性*将包含：

1. 属性名称、城市、租金金额、所有者和绘图的实例变量。 有关每个实例变量的数据类型，请参阅 JavaDoc。
2. to字符串 method 表示属性对象。
3. 构造函数和 getter 和 setter 方法。请参阅属性类的 Javadoc of the 。

**数据元素类** **=** **绘图.java**

类*图*将包含：

1. 实例变量表示位置左上角的 x 和 y 坐标，表示深度和宽度表示绘图的垂直和水平范围。
2. 表示绘图对象的 ToString 方法
3. 构造函数，请参阅绘图类的 Javadoc。
4. 名为 的方法*重叠，*该方法采用 Plot 实例并确定是否由当前绘图重叠。
5. 名为 的方法*包含采用* Plot 实例并确定当前绘图是否包含它的方法。 请注意，确定应该是包容性的，换句话说，如果边缘位于当前绘图的边缘，这是可以接受的。

**数据**结构 =属性对象的 Array 以保存管理公司处理的属性。此数组将声明为管理**竞争类的属性**。

**数据管理器类**  **=人一个公司. java**

此类不应具有任何输出功能（例如，没有 GUI 相关或打印相关功能），但应获取输入、对数据结构进行操作，并且使用 getters 访问返回值或设置变量 tha t。

它将包含名称、税号、管理费、MAX\_PROPERTY（常数设置为 5）的 i nstance变量，以及大小为 MAX\_PROPERTY 的属性属性的数组，以及两个常数MGMT\_WIDTH 和 MGMT\_DEPTH，都设置为 10; \_WIDTH and M定义管理公司类的绘图的类型的"绘图"的属性图。有关详细信息，请参阅 Javadoc。

类*管理公司将*包含以下方法，以及获取和设置方法：

1. 构造函数（有关更多详细信息，请参阅 Javadoc）
2. 方法**添加属性 （3 个版本）：**
   1. 方法添加属性版本 1：
      1. 输入类型属性对象的*参数*（调用属性*副本*构造函数）。.它将属性对象的副本*添加到*属性*数组*。
   2. 方法添加属性版本 2：
      1. 传递四种类型的参数：
         * 字符串属性名称，
         * 弦城，
         * 双倍租金，
         * 字符串所有者名称。
      2. 调用*属性* 4 参数构造函数。
   3. 方法添加属性版本 3：
      1. 传递八种类型的参数：
         * 字符串属性名称，
         * 弦城，
         * 双倍租金，
         * 字符串所有者名称，
         * int x，
         * int 和，
         * int 宽度
         * 深度。
      2. 调用*属性* 8 参数构造函数。
   4. add 属性方法将返回添加属性的数组的索引。 如果添加属性时出现问题，则如果数组已满，则此方法将返回 -1;如果属性为 null，则此方法将返回 -2;如果该属性的绘图or未包含在管理公司绘图中 will return -1 ，则此方法将返回 -2;如果该属性的绘图与任何其他属性的绘图重叠，则此方法将返回 -2。
3. 方法**totalRent**= 返回属性数组中属性*的总*租金。
4. 方法**maxRentPropertyIndex**-returns the 返回具有最高租金*金额的属性*数组中的属性索引。此方法将是私有**的**。
5. 方法**最大租金Prop**道具 - R在属性数组中提供物业的最高租金*properties* 金额。F或简单性假定每个"属性"对象的租金金额不同。此方法应调用 **maxRentPropertyIndex 方法**method**。**
6. 方法**字符串**- 通过访问"属性"数组返回此管理公司内所有属性的信息。格式如下示例：

联盟属性列表， 税 ID： 1235

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

物业名称 ：贝尔马

位于银泉

属于：约翰·史密斯

租金金额： 1200.0

物业名称： 卡姆登湖路

位于罗克维尔

属于：安·泰勒

租金金额： 2450.0

属性名称： 汉普顿

位于罗克维尔

属于：里克·史蒂夫斯

租金金额： 1250.0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

总管理费： 294.0

**可能需要其他方法才能包括在此类中。按照提供的 Javadoc 文件进行操作。**

**驱动程序类** **=**  **（提供）**

提供的 PropertyMgmDriverNoGui.java 是一个类，允许您测试管理公司的方法。

**GUI 驱动程序类** **=**  **（提供）**

提供图形用户界面（GUI） . 确保 GUI 将编译并运行您的方法。如果管理公司.java 中的方法不完全采用指定的格式，则 GUI 不会编译。

不要修改 GUI。

**联合测试**

运行 JUnit 测试文件（提供）。 确保 JUnit 测试全部成功。

不要修改 JUnit 测试。

在管理公司测试中实施测试。 这些测试应类似于 Junit the Junit 测试。

**分配详细信息**

编写一个名为"属性"的数据元素类，该类具有用于保存属性名称的字段、属性位于, the rent amount, the 的城市、租金金额、所有者名称和要由属性占用的绘图，以及要访问和设置这些字段的 getters 和设置器。编写参数化的 constructor（即，将字段的值作为参数）和副本 c上结构（以 P绳索对象作为参数）。 按照提供的 Javadoc 文件进行操作。

编写名为 P lot 的数据元素类，该类包含字段，指定每个绘图左上角的 X 和 Y 位置以及每个绘图的深度和宽度。 请注意，由于计算机监视器采用的网格系统，X、Y位置位于左上角，与正常笛卡尔坐标不同。

提供了一个驱动程序类，用于创建出租属性以测试属性管理器。 使用 JavaFX 提供图形用户界面，该界面复制了此驱动程序的功能。 您不需要读取任何数据，但 GUI 将允许您手动进入物业管理公司和每个属性。提供了图像目录。 **请务必将"图像"目录（提供）放在** **Eclipse 中的"src"目录中。**为了继续运行 GUI，不需要显示映像。

将初始文件从 Blackboard 和最终的 java 文件上传到 GitHub，在名为"CMSC203\_Assignment4"的目录中从实验室 1 的存储库中CMSC203\_Assignment4。

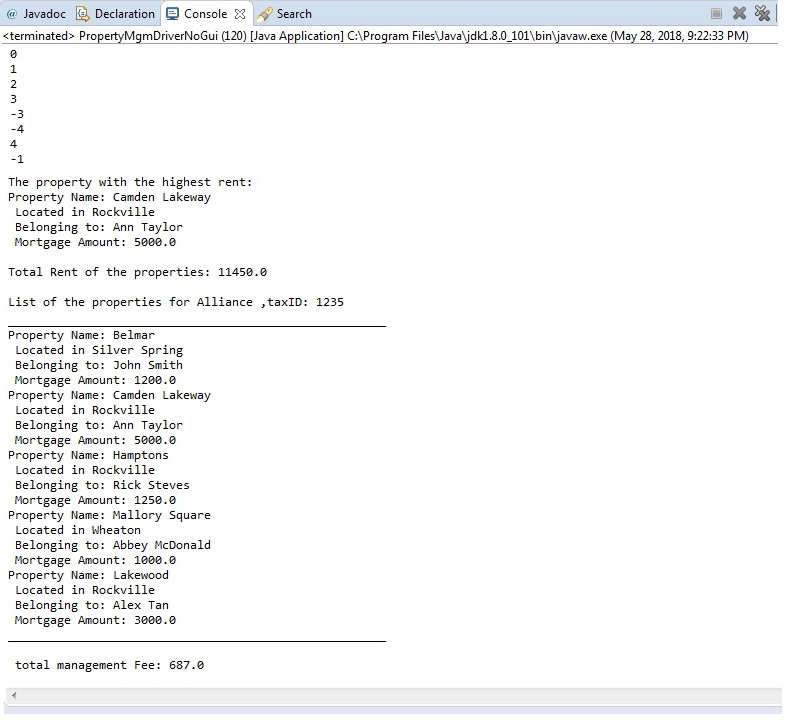
## 操作

当驱动程序驱动的应用程序启动时，驱动程序类（提供）创建管理公司，创建租赁属性，将它们添加到属性管理器，并使用属性管理器的方法打印有关属性的信息。

当 GUI 驱动的应用程序启动（提供）时，将创建一个窗口，如下面的屏幕截图所示，允许用户输入适用的数据并显示结果属性。 驱动程序和 GUI 都将使用相同的类和方法进行操作。

JUnit 测试类还测试与驱动程序和 GUI 相同的类。

**例子**

***运行PropertyMgm 驱动程序诺吉.***java 的预期***输出***

***使用 GUI 运行的预期输出：***

***PropertyMgmGui.java 在启动时***

手机截图

自动生成的描述

***添加管理 Co 信息（注意 Mgmt Co 图）***

社交媒体帖子的屏幕截图

自动生成的描述

***添加属性信息- 绘图大纲***

社交媒体帖子的屏幕截图

自动生成的描述

***添加属性信息- 成功添加***

手机截图

自动生成的描述

***添加属性信息- 不成功：重叠***

***社交媒体帖子的屏幕截图

自动生成的描述***

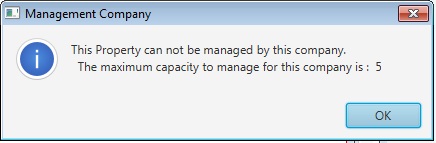
***添加属性信息- 不成功：Mgmt Co 绘图不包含属性图***

***注意：红色矩形的宽度延伸到窗口右侧。***

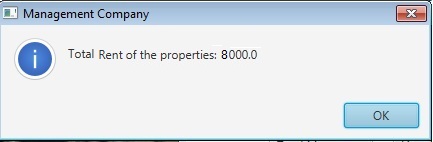
***社交媒体帖子的屏幕截图

自动生成的描述***

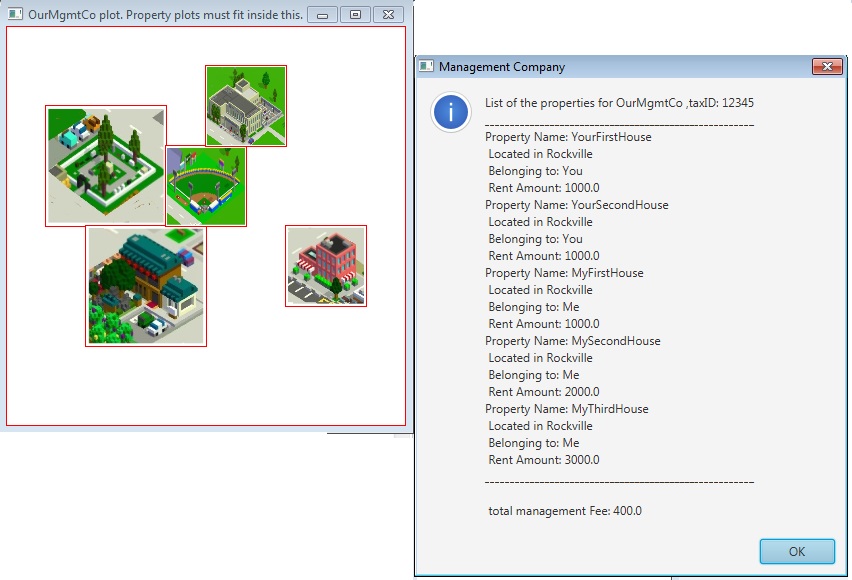
***添加属性信息- 不成功：属性太多***

******

***"总租金"按钮 结果"总租金"按钮的结果***

***"属性列表"按钮的结果***



**交付**

* 在 Word文档中打开 UML diagram 类关系图。
* 在管理公司.java、Property.java 和 Plot.java中指定的主要方法的 S ubmit伪Word 代码。 不要只列出打印出来时读取内容，而是解释使用的算法。
* 学习经验：突出您从本项目工作的经验教训和经验教训。
* 你学到了什么？
* 你在挣扎什么？
* 在下次项目上，您将采取哪些不同的做法？
* 包括您成功的项目哪些部分，以及您未成功的项目部分（如果有）。
* GitHub：在存储库中（请参阅实验 1），上传 Word 文件和 java 文件。您需要将这些文件作为目录的内容上载，以便将来的上载可以分开进行。 获取并提交 GitHub 存储库的屏幕截图。

**笔记：**

* 正确的命名约定：应命名除 0 和 1 以外的所有常量。常号应该是所有大写，变量名称应该以小写开始，但后续单词应该是标题大小写。变量和方法名称应描述变量或方法的作用。应避免单字母名称。
* 文档：所有编程项目的文档要求是程序顶部的一个块注释，其中包含用于开发项目的课程名称、项目编号、名称、日期和平台 /编译器。如果使用未创建的任何代码或特定算法，应在相应的注释块中引用其源。如有必要，应提供其他意见，以澄清程序。

缩进：它必须在整个程序中保持一致，并且必须反映控制结构

**提交详细信息**

提交以下文件：

* 名称为 FirstInitialLastName\_Assignment3.docx 的 Word 文档应包括：
* UML 类图
* 在ManagementCompany.java、property.java 和 Plot.java 中指定的每个方法的伪代码。
* 具有多个属性的 GUI 的屏幕快照
* GitHub 提交的屏幕快照
* 经验 教训
* 检查列表
* 包含 javadoc 文件的文档（目录）
* src （目录） *包含您的* *（. java） 文件*
  + 文件1.java（示例）
    - 文件2.java（示例）
    - File\_Test.java（示例）
* zip 文件将仅包含 .java 文件，并将命名为：FirstInitialLastName\_Assignment4\_Moss.zip。 **此 .zip 将没有任何文件夹 - 只有 .java 文件。**

**分级鲁比克**

参见附件： CMSC203 分配 4 Rubric\_Summer20.xlsx

作业4 检查列表

|  |  |  |  |
| --- | --- | --- | --- |
| **#** |  | **Y/N** | **评论** |
|  | **分配文件：** |  |  |
|  | * 第一初始姓氏 \_分配 # \_Moss.zip | **<是或否>** |  |
|  | * FirstInitialLastName\_Assignment#.docx/.pdf | **<是或否>** |  |
|  | * 源 java 文件 | **<是或否>** |  |
|  | **程序编译** | **<是或否>** |  |
|  | **程序运行时具有与测试计划相关的所需输出** | **<是或否>** |  |
|  | **文档文件：** |  |  |
|  | * 综合测试计划 | **<是或否>** |  |
|  | * 测试计划中列出的每个测试用例的屏幕截图 | **<是或否>** |  |
|  | * 提交分配# 的 GitHub帐户的屏幕截图（如果需要） | **<是或否或不/A>** |  |
|  | * UML 图 | **<是或否 >** |  |
|  | * 算法/伪代码 | **<是或否 >** |  |
|  | * 流程图（如果需要） | **<是或否或不/A>** |  |
|  | * 经验 教训 | **<是或否>** |  |
|  | * 清单已完成并包含在文档中 | **<是或否>** |  |