Montgomery College
CMSC 203
Assignment 4 Design

Class: CMSC203 CRN 32689

Program: Assignment 4 Design

Instructor: Dr.Grinberg

Summary of Description: (Give a brief description for a Program)

Due Date: <03/25/2024>

Integrity Pledge: I pledge that I have completed the programming assignment independently.

I have not copied the code from a student or any source.

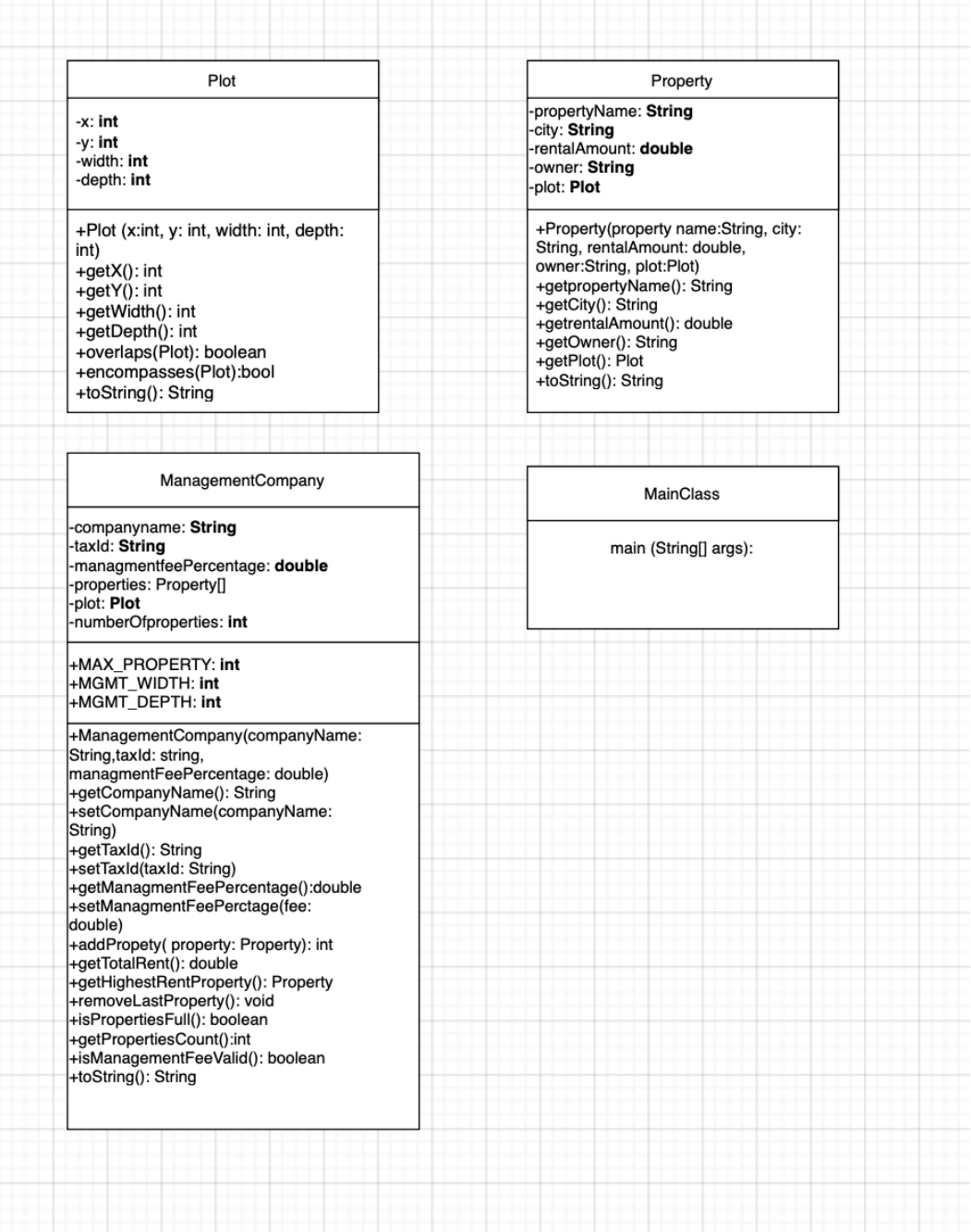
Student: Sousanna Chugunova

Part 1: Create UML diagrams for all the classes in this assignment.

Refer to the **UML Tutorial** on how to create UML diagrams.

Notes: To create UML Diagram you can use a simple free tool UMLSculptor
https://sourceforge.net/projects/umlsculptor/

See attached examples



Part 2: Pseudo Code for the primary methods specified in ManagementCompany.java, and Plot.java in a Word document. Do not just list what gets read in a printed out, but explain the algorithm being used.

Refer to the **Pseudocode Guideline** on how to write Pseudocode.

ManagementCompany	Plot
Class: ManagementCompany **// Attributes** name taxId mgmFeePer MAX_PROPERTY = 5 MGMT_WIDTH = 10 MGMT_DEPTH = 10 properties[MAX_PROPERTY] plot numberOfProperties **// Constructor with parameters** Constructor ManagementCompany(name, taxId, mgmFeePer): set name, taxId, and mgmFeePer to given values initialize plot with coordinates (0, 0) and dimensions (MGMT_WIDTH, MGMT_DEPTH) initialize properties array with size MAX_PROPERTY set numberOfProperties to 0 **// Getter/Setter methods for name, taxId, mgmFeePer** **// Method addProperty** Method addProperty(property): if properties array is full: return -1 if property is null: return -2 if plot does not encompass property's plot: return -3 for each existing property in properties array: if property's plot overlaps with existing property's plot: return -4 add property to properties array increment numberOfProperties return index of the added property **// Method getTotalRent** Method getTotalRent(): totalRent = 0 for each property in properties array: add property's rent amount to totalRent return totalRent	// Class: Plot **// Attributes** x, y, width, depth **// Constructor with parameters** function Plot(x, y, width, depth): set x to given x set y to given y set width to given width set depth to given depth **// Default constructor** function Plot(): set x to 0 set y to 0 set width to 0 set depth to 0 **// Getter/Setter methods** function getX(): return x function setX(newX): set x to newX // Similarly, define getY(), setY(), getWidth(), setWidth(), getDepth(), setDepth() methods **// Method named overlaps** function overlaps(otherPlot): if (otherPlot.x is between x and x + width) and (otherPlot.y is between y and y + depth): return true else if (x is between otherPlot.x and otherPlot.x + otherPlot.width) and (y is between otherPlot.y and otherPlot.y + otherPlot.depth): return true return false **// Method named encompasses** function encompasses(otherPlot):

```
// Method getHighestRentProperty
Method getHighestRentProperty():
    highestRentProperty = properties[0]
    for each property in properties array:
        if property's rent amount is greater than highestRentProperty's
rent amount:
            set highestRentProperty to property
    return highestRentProperty

// Method removeLastProperty
Method removeLastProperty():
    if numberOfProperties is 0:
        return
    remove the last property from properties array
    decrement numberOfProperties

// Method isPropertiesFull
Method isPropertiesFull():
    return numberOfProperties is equal to MAX_PROPERTY

// Method getPropertiesCount
Method getPropertiesCount():
    return numberOfProperties

// Method isMangementFeeValid
Method isMangementFeeValid():
    return mgmFeePer is between 0 and 100

// Method toString
Method toString():
    result = "List of the properties for " + name + ", taxID: " + taxId +
"\n"
    result +=
"_____\n"
    for each property in properties array:
        result += property.toString() + "\n"
    result +=
"_____\n"
    result += "total management Fee: " +
calculateTotalManagementFee()
    return result
```

```
    if (otherPlot.x >= x and otherPlot.x +
otherPlot.width <= x + width) and
(otherPlot.y >= y and otherPlot.y +
otherPlot.depth <= y + depth):
        return true
    return false

// toString method
function toString():
    return x + "," + y + "," + width + "," +
depth
```