Nicks Neon Objects v2.0

CIS 261

Prof. Collard

2/18/2020

Rodney Breslin

Problem Description

Creating a program that expands on the previous Nicks Neon Objects v1. This new program must contain all units in centimeters, Triangles with the ability to add 3 sides, and Brackets. The Brackets must account for the length of the tubes and the type of shape. Triangles must have at least 3, Circles must have at least 2 and Squares and Rectangles must both have at least 4. The program must also count how many different types of objects there are.

Algorithm/UML

\* SQUARE UML DIAGRAM

\* ----------------------

\* -tubeLength:double

\* -bracket:int

\* -height:double

\* ----------------

\* +Square()

\* +Square( height:double)

\* +Square(height:double, color:string, filled:boolean)

\* +getHeight():double

\* +setHeight(height:double)

\* +getArea():double

\* +getTubeLength():double

\* +getNumBrackets():int

RECTANGLE UML DIAGRAM

\* ----------------------

\* -width:double

\* -height:double

\* -tubeLength:double

\* -bracket:int

\* ----------------

\* +Rectangle()

\* +Rectangle(width:double, height:double)

\* +Rectangle(width:double, color:string, filled:boolean)

\* +getWidth():double

\* +setWidth(double width)

\* +getHeight():double

\* +setHeight(width:double)

\* +getArea():double

\* +getTubeLength():double

\* +getNumBrackets():int;

\* Geometric Object Superclass UML DIAGRAM

\* ------------------------------

\* -color: string

\* -filled: boolean

\* -dateCreated: string

\* ------------------------------

\* +GeometricObject()

\* +GeometricObject(color: string, filled: boolean)

\* +getColor(): string

\* +setColor(color: string)

\* +isFilled(): boolean

\* +setFilled(filled: boolean)

\* +getDateCreated(): string

\* +toString(): string

\* +getTubeLength():double

\* +getNumBrackets():int

\* CIRCLE CLASS UML

\* -----------------

\* -radius: double

\* -diameter: double

\* -tubeLength:double

\* -bracket:int

\* ---------------------

\* +Circle()

\* +Circle(diameter:double)

\* +getRadius():double

\* +setRadius(radius:double)

\* +getArea():double

\* +getDiameter():double

\* +setDiameter(diameter:double)

\* +getTubeLength():double

\* +printCircle():String

\* +getNumBrackets():int

\* TRIANGLE CLASS UML

\* -------------------

\* -side1:double

\* -side2:double

\* -side3:double

\* -tubeLength:double

\* -bracket:int

\* ---------------

\* +Triangle():

\* +Triangle(side1:double,side2:double,side3:double)

\* +getSide1():double

\* +getSide2():double

\* +getSide3():double

\* +setSide1(side1:double):

\* +setSide2(side2:double):

\* +setSide3(side3:double):

\* +getTubeLength():double

\* +getNumBrackets():int;

Step One: Create two new abstract methods getTubeLength() and getNumBrackets(). Set GeometricObject class to abstract. getTubeLength is the same as getPerimeter, getNumBrackets() is calculated via the shape and how many vertices it has. Create an if statement to check if getTubeLength is equal or less than twenty which will default the number of brackets. If the number is higher than 20, then more brackets will be added.

Step Two: Use instanceof to count the amount of rectangles, circles, squares, and triangles when the user presses Q.

Step Three: Make all measurements in centimeters.

Step Four: Use Printf to format the decimal places.

Test Set Design/Sample Output

Test Set #1 “20CM Square”

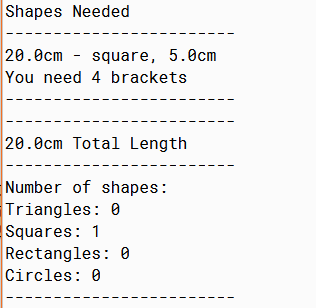
Expected Output: Should display the needed brackets as 4

Actual Output:

Test Set #2 “20CM Rectangle”

Expected Output: Should display the needed brackets as 4

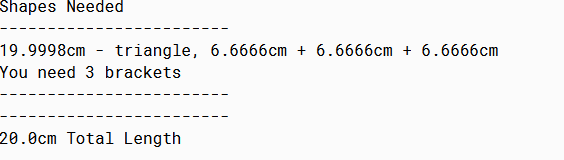
Actual Output:



Test Set #3“20CM Triangle”

Expected output: Should display the brackets needed as 3

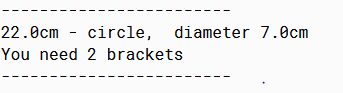
Actual Output:



Test Set #4“20CM Circle”

Expected Output: Should display as 2 brackets

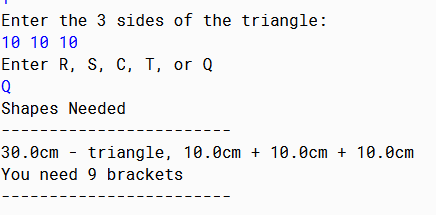
Actual Output:



Test Set#5“30CM Triangle”

Expected output: Should Display triangle as needing 9 brackets

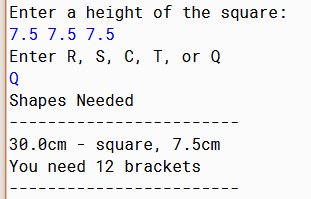
Actual Output:



Test Set#6 “30CM Square”

Expected Output: Should display the necessary brackets as 12

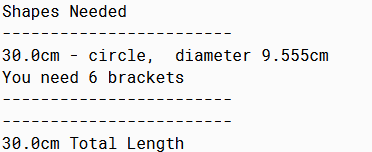
Actual Output:



Test Set#7 “30CM Circle”

Expected Output: Should display the necessary brackets as 6

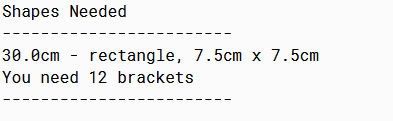
Actual Output:



Test Set#8 “30CM rectangle”

Expected Output: Should display the necessary brackets as 12

Actual Output:



Test Set#9 “Create 2 Triangles, 1 Square, 4 Rectangles, and 2 Circles”

Expected Output: The shape counter should display 2 Triangles, 1 Square, 4 Rectangles, and 2 circles

Actual Output:

