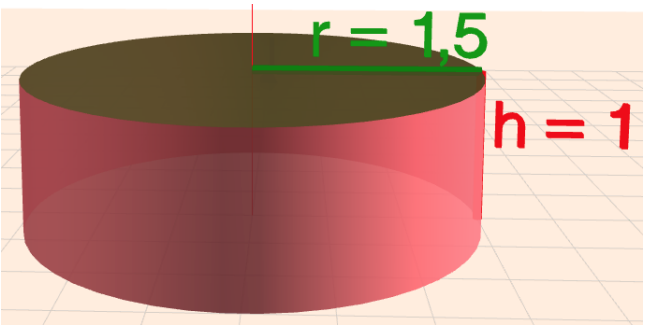
Puck Class Tests:

Table of Contents:

Default Constructor Test:

Puck defaulted:



Weight : ?
Thickness : ?
Radius : ?

Tester:

```
Puck defaulted = new Puck();
```

Should Return:

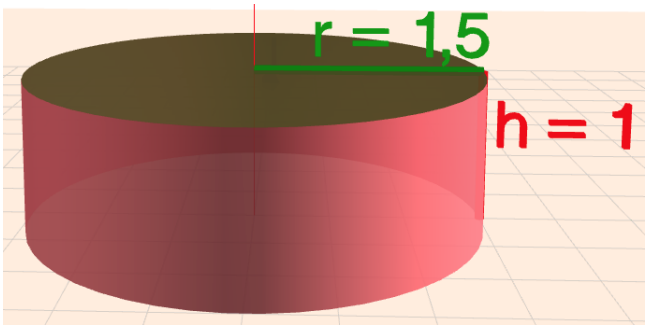
Weight : 1.00
Thickness : 1.00
Radius : 1.50
Serial : 100000

Result:

Weight : 1.00
Thickness : 1.00
Radius : 1.50
Serial : 100000

Second Constructor Test - Positive Weight:

Puck weighted:



Weight : 7.00
Thickness : ?
Radius : ?

Tester:

Puck weighted = new Puck(7.0);

Should Return:

Weight : 7.00
Thickness : 1.00
Radius : 1.50
Serial : 100050

Result:

Weight : 7.00
Thickness : 1.00
Radius : 1.50
Serial : 100050

Second Constructor Test - Negative Weight:

Puck antiGraviti:

NO PIC CAUSE CAN'T HAVE NEGATIVE WEIGHT

Weight : -8.00

Thickness : ?

Radius : ?

Tester:

```
Circle c1 = new Puck(-8);
Puck antiGraviti = (Puck)c1;
```

Should Return:

Weight : 0.00

Thickness : 1.00

Radius : 1.50

Serial : 100100

Result:

Weight : 0.00

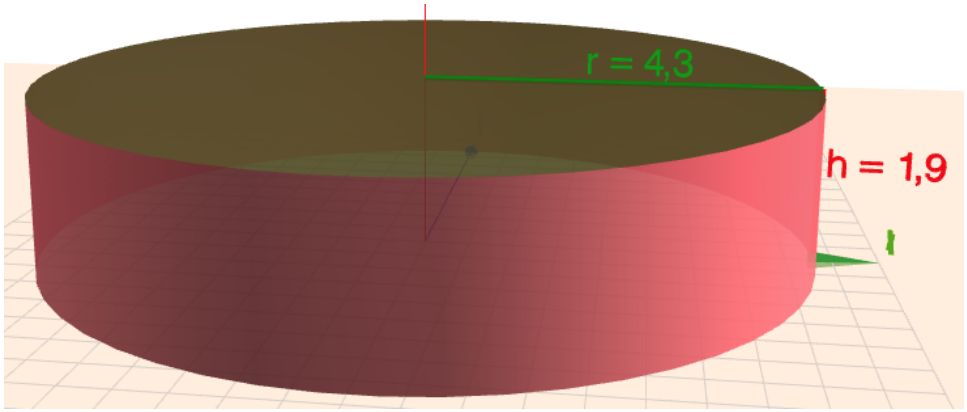
Thickness : 1.00

Radius : 1.50

Serial : 100100

Main Constructor Test - Positives:

Puck normal:



Weight : 6.2
Radius : 4.3
Thickness : 1.9

Tester:

Puck weighted = new Puck(6.2, 4.3, 1.9);

Should Return:

Weight : 6.20
Thickness : 4.30
Radius : 1.90
Serial : 100150

Result:

Weight : 6.20
Thickness : 4.30
Radius : 1.90
Serial : 100150

Main Constructor Test - Negatives:

Puck nonExist:

NO PICTURE DUE TO NEGATIVE VALUES.

Weight : -5.4

Radius : -2

Thickness : 0

Tester:

```
Object o1 = new Puck(-5.4, -2, 0);
Puck nonExist = (Puck)o1;
```

Should Return:

Weight : 0.00

Thickness : 0.00

Radius : 0.00

Serial : 100200

Result:

Weight : 0.00

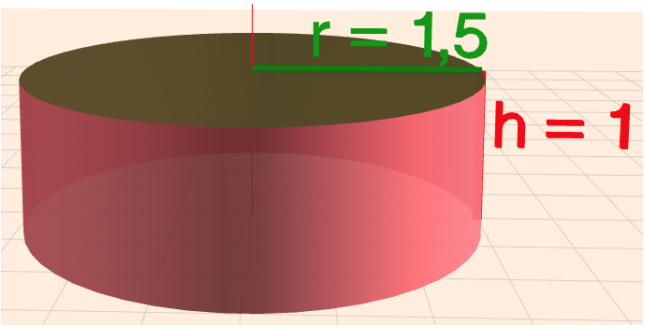
Thickness : 0.00

Radius : 0.00

Serial : 100200

‘getWeight’ & ‘setWeight’ Tests:

Puck weightTester:



Weight : 5.9
Radius : 1.5
Thickness : 1

Tester:

```
Disk d1 = new Puck(4.3);
Puck weightTester = (Puck)d1;
weightTester.setWeight(-5.9);
```

- **System.out**

```
weightTester.setWeight(5.9);
```

- **System.out**

Should Return:

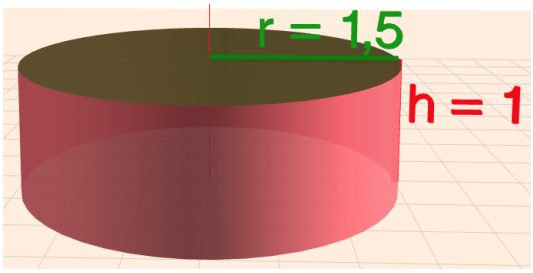
New Weight: 0.0
New Weight: 5.9

Result:

New Weight: 0.0
New Weight: 5.9

‘getDivision’ & ‘setDivision’ Tests - Standard:

Puck divStandart:



Weight : 5.5	Weight : 5.7	Weight : 6.1	Weight : 6
Radius : 1.5	Radius : 1.5	Radius : 1.5	Radius : 1.5
Thickness : 1	Thickness : 1	Thickness : 1	Thickness : 1

Tester:

Puck divStandart = new Puck(5.5);

- **System.out**
  divStandart.setWeight(5.7);
- **System.out**
  divStandart.setWeight(6.1);
- **System.out**
  divStandart.setWeight(6);

Should Return:

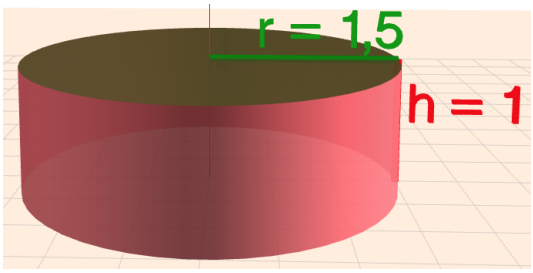
Puck Division: Standard
Puck Division: Standard
Puck Division: Non-Regulatory
Puck Division: Standard

Result:

Puck Division: Standard
Puck Division: Standard
Puck Division: Non-Regulatory
Puck Division: Standard

‘getDivision’ & ‘setDivision’ Tests - Youth:

Puck divYouth:



Weight : 3.9 Radius : 1.5 Thickness : 1	Weight : 4 Radius : 1.5 Thickness : 1	Weight : 4.2 Radius : 1.5 Thickness : 1	Weight : 4.6 Radius : 1.5 Thickness : 1

Weight : 4.5
Radius : 1.5
Thickness : 1

Tester:

Puck divYouth = new Puck(3.9);

- **System.out**
  divStandart.divYouth(4);
- **System.out**
  divStandart.divYouth(4.2);
- **System.out**
  divStandart.divYouth(4.6);
- **System.out**
  divStandart.divYouth(4.5);
- **System.out**

Should Return:	Result:
Puck Division: Non-Regulatory	Puck Division: Non-Regulatory
Puck Division: Youth	Puck Division: Youth
Puck Division: Youth	Puck Division: Youth
Puck Division: Non-Regulatory	Puck Division: Non-Regulatory
Puck Division: Youth	Puck Division: Youth

‘equals’ Tests:

Tester:

- **System.out** (defaulted.equals(defaulted));
  antiGraviti.setWeight(7);
- **System.out** (antiGraviti.equals(weighted));
- **System.out** (divYouth.equals(divStandart));

Should Return:

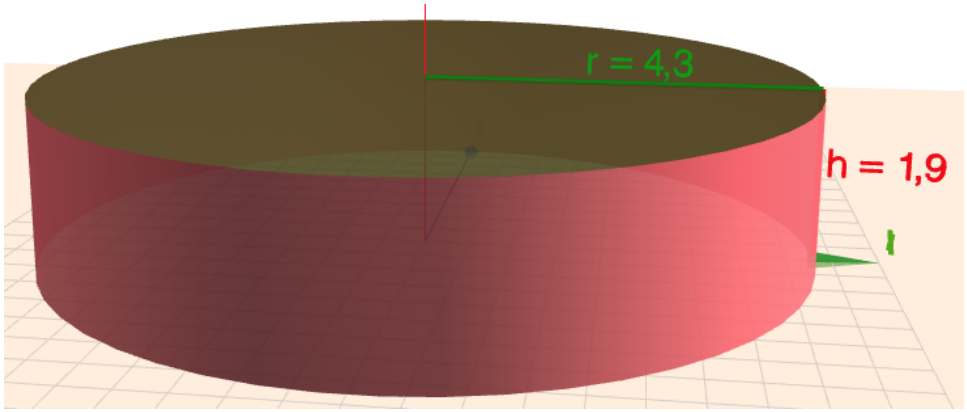
Equal? true
Equal? true
Equal? false

Result:

Equal? true
Equal? true
Equal? false

‘toString’ Test:

Puck normal:



Weight : 6.2
Radius : 4.3
Thickness : 1.9

Tester:

- **System.out** (normal);

Should Return:

Radius: 4.30” Thickness: 1.90”
Weight: 6.20oz
This puck is a Non-Regulation puck.
Serial Number 100150

Result:

Radius: 4.30” Thickness: 1.90”
Weight: 6.20oz
This puck is a Non-Regulation puck.
Serial Number 100150

‘compareTo’ Tests:

Tester:

- **System.out** (antiGraviti.equalsTo(weighted));
- **System.out** (nonExist.equals(defaulted));
- **System.out** (defaulted.equals(nonExist));

Should Return:

0
-1
1

Result:

0
-1
1

‘menu’ Test:

Tester:

Radius = 1
Thickness = 4
Weight = 8.4
Amount = 100

Should Return:

Radius: 1.00” Thickness: 4.00”
Weight: 8.40oz
This puck is a Non-Regulation puck.
Serial Number 100050

Radius: 1.00” Thickness: 4.00”
Weight: 8.40oz
This puck is a Non-Regulation puck.
Serial Number 105000

Result:

Radius: 1.00” Thickness: 4.00”
Weight: 8.40oz
This puck is a Non-Regulation puck.
Serial Number 100050

Radius: 1.00” Thickness: 4.00”
Weight: 8.40oz
This puck is a Non-Regulation puck.
Serial Number 105000

‘shipPucks’ Tests - 1 Small Package:

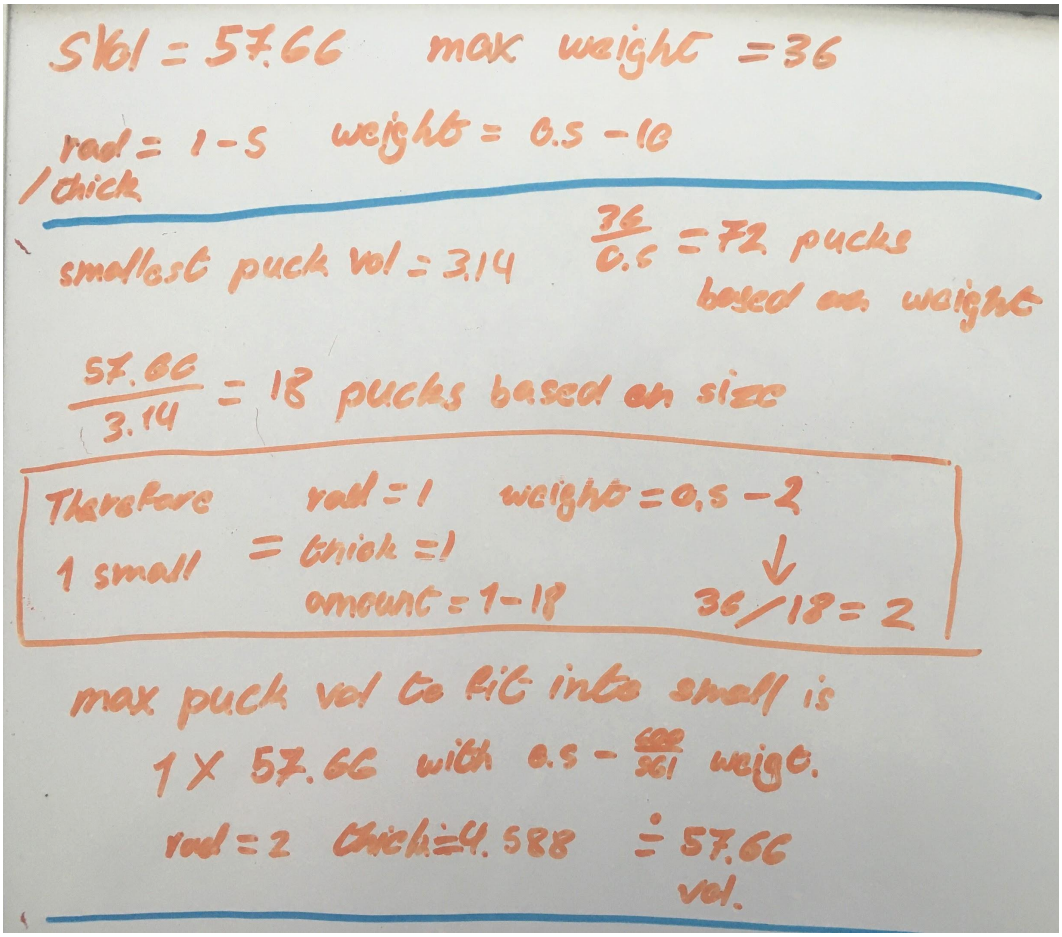
Testers:

1. Radius = 1, Thickness = 1, Weight = 0.5, Amount = 1.
2. R = 1, T = 1, W = 0.5, Amount = 18.
3. R = 1, T = 1, W = 2, Amount = 1.
4. R = 1, T = 1, W = 2, Amount = 18
5. R = 1, T = 1, W = 10, Amount = 1
6. R = 1, T = 1, W = 10, Amount = 3.
7. R = 2, T ≈ 4.588, W = 0.5, Amount = 1.
8. R = 2, T ≈ 4.588, W = 10 = Amount = 1.

Should **ALL** Return:

1xSMALL PKG $5.00

Logic:



(Max weight for last part is 10, because 10*1 = 10 which is smaller than 36.)

Tot Weight = 65/1 = 65 ≈ 10

‘shipPucks’ Tests - 1 Medium Package:

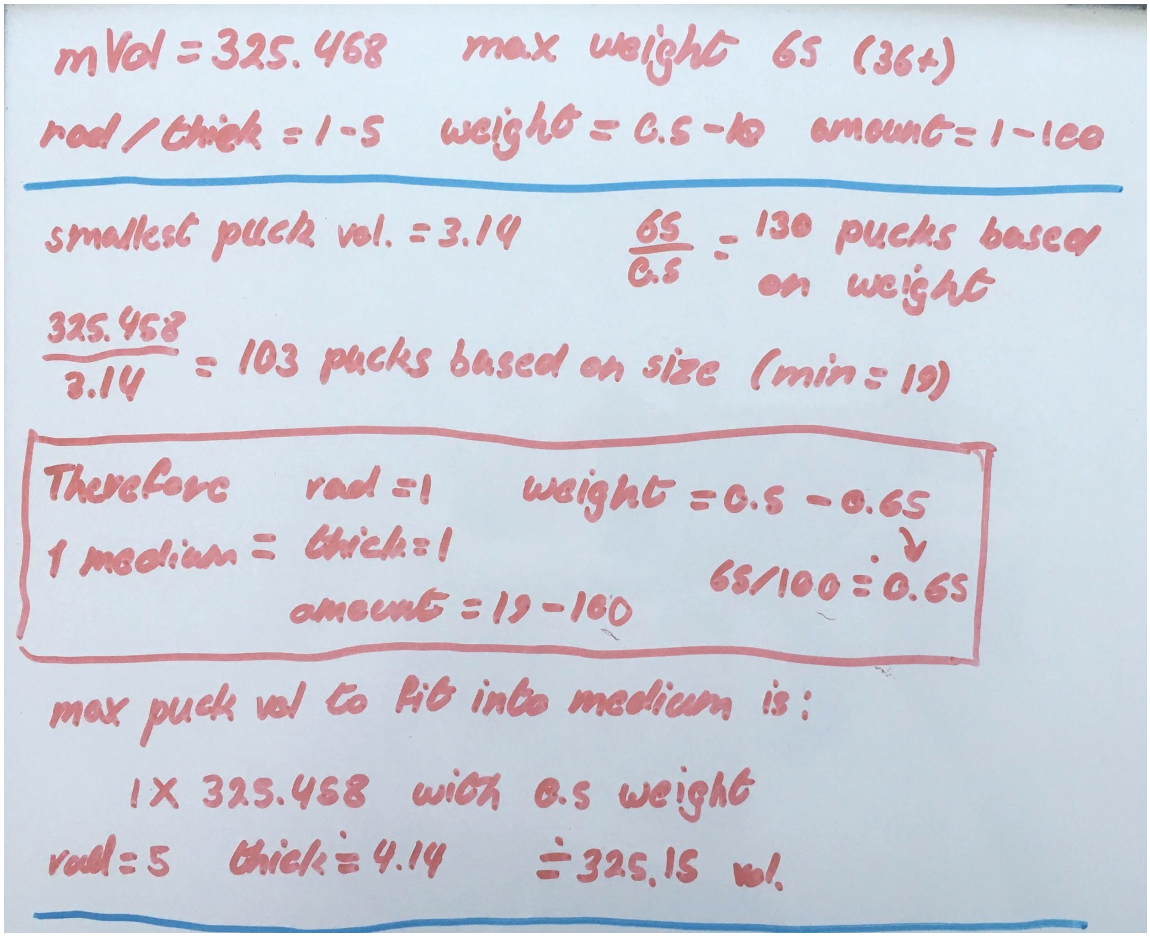
Testers:

1. Radius = 1, Thickness = 1, Weight = 0.5, Amount = 19.
2. R = 1, T = 1, W = 0.5, Amount = 100.
3. R = 1, T = 1, W = 0.65, Amount = 19.
4. R = 1, T = 1, W = 0.65, Amount = 100.
5. R = 1, T = 1, W = 3.42, Amount = 19.
6. R = 1, T = 1, W = 10, Amount = 6.
7. R = 5, T = 4.14, W = 0.5, Amount = 1.
8. R = 5, T = 4.14, W = 10, Amount = 1.

Should **ALL** Return:

1xMEDIUM PKG $8.00

Logic:



Tot Weight = 65/19 = 3.42

(If weight is 1, can't enter more than 65 pucks.)

‘shipPucks’ Tests - 1 Large Package:

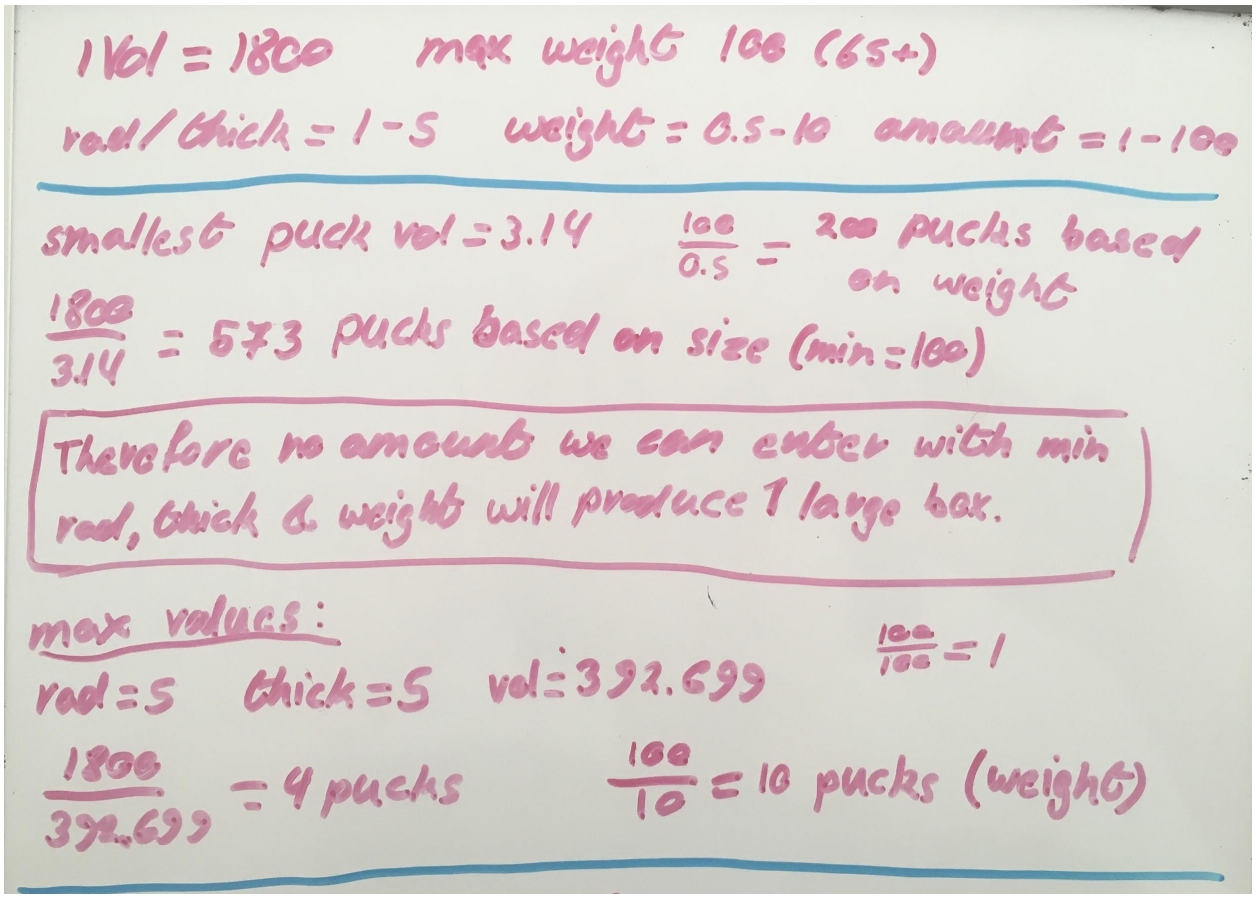
Testers:

1. Radius = 5, Thickness = 5, Weight = 0.5, Amount = 1.
2. R = 5, T = 5, W = 0.5, A = 4.
3. R = 5, T = 5, W = 10, A = 1.
4. T = 5, T = 5, W = 10, A = 4.
5. R = 1, T = 1, W = 1, A = 100.
6. R = 1, T = 1, W = 1, A = 66.

Should **ALL** Return:

1xLARGE PKG $10.00

Logic:



‘shipPucks’ Tests - Max Puck Values:

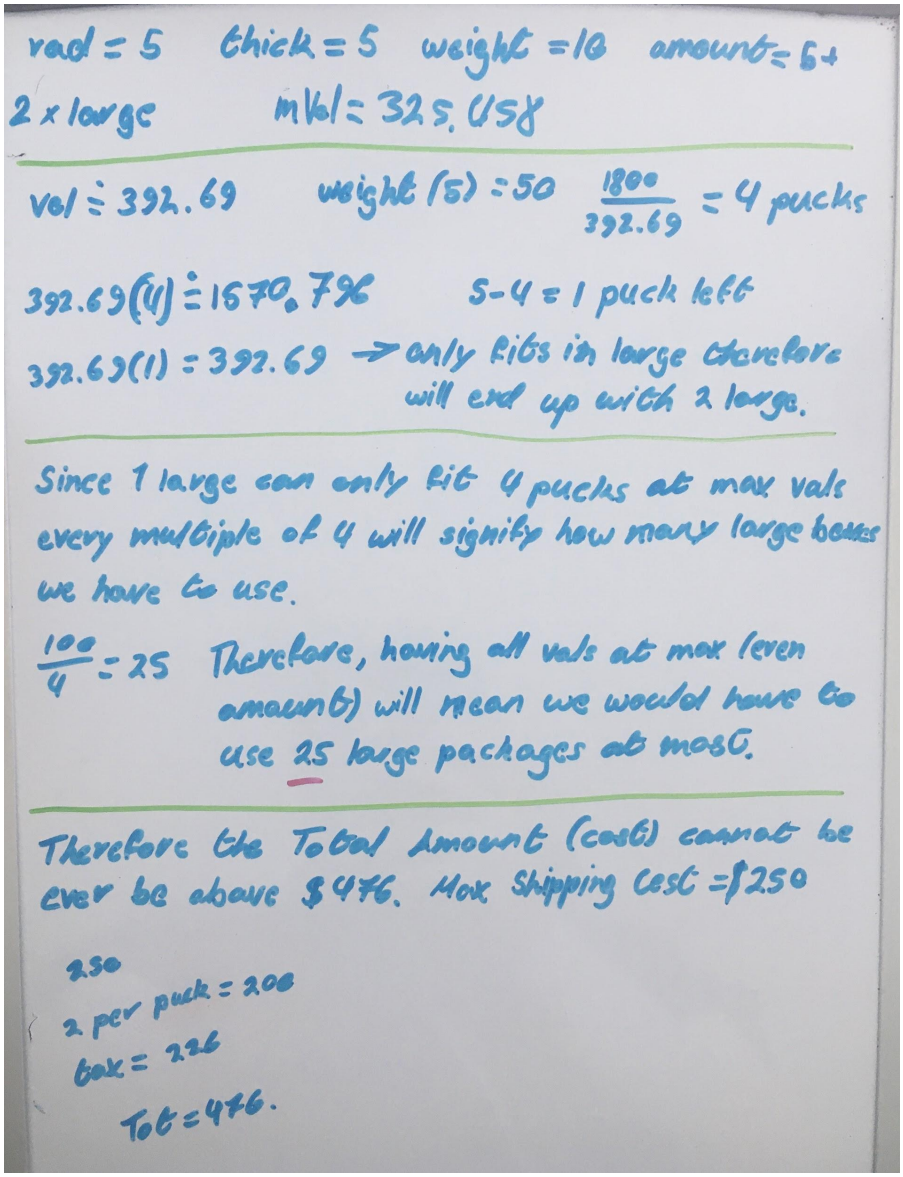
Testers:

1. Radius = 5, Thickness = 5, Weight = 10, Amount = 5 - 8
2. R = 5, T = 5, W = 10, A = 9 - 12
3. R = 5, T = 5, W = 10, A = 13 - 16
4. R = 5, T = 5, W = 10, A = 41 - 44
5. R = 5, T = 5, W = 10, A = 97 - 100

Should Return:

6. 2xLARGE PKG $20.00
7. 3xLARGE PKG $30.00
8. 4xLARGE PKG $40.00
9. 11xLARGE PKG $110.00
10. 24xLARGE PKG $240.00
11. 25xLARGE PKG $250.00

Logic:



‘shipPucks’ Tests - Random Values:

Testers:

1. Radius = 3.5, Thickness = 2.3, Weight =1.7, Amount = 10
2. R = 1, T = 5, W = 6, A = 100
3. R = 2.5, T = 1.0, W = 2.5, A = 27
4. R = 2, T = 5, W = 5, A = 57
5. R = 1.5, T = 1, W = 1, A = 30
6. R = 4.6, T = 3.6, W = 2.3, A = 17
7. R = 4.3, T = 2.2, W = 9.0, A = 56
8. R = 4.3, T = 3.4, W = 7.6, A = 36
9. R = 1.3, T = 2.3, W = 4, A = 11
10. R = 3, T = 2, W = 1, A = 36

Should Return:

1. 1xLARGE PKG $10.00
2. 6xLARGE PKG & 1xMEDIUM PKG $68.00
3. 1xLARGE PKG $10.00
4. 3xLARGE PKG $30.00
5. 1xMEDIUM PKG $8.00
6. 3xLARGE PKG $30.00
7. 5xLARGE PKG & 1xMEDIUM PKG $58.00
8. 4xLARGE PKG $40.00
9. 1xMEDIUM PKG $8.00
10. 1xLARGE PKG & 1xMEDIUM PKG $18.00

Note: There would never be more than 1 medium box / 1 small box since once either of them hits becomes 2 of that type of box, it can easily be converted to the one size above them since it will be more cost efficient to have a small amount of large boxes than many small boxes.

Example:

1 x Large Box	$10.00
2 x Medium Boxes	$16.00
Total:	$26.00

Convection:

2 Medium = 1 Large OR 1 Large & 1 small.
1 Large + 1 Large = 2 Large.

2 x Large Boxes	$20.00
Total:	$20.00

OR

2 x Large Boxes	$20.00
1 x Small Box	$ 5.00
Total:	$25.00

Both are more cost efficient than the original.

For ‘formatText’ compare the result of your entered value to the ‘FormattedInvoice.png’ picture to see if they match.
The Total Shipping was an added addition.