

# Programming Languages: Smalltalk Assignments

Jan Daciuk

January 4, 2022

# Smalltalk Assignments

The assignments should fulfill the following conditions:

- ① they should teach significant features of the programming language,
- ② to prevent copying, they should be different for different students.

Messages are an important feature of Smalltalk.

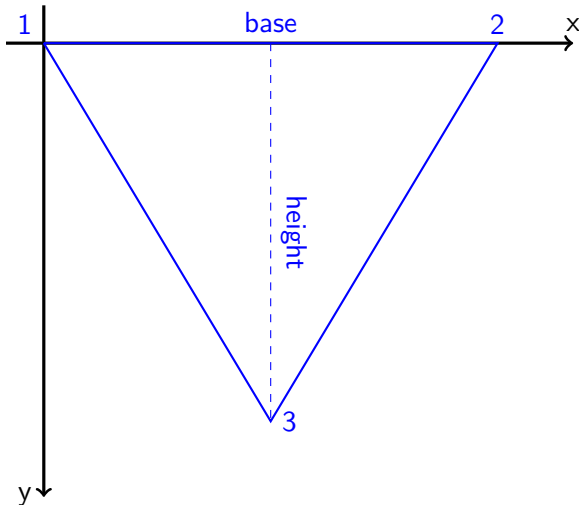
## Smalltalk Assignments

The last digit of the enrollment number *modulo* 5 determines the geometric figure that should be defined as a class being a subclass of `Wielokat` class:

- ① isosceles triangle (base and height)
- ① right triangle (legs, right angle is upper left)
- ② rhombus with 60° angle (side)
- ③ right trapezoid (bases and height)
- ④ regular hexagon (side)

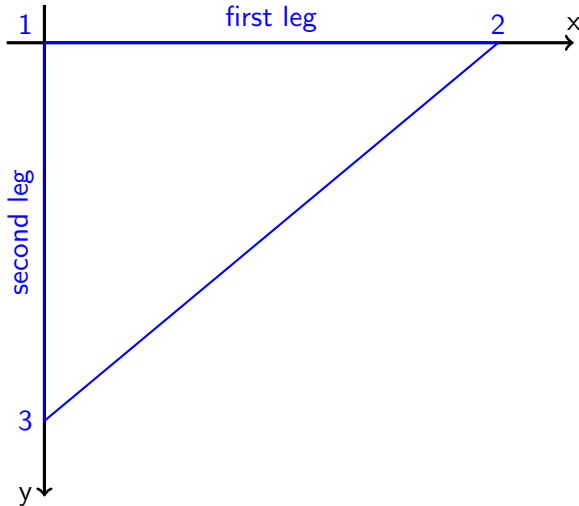
The new class should make it possible to add figures by their area. The result of addition should have the area equal to the sum of the areas of added figures, and the proportions of the recipient of the message. For the given figure, one should define `pole` (area) method. If the last digit of the enrollment number is even, one should define a method for translation of **a polygon** by a given vector (`Point` object). If odd — a rotation right by 90° round the first vertex. One should also define `drukuj` (print) method **printing** the vertices and the area of a **polygon**.

# Isosceles Triangle



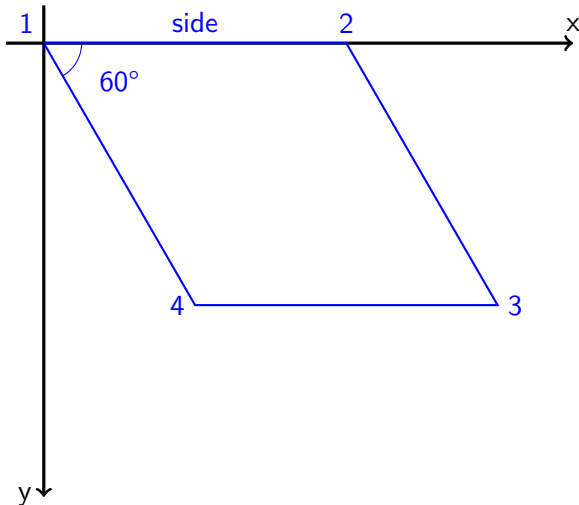
The result of the addition should have the same proportions as the recipient of the message.

# Right Triangle



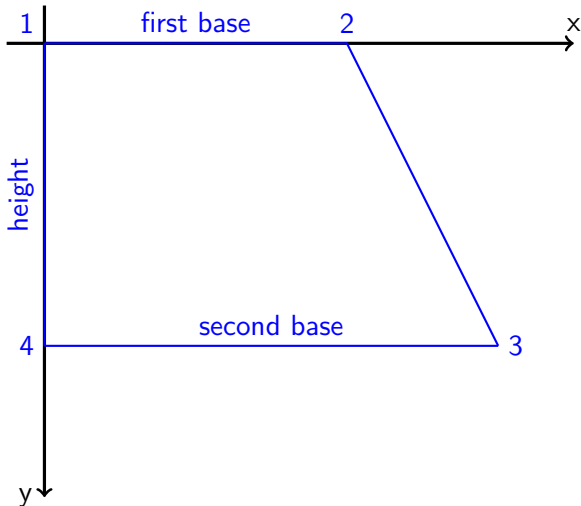
The result of the addition should have the same proportions as the recipient of the message.

# Rhombus



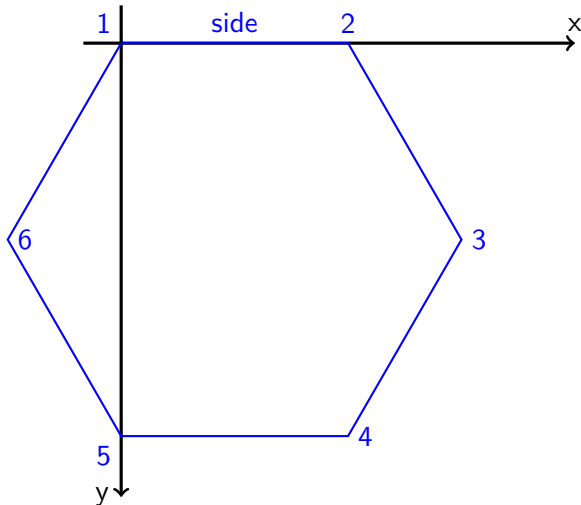
The result of the addition should have the same proportions as the recipient of the message.

# Right Trapezoid



The result of the addition should have the same proportions as the recipient of the message.

# Regular Hexagon



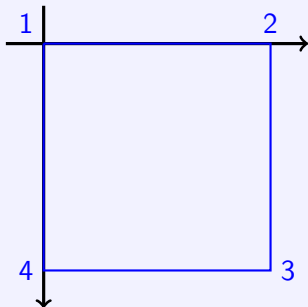
The result of the addition should have the same proportions as the recipient of the message.



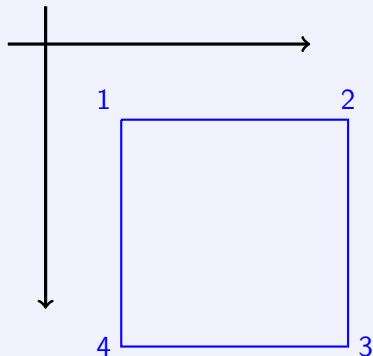
# Pitfalls

Rotation and translation can damage something that used to work:

was



is (translation)

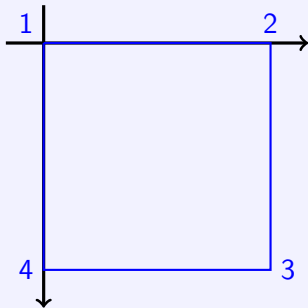


The area of the square was computed as the square of the x coordinate of the second vertex. . . The problem is not specific to the square. . .

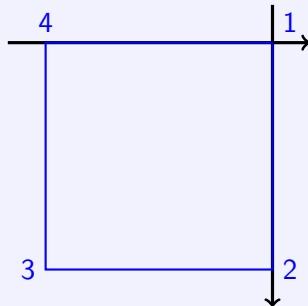
# Pitfalls

Rotation and translation can damage something that used to work:

was



is (rotation)



The area of the square was computed as the square of the x coordinate of the second vertex. . . The problem is not specific to the square. . .

# Tests for Isosceles Triangle

## with translation

```
t := (Trojkat new) initialize: 3
wysokosc: 4.
t drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 't+t'; cr.
t1 := t przesun: 2@3.
t1 drukuj.
t2 := t1 + t1 + t1 + t1.
t2 drukuj.
Transcript show: 't+k'; cr.
k1 := k przesun: (1 negated)@(1
negated).
t2 := t1 + k1.
t2 drukuj.
Transcript show: 'k+t'; cr.
k2 := k1 + t1.
k2 drukuj.
```

## with rotation

```
t := (Trojkat new) initialize: 3
wysokosc: 4.
t drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 't+t'; cr.
t1 := t obrot.
t1 drukuj.
t2 := t1 + t1 + t1 + t1.
t2 drukuj.
Transcript show: 't+k'; cr.
k1 := k obrot.
t2 := t1 + k1.
t2 drukuj.
Transcript show: 'k+t'; cr.
k2 := k1 + t1.
k2 drukuj.
```

# Tests for Right Triangle

## with translation

```
t := (Trojkat new) initialize: 3
przyprostokątna: 4.
t drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 't+t'; cr.
t1 := t przesun: 2@3.
t1 drukuj.
t2 := t1 + t1 + t1 + t1.
t2 drukuj.
Transcript show: 't+k'; cr.
k1 := k przesun: (1 negated)@(1
negated).
t2 := t1 + k1.
t2 drukuj.
Transcript show: 'k+t'; cr.
k2 := k1 + t1.
k2 drukuj.
```

## with rotation

```
t := (Trojkat new) initialize: 3
przyprostokątna: 4.
t drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 't+t'; cr.
t1 := t obrot.
t1 drukuj.
t2 := t1 + t1 + t1 + t1.
t2 drukuj.
Transcript show: 't+k'; cr.
k1 := k obrot.
t2 := t1 + k1.
t2 drukuj.
Transcript show: 'k+t'; cr.
k2 := k1 + t1.
k2 drukuj.
```

# Tests for Rhombus

## with translation

```
r := (Romb new) initialize: 4.  
r drukuj.  
k := (Kwadrat new) initialize: 2.  
k drukuj.  
Transcript show: 'r+r'; cr.  
r1 := r przesun: 2@3.  
r1 drukuj.  
r2 := r1 + r1 + r1 + r1.  
r2 drukuj.  
Transcript show: 'r+k'; cr.  
k1 := k przesun: (1 negated)@(1  
negated).  
r2 := r1 + k1.  
r2 drukuj.  
Transcript show: 'k+r'; cr.  
k2 := k1 + r1.  
k2 drukuj.
```

## with rotation

```
r := (Romb new) initialize: 4.  
r drukuj.  
k := (Kwadrat new) initialize: 2.  
k drukuj.  
Transcript show: 'r+r'; cr.  
r1 := r obrot.  
r1 drukuj.  
r2 := r1 + r1 + r1 + r1.  
r2 drukuj.  
Transcript show: 'r+k'; cr.  
k1 := k obrot.  
r2 := r1 + k.  
r2 drukuj.  
Transcript show: 'k+r'; cr.  
k2 := k1 + r1.  
k2 drukuj.
```

# Tests for Right Trapezoid

## with translation

```
r := (Trapez new) initialize: 4
podstawa: 2 wysokosc: 3.
t drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 't+t'; cr.
t1 := t przesun: 2@3.
t1 drukuj.
t2 := t1 + t1 + t1 + t1.
t2 drukuj.
Transcript show: 't+k'; cr.
k1 := k przesun: (1 negated)@(1
negated).
t2 := t1 + k1.
t2 drukuj.
Transcript show: 'k+t'; cr.
k2 := k1 + t1.
k2 drukuj.
```

## with rotation

```
t := (Trapez new) initialize: 4
podstawa: 2 wysokosc: 3.
t drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 't+t'; cr.
t1 := t obrot.
t1 drukuj.
t2 := t1 + t1 + t1 + t1.
t2 drukuj.
Transcript show: 't+k'; cr.
k1 := k obrot.
t2 := t1 + k.
t2 drukuj.
Transcript show: 'k+t'; cr.
k2 := k1 + t1.
k2 drukuj.
```

# Tests for Regular Hexagon



## with translation

```
s := (Szesciokat new) initialize:
4.
s drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 's+s'; cr.
s1 := s przesun: 2@3.
s1 drukuj.
s2 := s1 + s1 + s1 + s1.
s2 drukuj.
Transcript show: 's+k'; cr.
k1 := k przesun: (1 negated)@(1
negated).
s2 := s1 + k1.
s2 drukuj.
Transcript show: 'k+s'; cr.
k2 := k1 + s1.
k2 drukuj.
```

## with rotation

```
s := (Szesciokat new) initialize:
4.
s drukuj.
k := (Kwadrat new) initialize: 2.
k drukuj.
Transcript show: 's+s'; cr.
s1 := s obrot.
s1 drukuj.
s2 := s1 + s1 + s1 + s1.
s2 drukuj.
Transcript show: 's+k'; cr.
k1 := k obrot.
s2 := s1 + k.
s2 drukuj.
Transcript show: 'k+s'; cr.
k2 := k1 + s1.
k2 drukuj.
```

# Notes

- Prepare a file with the test.
- Only drukuj (print) prints anything.
- przesun (move or translate) and obrot (rotate) do not create new objects; they modify existing ones.
- New objects are created with the first vertex in the center of the coordinate system (0,0).
- New objects are oriented as in the figures on previous overheads.
- It is difficult to save the program in the file in format (the same as for kwadrat.st file).
- One can save it in the XML format using Package/File out/Package. . .
- Such file can be read in the file browser (File/File Browser or F2 or ) using File/File In or .