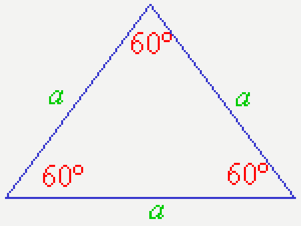
TRIANGLE EXERCISE OUTLINE SOLUTION

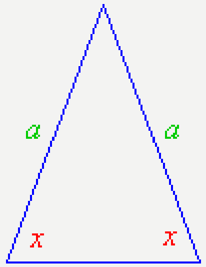
DOING YOUR FIRST TESTING!

- You need a set of test cases (i.e. specific sets of data) that will adequately test this program:

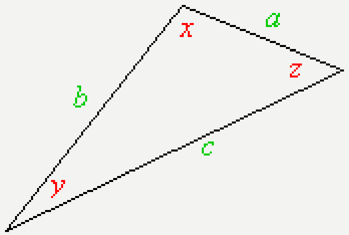
> *The program reads three integer values from an input dialog. The three values represent the lengths of the sides of a triangle. The program displays a message that states whether the triangle is scalene (ingen ens sider), isosceles(ligebenet), or equilateral (ligesidet)*



An **equilateral triangle** has all three sides of equal length.



An **isosceles triangle** has two sides of equal length.



A **scalene triangle** has no sides of equal length.

Triangle Exercise

MISSING INFO?

- Remember, that writing test cases is a really good way of ensuring that you get the requirements right.
- Also remember to identify the invalid test cases!

- Anything missing in the exercise description?
  - A more precise definition of a triangle:
    - A triangle is a closed figure with three sides
    - Interior angles always add up to 180 degrees (maybe not important for your solution)
    - The impossible case happens if the two shorter lengths add up to less than or equal to the longest length e.g.:
      - 2, 3, 10 cannot form a triangle
      - 2, 3, 5 cannot form a triangle

DO YOU HAVE …

1. A test case that represents a valid scalene triangle?
2. A test case that represents a valid equilateral triangle?
3. A test case that represents a valid isosceles triangle?
4. At least three test cases that represent valid isosceles triangles such that you have tried all three permutations of two equal sides
5. A test case in which one side has a zero value?
6. A test case in which one side has a negative value?
7. A test case with three integers greater than zero where the sum of two of the integers is equal to the third? (e.g. 1, 2, 3)
8. At least three test cases in category 7 such that you have tried all three permutations where the length of one side is equal to the sum of the lengths of the other two sides
9. A test case with three integers greater than zero such that the sum of two of the integers is less than the third (e.g. 2, 5, 8)
10. At least three test cases in category 9 such that you have tried all three permutations
11. A test case in which all sides are zero (0,0,0)?
12. At least one test case specifying non-integer values
13. At least one test case specifying the wrong number of values
14. For each test case did you specify the expected output?

HOW DID YOU DO?

- If you are typical, you have done poorly on this test.
- Before you become concerned about your own score, consider this: highly qualified professional programmers score, on the average, only 7.8 out of a possible 14.
- If you've done better, congratulations ☺

ID	Test Case Description	Test Case Input			Expected Output
		a	b	c	
1	Valid scalene triangle	5	3	4	Scalene
2	Valid isosceles triangle	3	3	4	Isosceles
3	Valid equilateral triangle	3	3	3	Equilateral
4	First permutation of two equal sides	50	50	25	Isosceles
5	Second permutation of two equal sides	25	50	50	Isosceles
6	Third permutation of two equal sides	50	25	50	Isosceles
7	One side zero length	1000	1000	0	Invalid
8	One side has negative length	3	3	-4	Invalid
9	Three sides greater than zero, sum of two smallest is equal to the largest	1	2	3	Invalid
10	2nd permutation of 9	1	3	2	Invalid
11	3rd permutation of 9	3	1	2	Invalid
12	Three sides greater than zero, sum of two smallest is less than the largest?	2	5	8	Invalid
13	2nd permutation of 12	2	8	5	Invalid
14	3rd permutation of 12	8	5	2	Invalid
15	All sides zero	0	0	0	Invalid

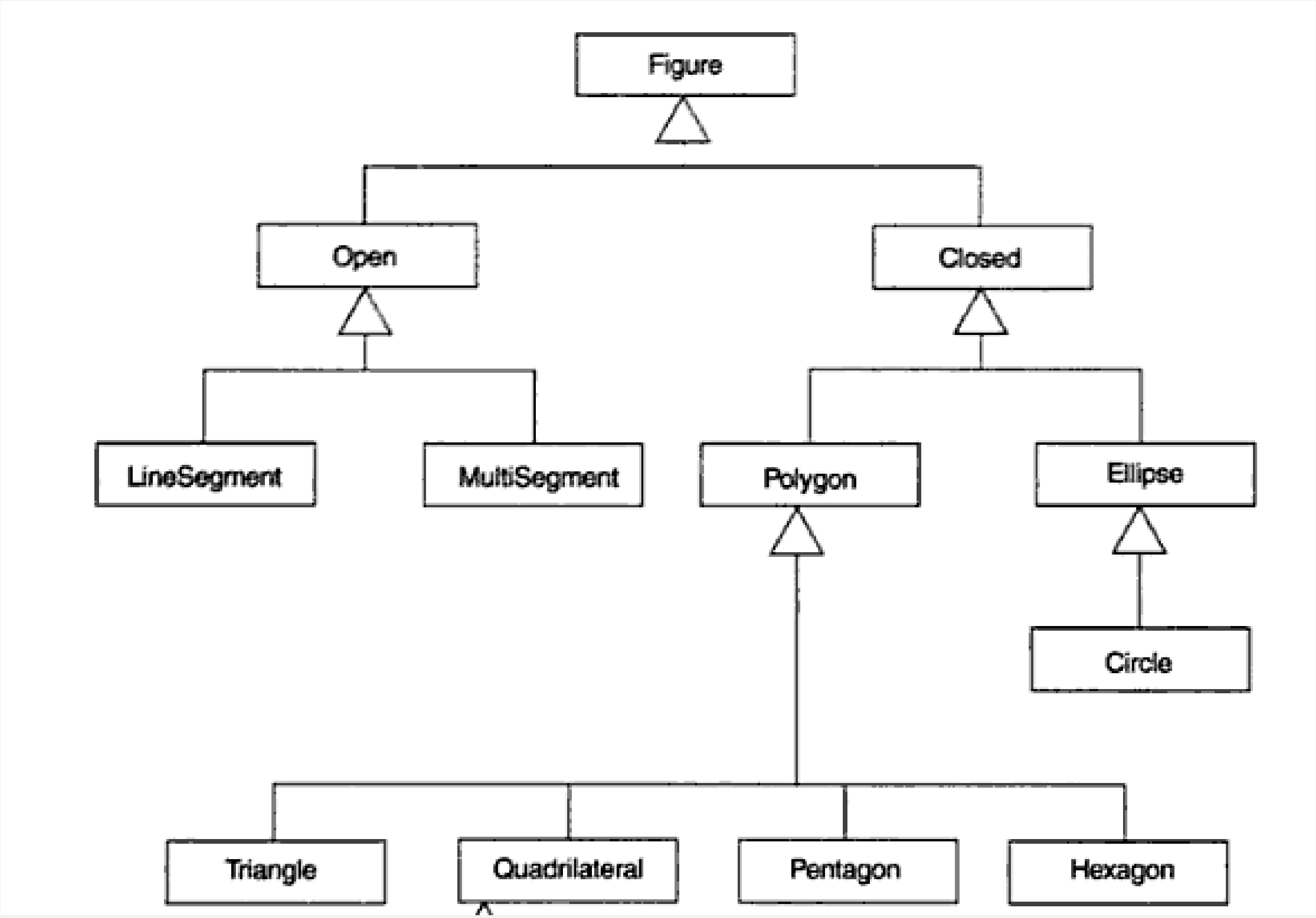
16	Non-integer input a†	@	4	5	Invalid
17	Non-integer input b†	3	$	5	Invalid
18	Non-integer input c†	3	4	%	Invalid
19	Missing input a†		4	5	Invalid
20	Missing input b†	3		5	Invalid
21	Missing input c†	3	4		Invalid
22	Three sides at maximum possible value	32767	32767	32767	Equilateral
23	Two sides at maximum possible value	32767	32767	1	Isosceles
24	One side at maximum possible value	1	1	32767	Invalid

† infeasible for Java/C# implementation

A set of test cases that satisfy these conditions do not guarantee that all possible errors will be found, but an adequate test should expose at least these errors.

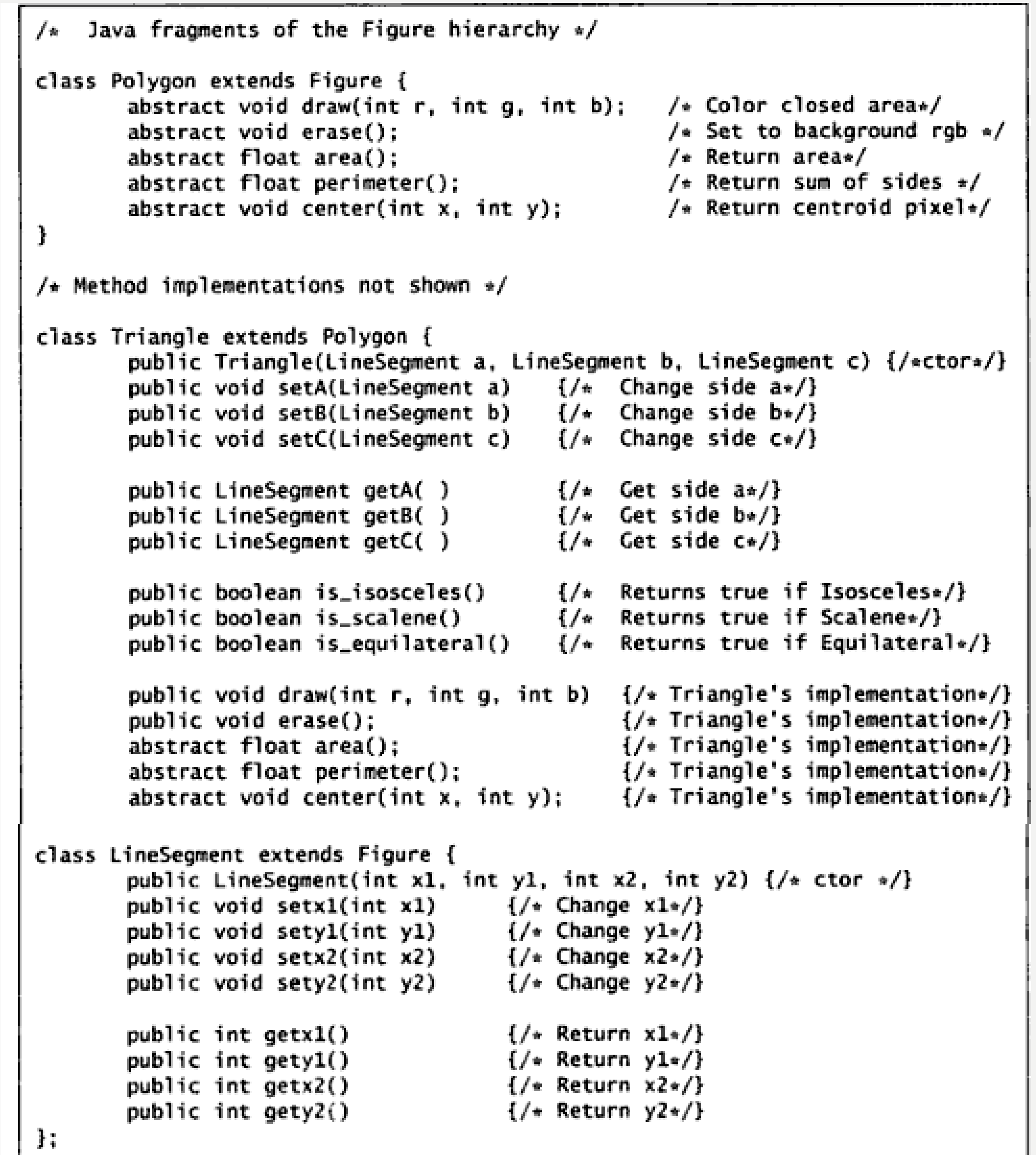
TRIANGLE PROGRAM - OO DESIGN

Design for
`Figure` hierarchy



Source: Binder 2000

TRIANGLE PROGRAM - CLASSES



```
/*  Java fragments of the Figure hierarchy */

class Polygon extends Figure {
        abstract void draw(int r, int g, int b);   /* Color closed area*/
        abstract void erase();                     /* Set to background rgb */
        abstract float area();                     /* Return area*/
        abstract float perimeter();                /* Return sum of sides */
        abstract void center(int x, int y);        /* Return centroid pixel*/
}

/* Method implementations not shown */

class Triangle extends Polygon {
        public Triangle(LineSegment a, LineSegment b, LineSegment c) {/*ctor*/}
        public void setA(LineSegment a)    {/*  Change side a*/}
        public void setB(LineSegment b)    {/*  Change side b*/}
        public void setC(LineSegment c)    {/*  Change side c*/}

        public LineSegment getA( )         {/*  Get side a*/}
        public LineSegment getB( )         {/*  Get side b*/}
        public LineSegment getC( )         {/*  Get side c*/}

        public boolean is_isosceles()      {/*  Returns true if Isosceles*/}
        public boolean is_scalene()        {/*  Returns true if Scalene*/}
        public boolean is_equilateral()    {/*  Returns true if Equilateral*/}

        public void draw(int r, int g, int b)  {/* Triangle's implementation*/}
        public void erase();                   {/* Triangle's implementation*/}
        abstract float area();                 {/* Triangle's implementation*/}
        abstract float perimeter();            {/* Triangle's implementation*/}
        abstract void center(int x, int y);    {/* Triangle's implementation*/}

class LineSegment extends Figure {
        public LineSegment(int x1, int y1, int x2, int y2) {/* ctor */}
        public void setx1(int x1)      {/* Change x1*/}
        public void sety1(int y1)      {/* Change y1*/}
        public void setx2(int x2)      {/* Change x2*/}
        public void sety2(int y2)      {/* Change y2*/}

        public int getx1()             {/* Return x1*/}
        public int gety1()             {/* Return y1*/}
        public int getx2()             {/* Return x2*/}
        public int gety2()             {/* Return y2*/}
};
```

Source: Binder 2000

OO TRIANGLE PROGRAM NEEDS EXTRA TESTS

- Test that the constructor creates the lines you designate
- Try min. and max. values for each `LineSegment` parameter
- Try to repeat a result after permuting line lengths
- Try to repeat a result after erase
- Try to repeat a result after draw
- Two parallel lines
- Three parallel lines
- Three nonintersecting, nonparallel lines
- …