CSG 100 Data Structures

- Goals
  - Get you comfortable with using Java for your programming needs
  - Introduce Data Structures and Algorithms
    - usage,
    - implementing
    - creating your own.
- Requirements
  - Reading
  - Coding

Class Information

- All relevant information about the class are available on the web

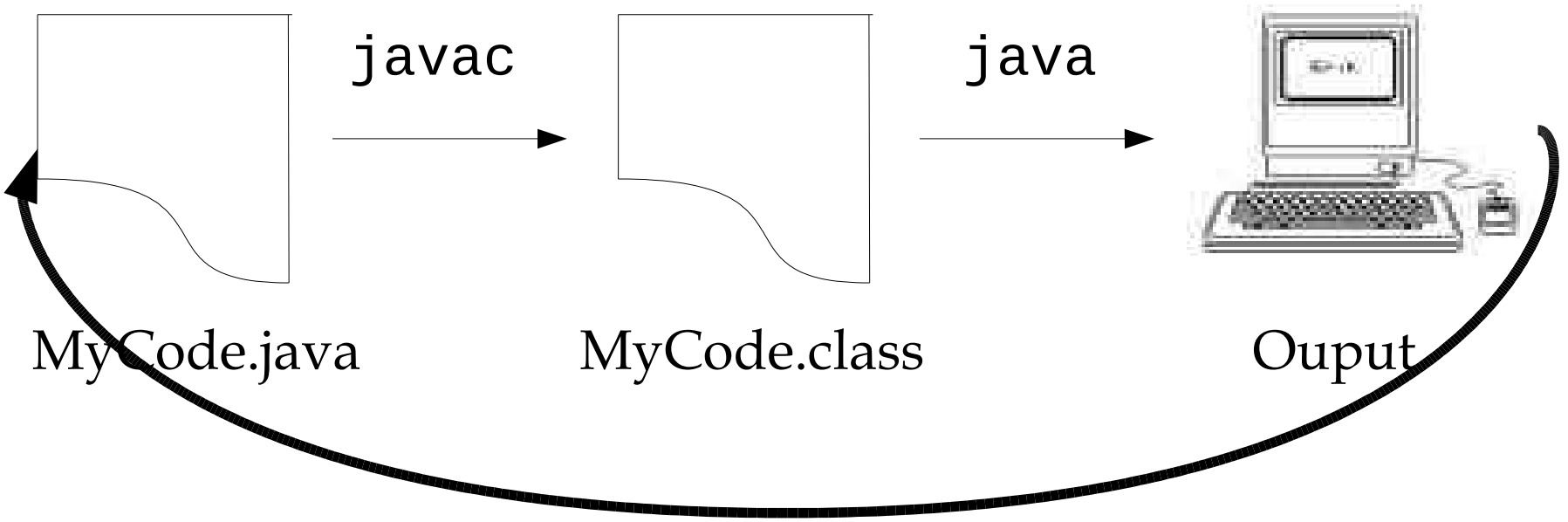
  http://www.ccs.neu.edu/home/skotthe/classes/csg100/f04/

- Textbooks
  - The Java Tutorial (available on line)
  - Thinking in Java 3rd Edition (available on line)
- Grades
  - assignments 20%, midterm 20%, final 40%, project 20%
    - (may eliminate final)

Class Information (cont.)

- Resources
  - CCIS accounts have all the necessary programs.
    - You can work remotely through SSH (check web page)
  - On your own machine
    - You need to download Java (check web page)
- Getting Help
  - Through the class mailing list
  - Office Hours (Monday 3:00 – 5:00 pm WVH Room 330)
  - Arrange some other time

Write, Compile and Run Cycle



- Create `MyCode.java` with your editor
- Compile using `javac` to get machine readable code
- Run using `java`

Java's “compile once run anywhere”

- The Java Virtual Machine (JVM)
  - is responsible for the communication between the Java program and your OS
  - Think and use Java concepts for manipulating OS resources
  - Provides extensive library and an interface to it called an API (Application Programming Interface)

Programs

- A program consists of a sequence of instructions
- This sequence can be subdivide into smaller sub-sections
  - Methods
  - Classes
  - Modules
  - Functions
- In Java there is one special method (main) indicating the starting point of the program (instruction sequence)
  - `public static void main(String[] args){..}`

Programs (cont.)

- Kinds of instructions
  - variables: An item of data named by an identifier
    - e.g. `int age;`
  - expressions: a series of variables, operators, and method calls that evaluates to a single value
    - e.g. `age++, age = getAge()`
  - statements: A statement forms a complete unit of execution (instruction)
    - e.g. `age = getAge();,`

    `Character.toUpperCase("s");`
  - blocks: a group of zero or more statements enclosed in balanced braces. It can be used in the place of a statement

Programs (cont.)

- e.g.

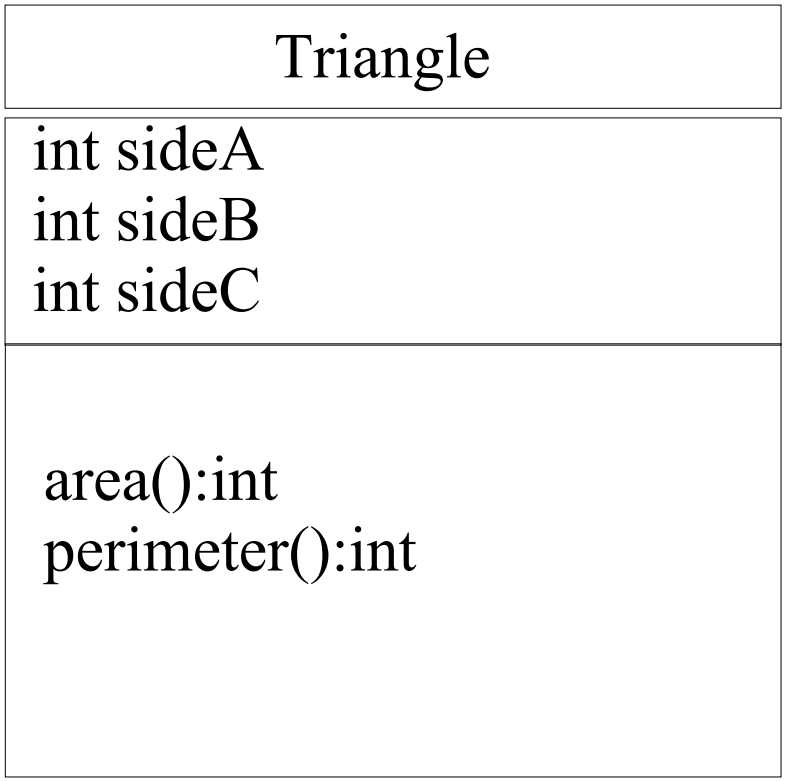
```
{
  age = getAge();
  Character.toUpperCase("s");
}
```

- methods (in OO): consists of a sequence of statements to perform an action, a set of input parameters to parameterize those actions, and possibly an output value (called return value) of some kind.
  - e.g. `public void setAge(int value)`

Data Structures

- Data Structure:
  - Any method of organizing a collection of data to allow it to be manipulated effectively. It may include meta-data describing the data it can hold
    - e.g. `List, Enumeration, Array, Stack`
- Data Type (or Type)
  - A set of values from which a variable, constant, function or other expression nay take its value. A classification of data that tells the compiler/interpreter how the programmer intends to use it.
    - e.g. `int, char, String`, int -> int, (int->int->int)->int

Java Types



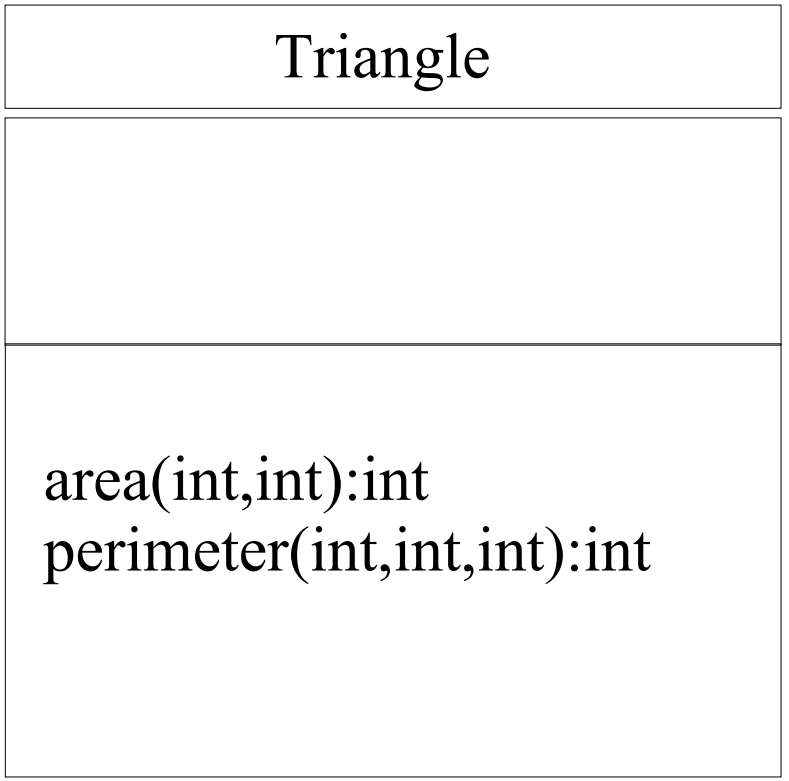
Type Name

Instance Variables or Members or Attributes

Instance Methods,
Understood Messages

- Textual notation for defining a Java Type

Java Types (cont)



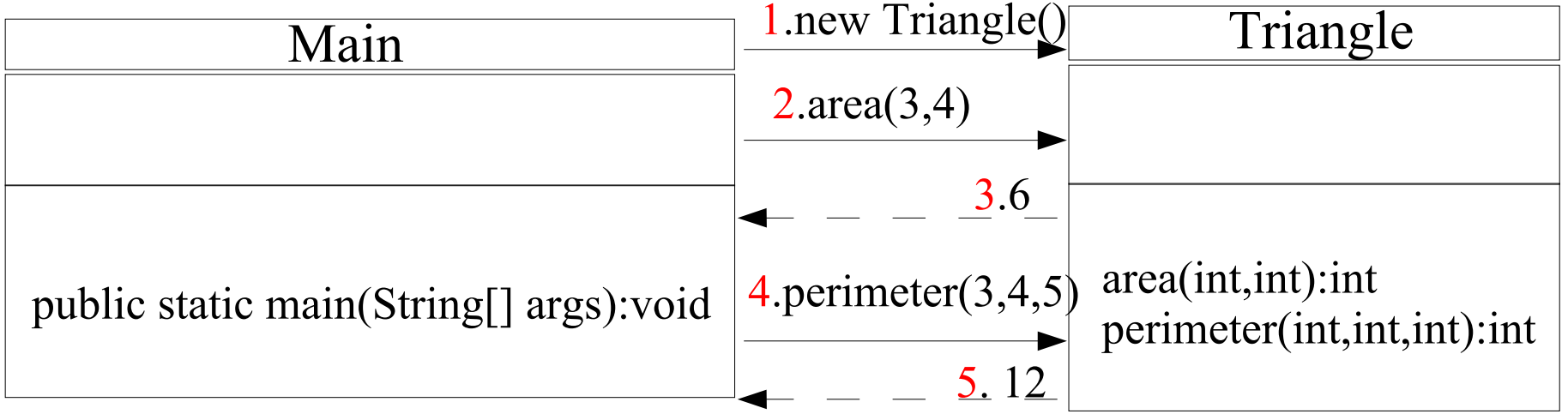

```
public class Triangle {



  public int area(int a, int b){
    int result = 0;
    result = (a*b)/2;
    return result;
  }

  public int perimeter(int a, int
b, int c){
    int result = 0;
    result = a + b + c;
    return result;
  }
}
```

Let's see some output

- **Object Oriented Programs**
  - a collection of entities (objects) communicating with each other by sending and receiving messages.
  - **NOTE:** Special "main" method that kick starts the computation.

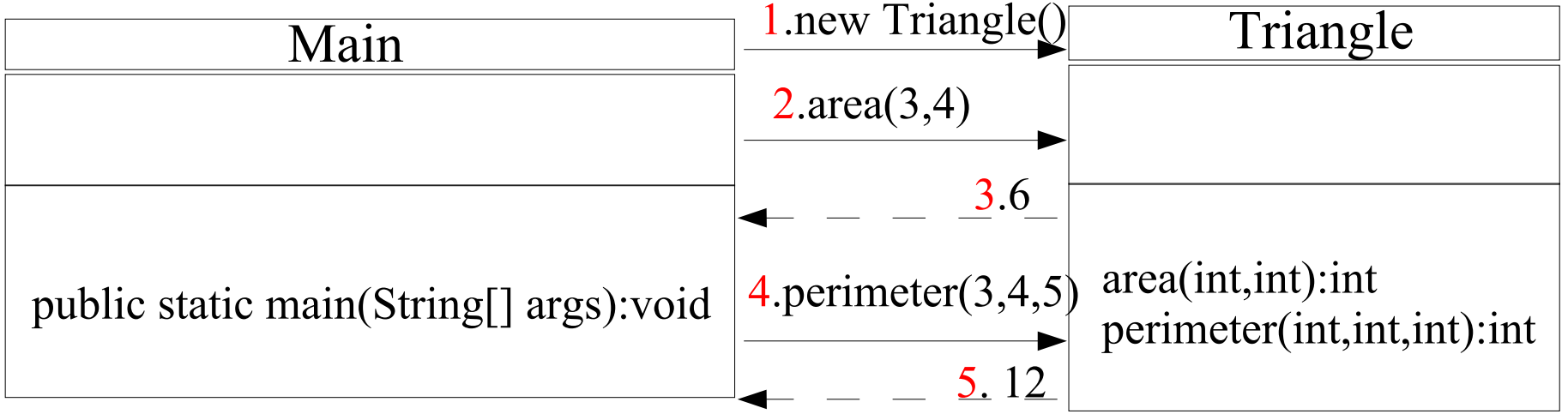
```
public class Main{

  public static void main(String[] args){
   // create an instance of Triangle called "aTriangle"
   Triangle aTriangle = new Triangle();
   int area ; // to store the result of area
   area = aTriangle.area();
   System.out.println("Triangle 3 4 5 has area : " + area);
   int perimeter ;
   perimeter = aTriangle.circumference();
   System.out.println("Triangle 3 4 5 has perimeter : " +
                      perimeter);
  }
}
```

Let's see some output (cont)



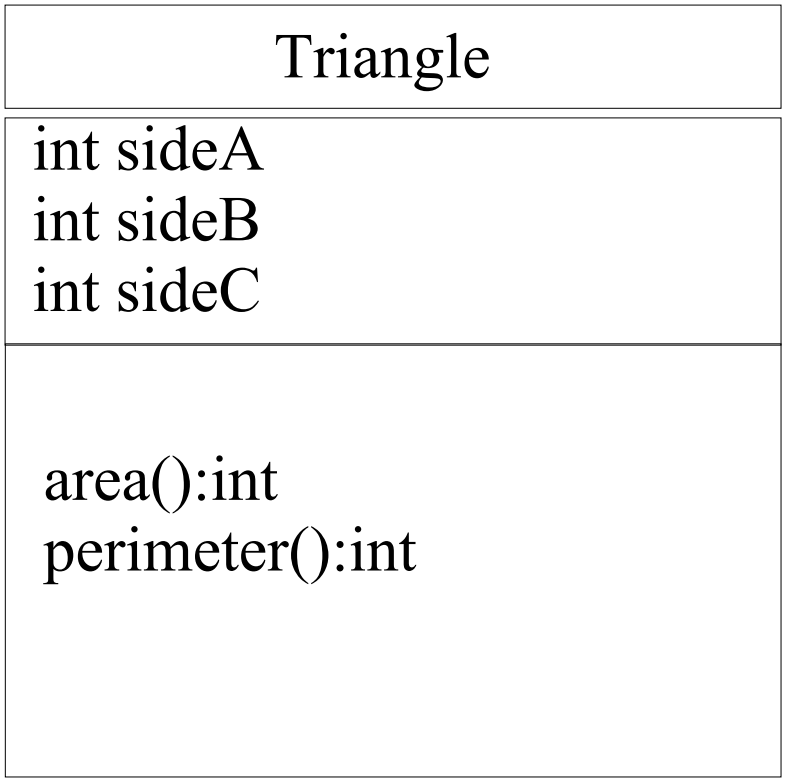
- **Execution Sequence**
  - start inside main method
    - we first create an **instance** of `Triangle` using the class definition as a prototype
  - order of commands is specified by (red) numbers on arrows
  - dashed arrows denote returned values

Let's see some output (cont)



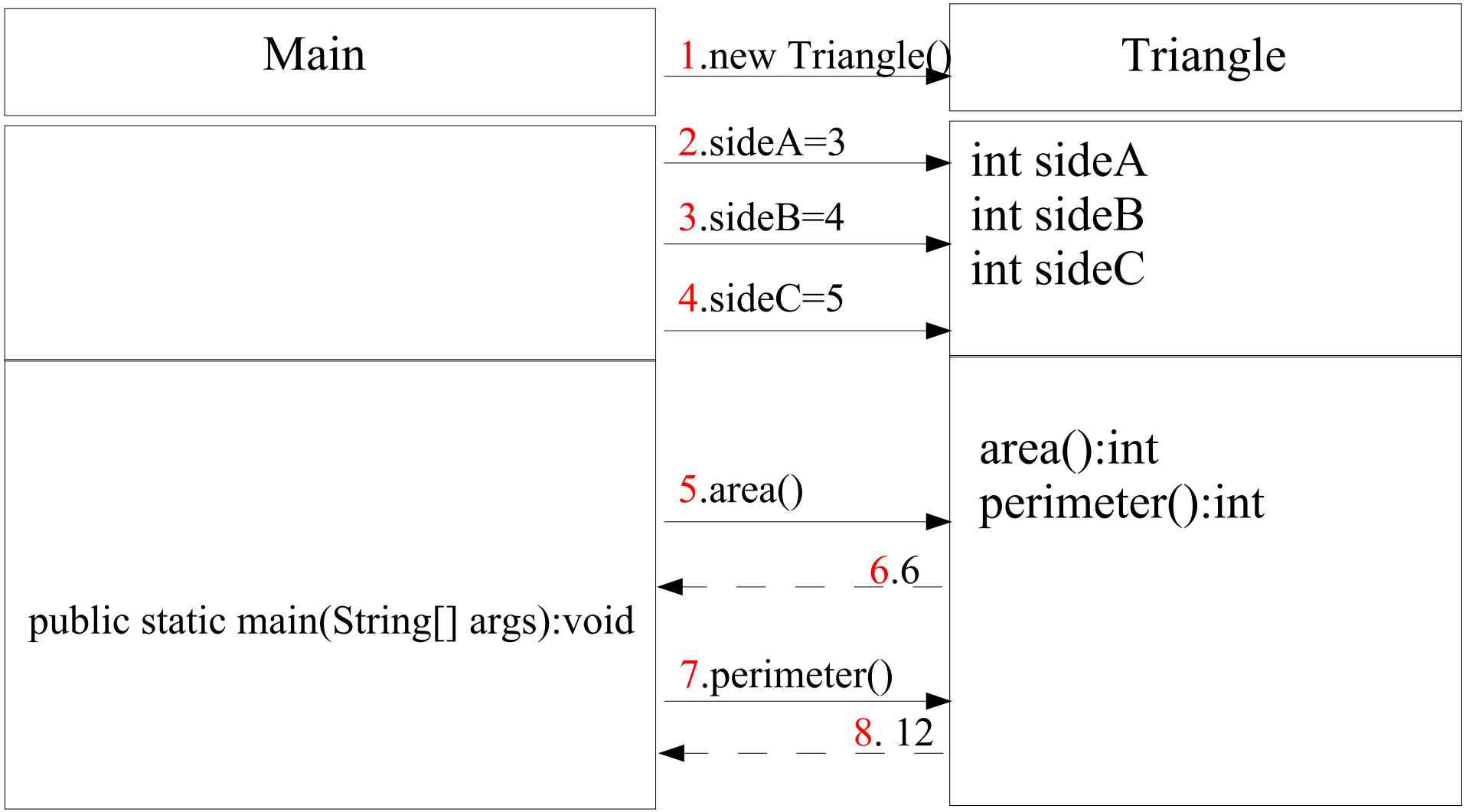

```
public class Main{

  public static void main(String[] args){
   // create an instance of Triangle called "aTriangle"
   Triangle aTriangle = new Triangle();
   int area ; // to store the result of area
   area = aTriangle.area();
   System.out.println("Triangle 3 4 5 has area : " + area);
   int perimeter ;
   perimeter = aTriangle.circumference();
   System.out.println("Triangle 3 4 5 has perimeter : " +
                                perimeter);
  }
}
```

Java Types (cont)




```
public class Triangle {

   public int sideA;
   public int sideB;
   public int sideC;

   public int area(){
      int result = 0;
      result = (sideA*sideB)/2;
      return result;
   }

   public int perimeter(){
      int result = 0;
      result = sideA + sideB + sideC;
      return result;
   }
}
```

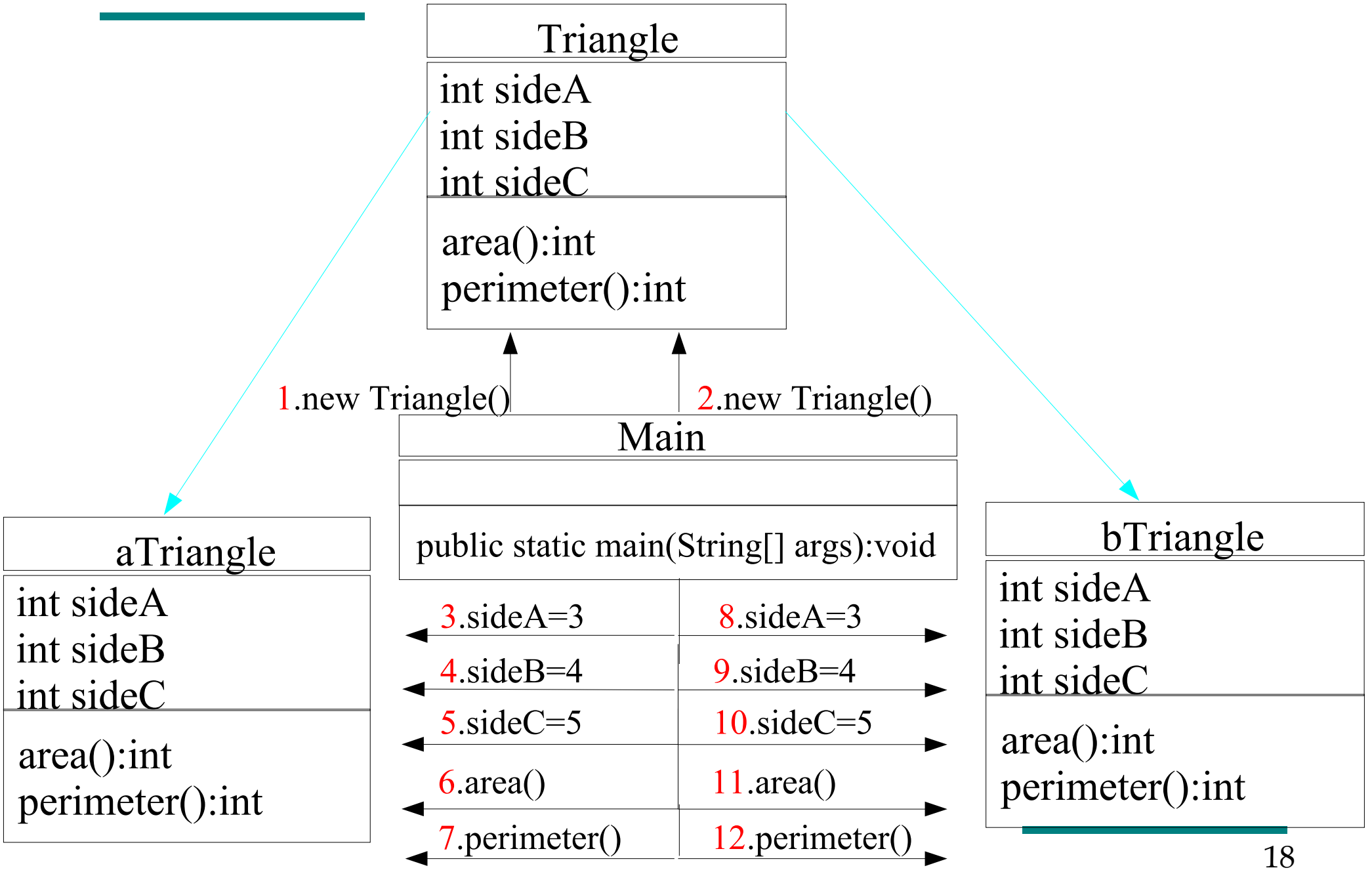
Let's see some output again



Let's see some output again (cont)

```
public class Main{

  public static void main(String[] args){
   //create a Triangle instance
    Triangle aTriangle;
    aTriangle = new Triangle();
   // initialize the three sides to 3,4,5
    aTriangle.sideA = 3;
    aTriangle.sideB = 4;
    aTriangle.sideC = 5;
   //get area and print it
    int area ;
    area = aTriangle.area();
    System.out.println("Triangle 3 4 5 has area : " +
                        area);
    int perimeter ;
    perimeter = aTriangle.circumference();
    System.out.println("Triangle 3 4 5 has perimeter :
                        " + perimeter);
  }
}
```

Playing with more objects



Playing with more objects (cont)

```
public class Main{

  public static void main(String[] args){

   Triangle aTriangle;
   Triangle anotherTriangle;
   aTriangle = new Triangle();
   anotherTriangle = new Triangle();

   aTriangle.sideA = 3;
   aTriangle.sideB = 4;
   aTriangle.sideC = 5;

   anotherTriangle.sideA = 5;
   anotherTriangle.sideB = 12;
   anotherTriangle.sideC = 13;

   int area ;
   area = aTriangle.area();
   System.out.println(
    "Triangle 3 4 5 has area : " +
     area);
   int perimeter ;
   perimeter = aTriangle.perimeter();
   System.out.println(
       "Triangle 3 4 5 has
        perimeter : " + perimeter);

   area = anotherTriangle.area();
   perimeter =
       anotherTriangle.perimeter();
   System.out.println(
       "Triangle "+
       anotherTriangle.sideA +
      " "+anotherTriangle.sideB+
      " "+ anotherTriangle.sideC +
      " has area : " + area);
   System.out.println(
       "Triangle "+
       anotherTriangle.sideA +
        " "+anotherTriangle.sideB +
       " "+ anotherTriangle.sideC +
        " has perimeter : " + perimeter);
  }
}
```